Cilt I

Adım Adım

LaTeX

Önsöz

Günümüzde uluslar arası bir çok bilimsel dergide ve bir çok kitap basımında LaTeX (TeX, $\mathcal{AMS}$-TeX, $\mathcal{AMS}$-LaTeX, ...) biçimli yazılar istenmektedir. Fakat bilim adamlarımızın (La)TeX yazım bilgisine sahip olmadığını görmek bizi fazlasıyla üzmüştür. Bu soruna bir çözüm getirmek için (La)TeX hakkında Türkçe ayrıntılı bir kitap yazılması fikri bizde oluşmuştu. Bu amacla, işte, bu kitap yazılmıştır.

(La)TeX 'de herhangi bir belge yazmanız için bu kitapta ele alınan konular yeterlidir. Kitapta (La)TeX 'i sadece kullanmanız için en az bilmeniz gereken konular yer almıştır. Bu konular kısaca şöyledir:
(La)TeX 'de

- bir belge yapısının oluşturulması;
- bir belgenin metin biçimi;
- bir belgede matematik metin oluşturulması.

Son zamanlar (La)TeX çok hızlı şekilde gelişmektedir. Bu kitaptaki bilgiler, aslında, (La)TeX 'e ait bilginin tahminen %40 'dır. Bundan dolayı, bu kitabın devamı olarak ikinci kısmının da yazılımı bitmek üzere, umarız ki kitabın her iki kısmınıda çok sayıda okuyucunun talebi olacaktır. Kitabın ikinci kısmına ait bir duyuru verelim. İkinci kısımda yer alacak konular kısaca şöyledir:
(La)TeX 'de

- derlemede hatalar;
- bir belge metninde şekil ve grafik çizilmesi;
- bir belge metnine dışarından bir resim ve/veya grafik eklenmesi;
- çeşitli renklerin kullanılması (**Renkli (La)TeX**);

- özel seçenekler oluşturulması;
- **html** sayfalarına uyarlanması.

(LA)TEX programı lisans ücreti ödemeden kullanabileceğiniz, kamusal bir programdır. Dolayısıyla bu program hakkında Internet'ten istediğiniz kadar bilgi bulabilir veya bu programın kendisini de indirebilirsiniz. Internet'te TEX 'in

"The Comprehensive TeX Archive Network " (kısaca CTAN)

olan özgü bir sayfası vardır:

http://www.ctan.org/

Web-sayfasının adından da belli olduğu üzere, burada TEX 'e ait tüm bilgiler arşivi toplanmaktadır. Aradığınız herhangi bilgi veya ek programı bu arşivden bulabilirsiniz, hatta TEX programının kendisini de bu sayfa aracıyla Internet'ten indirebilirsiniz. Üstelik, eğer TEX için kendi ürettiğiniz bir şeyler varsa, bunları herkesin kullanması için bu web-sayfasına gönderebilirsiniz.

Prof.Dr. Abdugafur Rakhimov

Öğr.Gör. Orhan Kesemen

İÇİNDEKİLER

I Adım Adım LaTeX **i**

Önsöz **iii**

Giriş. TeX 'in temelleri **xv**
0.1 TeX, LaTeX 'in tarihi xv
0.2 TeX, LaTeX nedir? xvi
0.3 Neden, nerede ve ne zaman TeX, LaTeX? xviii
0.4 TeX 'in biçimleri xx
0.5 TeX 'de hatalar xxii
0.6 TeX'in kullanılmasında neler gereklidir xxii
0.7 Kitap nasıl kullanılmalıdır xxiii

1 Giriş dosyası **1**
1.1 LaTeX 2_ε ve LaTeX 2.09 'de belgenin başlık kısmı 1
1.2 Belgenin metni 2
1.2.1 Başka dosyadan metin eklemek 3
1.3 Standart sınıfların seçenekleri 4
1.4 LaTeX 'de Türk dili 6
1.5 Açıklama 7
1.6 Dosyayı yazdırmak 8
1.6.1 Sayfa seçenekleri 8
1.6.2 İlk sayfa 10
1.7 Kısım, bölüm ve paragraf 12
1.7.1 Kısım 13
1.7.2 Bölüm 14
1.7.3 Paragraf 15
Satırbaşı 15
1.7.4 Kitabı kısımlara bölmek 15
Uygulama 16

1.8 İçindekiler, Şekil ve Tablo Listesi . . . 17
1.8.1 İçindekiler . . . 17
1.8.2 Şekil ve tablo listesi . . . 18
1.9 Üst ve alt bilgi . . . 19
1.9.1 Standart olmayan üst ve alt bilgi . . . 21
1.10 Çapraz gönderme . . . 21
1.10.1 varioref paketi . . . 22
1.10.2 xr paketi . . . 23
1.10.3 hyperref paketi . . . 24
Aşırı-gönderme metni . . . 25
Aşırı-gönderme metninde bölüm adı . . . 26
Başka dosyaya aşırı-gönderme . . . 27
Aşırı-gönderme renkleri . . . 28
1.11 Sayfa Dipnotu . . . 29
1.12 Kaynakça . . . 31
1.13 Sayfa numarası . . . 33
1.14 Birkaç sütunlu metin . . . 34
1.14.1 multicol paketi . . . 35
1.15 Dizin . . . 36
1.15.1 Dizin tabanı . . . 37
1.15.2 Dizinde yönlendirici (!, @ ve |) simgeler . . . 40
1.15.3 Türkçe Dizin . . . 40

2 Metin biçimi **43**
2.1 TEX 'in ölçü birimleri . . . 43
2.2 Boş aralık koymak . . . 44
2.2.1 Yatay aralık . . . 44
2.2.2 Düşey aralık . . . 46
2.3 Paragraf biçimi . . . 46
2.3.1 Biçim düzeninin değiştirilmesi . . . 47
2.3.2 Satır başı boşluğu . . . 48
2.3.3 Satırlar arasındaki boşluk . . . 48
2.3.4 TEX 'de sözcük hecelemesi . . . 49
2.3.5 Satır kesilmesi . . . 50
2.4 Paragraf biçimini yönetmek . . . 51
2.4.1 Metinin sayfalara bölünmesi . . . 51
2.4.2 Metin yüksekliğinin değiştirilmesi . . . 52
2.4.3 Yeni sayfa başlatmak . . . 52
2.5 Madde işaretleri ve numaralandırma . . . 53
2.5.1 İşaretli listeler . . . 53

2.5.2 Numaralı listeler . . . 54
2.5.3 Tanımlama listesi . . . 56
2.5.4 Özel listeler . . . 57
2.5.5 Sade liste . . . 59
2.6 Değişkenler . . . 60
2.7 Özel paragraflar . . . 62
2.7.1 Satırda metin yerini belirtmek . . . 62
2.7.2 Ayırtılan metin . . . 63
2.7.3 Matbaa harfli metin . . . 63
2.7.4 shortvrb paketi . . . 64
2.7.5 alltt paketi . . . 65
2.8 Kutu'lar . . . 66
2.8.1 Satır kutuları . . . 66
2.8.2 Kutunun düşey kaydırılması . . . 67
2.8.3 Kutunun önceden hazırlanması . . . 68
2.8.4 Metin kutusu . . . 69
2.8.5 Dolu kutular . . . 70
2.9 Sekmeler . . . 71
2.10 Tablolar . . . 74
2.10.1 tabular bloğu . . . 74
2.10.2 array paketi . . . 77
2.10.3 Verilen genişlikteki tablolar . . . 79
2.10.4 Birkaç sayfalı tablolar . . . 81
2.11 Kayan nesneler . . . 84
2.11.1 Şekil ve Tablolar . . . 84
Sayfaya kayan nesnenin yerleştirilmesi . . . 86
Nesneden önce ve sonra koyulacak düşey aralık . . . 87
2.11.2 Akan metinli şekil ve tablolar . . . 88
2.11.3 Boşluk notu . . . 90
2.12 Metin yazısı . . . 91
2.12.1 Yazı özellikleri . . . 91
2.12.2 Yazı tipi . . . 92
2.12.3 Yazı doygunluğu . . . 93
2.12.4 Yazı biçimi . . . 94
2.12.5 Yazı boyutu . . . 94
2.12.6 İstenilen yazı tanımı . . . 95
2.12.7 Metinin taban yazısı . . . 97
2.13 Yeni makro tanımlar . . . 98
2.13.1 Komutlar . . . 98
2.13.2 Komutlu parantezler . . . 101

2.13.3 Makro tanımların *-yıldızlı şekli . . . 103
2.14 Simgeler . . . 103
2.14.1 Yardımcı simgeler . . . 103
2.14.2 Avrupa alfa-sayısalın ulusal simgeleri . . . 103
2.14.3 Tırnak işareti . . . 105
2.14.4 Ek simgeler . . . 106
2.14.5 textcomp paketi . . . 106

3 LaTeX 'de matematiksel formüller 109
3.1 Matematik blokları . . . 109
3.1.1 Metin içinde formül . . . 109
3.1.2 Metinde uzun satırların engellenmesi . . . 112
3.2 Metinden ayrılan formüller . . . 114
3.2.1 Tek satırlı formüller . . . 114
3.2.2 Denklem sistemleri . . . 117
3.2.3 Uzun formüllerin bölünmesi . . . 121
3.2.4 Bir satırda bağımsız denklemler . . . 123
3.2.5 Çok satırlı formülün bölünmesi . . . 126
3.3 Simgeler arasındaki boşluk . . . 127
3.4 Simge boyutu . . . 129
3.5 Matematiksel simgeler . . . 130
3.5.1 Üst-alt indis ve ayraçlar . . . 130
3.5.2 Üç nokta . . . 130
3.5.3 Aritmetik işlem simgeleri . . . 131
3.5.4 Mantıksal ilişki simgeleri . . . 132
3.5.5 Yunanca harfler . . . 134
3.5.6 Noktalama işaretleri . . . 136
3.5.7 Vurgular . . . 136
3.5.8 Köklü ifadeler . . . 137
3.5.9 Kesirli ifadeler . . . 139
3.5.10 Limitlerle operatörler . . . 141
İntegraller . . . 143
Çok satırlı indis . . . 144
3.5.11 Parantez ve başka ayraçlar . . . 146
Değişen boyutlu parantezler . . . 146
Ayraçlar . . . 147
Ayraç boyutunu tanımlayan komutlar . . . 148
3.5.12 Oklar . . . 150
3.5.13 Karışık simgeler . . . 151
3.6 Harf üstü ve harf altı simgeleri . . . 152

3.6.1 Şapka ve dalgalar . . . 152
3.6.2 Üst ve alt çizgiler . . . 153
3.6.3 Küme parantezleri . . . 153
3.6.4 Oklar . . . 154
3.6.5 Keyfi simgeler . . . 155
3.7 İndisli oklar . . . 155
3.8 $\mathcal{AMS}$ 'ın Binom katsayıları . . . 156
3.9 Matematiksel fonksiyonlar . . . 157
3.9.1 Logaritmik ve trigonometrik fonksiyonlar . . . 157
3.9.2 Limitli fonksiyonlar . . . 157
3.9.3 Yeni işlev adını tanımlamak . . . 158
3.9.4 Modül fonksiyonu . . . 160
3.10 Matris tipli şablonların yapısı . . . 160
3.10.1 Matrisler . . . 160
3.10.2 $\mathcal{AMS}$ 'ın matrisleri . . . 165
3.10.3 Kesir tipli bir formül . . . 167
3.10.4 Parçalı fonksiyonlar . . . 168
3.11 Yazı stilleri . . . 169
3.11.1 Formül içinde metin . . . 169
Denklemler arasına metin koyulması . . . 170
3.11.2 Matematik alfa sayısalı . . . 170
3.11.3 Simgelerin koyu doygunluğu . . . 172
3.12 Formüllerin ayarlanması . . . 173
3.12.1 Matematik bloklarında boşluklar . . . 173
3.12.2 Hecelemede işleçlerin çiftlenmesi . . . 175
3.12.3 Bölünmez tire . . . 176
3.12.4 Görünmez simgeler . . . 177
3.12.5 Yer tutmayan görünen simgeler . . . 178
3.13 Kuram, önerme ve tanımlar . . . 179
3.14 Denklemlerin ek numaralandırılması . . . 182
3.14.1 Denklemlerin özel numaralandırılması . . . 184
3.15 Değişik formüller . . . 185
3.15.1 Diyagramlar . . . 185
3.15.2 Çerçeveli formüller . . . 185
3.15.3 Formül boyutunun değiştirilmesi . . . 186
3.15.4 Ayraç boyutunun değiştirilmesi . . . 187

ŞEKİL LİSTESİ

1.1 Sayfa parametre boyutunu tanımlayan komutlar. 9

2.1 Sayfada bir liste modeli 57
2.2 `floatingfigure` bloğu 90

TABLO LİSTESİ

2.1 TEX ve LATEX 'in ölçü birimleri 43
2.2 `floatingtable` bloğu 90
2.3 Yazı boyutun komutları 95
2.4 Latince'de olmayan Türkçe harfler 104
2.5 Latince'de olmayan simgeler 104
2.6 Avrupa alfa-sayısalın önemli simgeleri 104
2.7 `T1` şifreli önemli simgeler 105
2.8 Tırnak işaretleri 105
2.9 `T1` şifreli tırnak işaretleri 105
2.10 Özgü simgeler 106
2.11 Metinde matematiksel simgeler 106
2.12 textcomp paketinin matematiksel simgeleri 106

3.1 Üç noktalar 130
3.2 Aritmetik işlem simgeleri 131
3.3 $\mathcal{AMS}$'ın aritmetik işlem simgeleri (amssymb paketi) 132
3.4 Mantıksal ilişki simgeleri 132
3.5 $\mathcal{AMS}$'ın mantıksal ilişki simgeleri (amssymb paketi) 133
3.6 $\mathcal{AMS}$'ın karşıt mantıksal ilişki simgeleri (amssymb paketi) 134
3.7 Yunanca harfler 135
3.8 Yunanca büyük harfler 135
3.9 $\mathcal{AMS}$'ın Yunanca harfleri (amssymb paketi) 136
3.10 Noktalama işaretleri 136
3.11 Matematiksel blokta vurgular 137
3.12 Limitli operatörler 142
3.13 Ayraçlar 147
3.14 Büyük ayraçlar 147
3.15 $\mathcal{AMS}$'ın ayraçları 148
3.16 Özel ayraçlar 149
3.17 Oklar 150

3.18 $\mathcal{AMS}$'ın okları (amssymb paketi) 151
3.19 $\mathcal{AMS}$'ın karşıt okları (amssymb paketi) 151
3.20 Metinde ve matematik blokta kullanılan simgeler 151
3.21 Ek simgeler 152
3.22 $\mathcal{AMS}$'ın ek simgeleri (amssymb paketi) 152
3.23 Logaritmik ve trigonometrik fonksiyonlar 157
3.24 Limitli fonksiyonlar 158
3.25 $\mathcal{AMS}$'ın limitli fonksiyonları (amsopn paketi) 158
3.26 Matematik alfa sayısalı 171
3.27 $\mathcal{AMS}$ 'ın matematik alfa sayısalı 171
3.28 RSFS 'in matematik alfa sayısalı (mathrsfs paketi) 171
3.29 Matematik bloklarında boşluklar 174

Giriş

TEX 'in temelleri

0.1 TEX, LATEX 'in tarihi

1970'li yıllarda Standford Üniversitesi'nde Bilgisayar Bilimcisi olan Donald E. Knuth yazdığı "*Bilgisayarda programlama sanatı*" (The Art of Computer Programming) adlı kitabı yüksek kalitede bastırmak için uygun bir basım evi bulamamış. Bu durumda, kitabını yüksek kalitede yazdıracak bir bilgisayar programını kendisi yazmaya karar vermiş. O yıllarda Bilgisayar Bilimcisi D.E.Knuth bu programı TEX adıyla üretmiş ve bu güne kadar bu program lisans ücreti ödemeden kullanabileceğiniz, kamusal bir program olmuştur.

TEX kelimesinin kökü Yunanca $\tau\epsilon\chi$ (tau, epsilon, chi) harflerin üçlüsünün köküne benzetilir. Yunanca'da bunlar *teknoloji* ve *sanatı* anlatır. Üstelik, bu harfler "technology" kelimesin kökünden de kaynaklanır.

TEX 'in ilk sürümü kamuda 1982 'li yıllarda kullanmaya başlamıştır. Onun şimdiki geçerli sürüm numarası `3.14159` 'dur. TEX 'in her yeni sürümü bir önceki sürüm numarasına π sayısının sıradaki ondalısının eklenmesi kabul edilmiştir. Dolayısıyla, TEX 'in sıradaki yeni sürümü `3.141592` olacaktır. Bunun anlamı şu ki, TEX 'in her yeni sürümü önceki tüm sürümlerini içermesi gerekir, yani bir sürümde yazılan herhangi bir belge bu sürümden sonradaki tümünde geçerli olmalıdır. Böylece, TEX D.E.Knuth tarafından dondurulmuştur, yani TEX 'in bir sürümü program temel ve yapısını değiştiremez. Aksi halde, bu programa TEX adı verilemez. Dolayısıyla, programın her sürümü TEX= TEX eşitliğini sağlamalıdır.

LATEX programı 1985'li yıllarda Bilgisayar Bilimcisi olan Leslie Lamport tarafından yazılmış TEX 'in bir özel sürümüdür. Programın adındaki "La" kelimesin kökü İngilizce "Lay" (koymak, yatırmak) kelimesinin anlamından kaynaklanır.

LATEX programı

- TEX 'i daha da iyi bir duruma getirmekte TEX= TEX eşitliğini sağlayan bir yarışmayı başlatan bir sürümdür. Bu yarışmada, bir sürümün amacı TEX 'i mümkün olduğunca kolay ve basit anlaşılan duruma getirmek ve TEX 'in kullanılma oranını mümkün olduğunca genişletmektir.
- TEX 'e yeni komut ve tanıtımlar ailesini eklemektedir;
- TEX 'in bazı komutlarına yeni seçenekler eklemektedir;
- TEX 'in çok sayıda komutlarının imkan dairesini genişleterek, komutlu paranteze dönüştürmektedir;
- TEX 'in başka sürümüne ait komut ve tanıtımlardan yararlanmasına izin vermektedir;
- metine herhangi bir paketi yükleyerek, program tabanını genişletebilir, yani pakette tanımlanan tüm komutlarının metinde kullanılmasına izin vermektedir.

LATEX 'de yazılan bir metin TEX 'de yazılan metine göre daha kolay anlaşılmaktadır. LATEX 'in ilk yazılan LATEX2.09 (ve EmTEX) sürümü son zamanlarda pek fazla kullanılmamaktadır. Çünkü, LATEX 'in yeni LATEX 2_ε, $\mathcal{AMS}$-LATEX, 4TEX, ... sürümleri LATEX 'i yine de kolay ve basit anlaşılan duruma getirmektedir ve onun kullanılma imkan derecesini de iyice genişletmektedir.

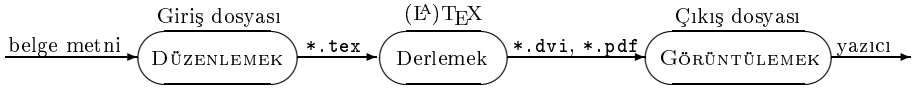
0.2 TEX, LATEX nedir?

(LA)TEX, MS DOS, Linux ve Windows ortamlarında çalışan, çok hızlı ve kolay bir şekilde en yüksek kalitede bir kitap, makale ve herhangi bir belgeyi üretmenize olanak veren bir programdır. (LA)TEX 'de bir belgeyi hazırlanması "Düzenlemek-Derlemek-Görüntülemek " olarak üç adımdan oluşmaktadır:

* "Düzenlemek" 'te belgenin metni (*giriş dosyası*) yazılır;
* metin (LA)TEX 'de "derlenir", yani metin programda okutulur. Bu durumda, (LA)TEX belgenin bir *çıkış dosyası*nı üretir;
* çıkış dosyası "görüntülenir" ve/veya yazıcıda yazdırılır.

TEX 'de yazılacak bir belge dosyasının uzantısı "`*.tex`" olarak kabul edilmektedir. Bu dosyaya belgenin **giriş dosyası** denir. Belgenin giriş dosyası herhangi bir düzenleme programında yazılabilir, hatta "Not Defterinde" de yazılabilir. Bir metin, genelde, PFE, WINEDT ve MED programlarının birinde yazılır. Bu programların penceresi, genelde, Windows Word 'ün bir belge penceresine çok benzerdir ve araç çubuklarının işlevi de hemen hemen aynıdır. Bundan dolayı, bu kitapta, (LA)TEX 'de düzenleme programının kullanılması ele alınmayacaktır.

Belgenin `*.tex` uzantılı giriş dosyası (LA)TEX 'de derlendiğinde (LA)TEX `*.dvi` (veya `*.pdf`) uzantılı bir çıkış dosyasını üretir. Bu dosya (LA)TEX 'de görüntülenir ve yazıcıda yazdırılabilir:



(LA)TEX 'in, bilinen `Pascal`, `C`, `C++`, ... programları gibi, kendi komutları vardır. Belgenin giriş dosyası düzenleme programında bu komutlar yardımıyla yazılır [1], yani metinde her şey (LA)TEX 'in komutlarıyla ayarlanır. Giriş dosyasının yazılması için (LA)TEX 'in tüm komutlarını bilmeniz gerekmez, sadece normal bir metinin yazılması için gerekli olan az sayıda komutlarını bilmeniz yeterlidir.

Bir belgenin yapısı, genelde,

title:	başlık
author:	yazar
table of contents and list of tables	içindekiler ve tablolar listesi
chapter, section, ...	bölüm, paragraf, ...
appendix:	ek

[1] Düzenleme programı, genelde, (LA)TEX 'in komutlarını tanır, yani program `Pascal`, `C`, `C++` programları gibi metinde doğru yazılan komutun rengini otomatik belli bir renge (mavi renge, kırmızı ...) dönüştürür

bibliography:	kaynakça
index	dizin

gibi kısımlardan oluşmaktadır. (LA)TEX bu kısımların her birini özel komutlarla ayarlayarak bir bütün olarak belgeyi oluşturmaktadır. Üstelik, (LA)TEX bu kısımlar arasında herhangi bir bağlantı da yapabilmektedir. Bir belge sadece bazı kısımlardan oluşuyor olabilir, yani bir belgede her kısmın varolması şart değildir. Dolayısıyla, çok basit bir belge (mektup, özet, formülsüz bildiri, ...) (LA)TEX 'in sadece 6-8 tane komutuyla yazılabilir.

0.3 Neden, nerede ve ne zaman TEX, LATEX?

Dünyanın bütün Bilgisayar Bilimcileri (LA)TEX 'in diğer programlardan daha da üstün olduğunu bilmektedirler.

Neden (LA)TEX tercih edilmektedir? Çünkü:

- ✓ *Yüksek kaliteli çıktı*: TEX çok yüksek kaliteli çıkış dosyası üretir. Bunu görmeniz için sadece çıkış dosyası yazıcıda yazdırılıp, başka bir programda (örneğin Word'de) yazdırılan bir belge ile kıyaslanmalıdır. Üstelik, TEX çıkış dosyayı `Adobe Acrobat` (`*.pdf`) ve `PostScript` (`*.ps`) uzantılı olarak da üretebilir.

- ✓ *Tutarlılık*: TEX dondurulmuş bir program olup, yıllar geçince onun temeli ve yapısı değişmeyecektir. Dolayısıyla, TEX 'de yazılan bir belge bir dosya olarak yıllarca kendi geçerliliğini koruyabilir. Üstelik, bir belge keyfi boyutlu olarak oluşturulabilir, hatta binlerce sayfalı da olabilir ve TEX bu dosyayı belleği çok düşük olan bilgisayarda da yazdırabilir.

- ✓ *Programlaştırılması*: TEX birkaç basit komutlar ile inanılmaz karmaşık şeyler yapabileceğiniz bir makro dildir. Hatta kendi özel bir uygulamanız için, gerekirse, TEX 'de her şeyi yeniden tanımlayabilirsiniz.

- ✓ *Uysallık*: TEX dünyanın tüm yerinde, gerekirse, bazı komutları yeniden tanıtılarak, çeşitli belgeler yazmak için kullanılmaktadır. Örneğin, TEX 'de bir metin istenilen dilde yazılabilir, hatta Arapça'da olduğu gibi metin sağdan sola veya büyük karakterli Çince harfler de yazdırılabilir.

- ✓ *Basitlik*: TEX 'in bir belgesi Plain ASCII 'de yazılır. Dolayısıyla, TEX programı olmayan bir bilgisayarda da bu belge herhangi bir `edit` programında okunabilir. Üstelik, belgenin dosyası bozulmuşsa (örneğin, bozuk bir diskete kopyalansa), bu dosyanın özel bir biçimi olmadığından kolayca onarılabilir. Zaten bundan dolayı, Internet hattında veya e-maillerde dosya olarak TEX 'in dosyası tercih edilir.
- ✓ *Ulaşabilirlik*: TEX aşağı yukarı tüm bilgisayar sistemine kurulabilen bir programdır. Örneğin, Atari, Apple Macintosh, Unix, VMS, CMS, MVS, MS DOS, OS/2, LINUX ve Microsoft WINDOWS ortamları TEX 'i desteklemektedir. Bir ortamda yazılan bir belge diğer bir ortamda da geçerlidir.
- ✓ *Ucuzluluk*: TEX lisans ücreti ödemeden kullanabileceğiniz, kamusal bir programdır. Sadece bazı özel araç veya edit programları, aynı zamanda programa ait kitaplar ücretlidir, bu kadar. Bundan başka, programın ticari amaçla yazılan bazı özel sürümleri de ücretlidir.
- ✓ *Geniş destek*: TEX 'in lisansı olmadığından, program sürümlerini üreten ve destekleyen belli bir şirket yoktur. TEX ücretsiz olarak, genelde, Internet, e-mail ve gönülülerce desteklenmektedir.

Nerede ve ne zaman (LA)TEX kullanılır (üstünlüğü ve eksikliği):

- ▲ TEX programı bilimsel çevrelerde çok ünlüdür. Özellikle, dünyanın tüm matematikçileri bir bilimsel çalışmayı (kitap, makale, ...) TEX 'de yazmaktadırlar. Program bir matematiksel formül için gerekli font türü, yazı boyutu ve tiplerini tümünü desteklemektedir.
- ▲ Eğer oluşturulacak bir belgenin mantıklı bir doğal yapısı varsa, TEX bu yapıyı güzel bir şekilde oluşturabilir. Örneğin, TEX bir kitabın mantıklı doğal: "içindekiler, önsöz, bölüm, paragraf (dip not, boşluk notu, ...), ek, dizin, kaynakça" yapısını kolayca oluşturabilir.
- ▼ TEX programı, "M.S.Office, ... " 'in bazı programları gibi önemsiz olarak öğrenilemez. Eğer TEX 'i sadece tesadüfen kullansanız, bu programı öğrenmek isteğiniz de vazgeçemeyeceksiniz. Dolayısıyla, TEX programı büyük veya düzenli çalışmalar yapılması için öğrenilir ve kullanılır.
- ▲ Bir belge çok sayıda referans (ve/veya metin, formül, dizin, ...) göndermesi, dip notu, boşluk notu ve kaynakça elemanlarını içerirse,

bu belge TEX 'de yazılmalıdır. TEX bu yapıların hepsini otomatik kendisi ayarlar.

▼ TEX kendi "aklı" ile herhangi bir fontu üretebilir. Fakat bu program bilgisayar işletim sisteminin fontlarını kullanmayacaktır. Bir belgede TEX sadece kendi ürettiği özel fontları kullanır. Ancak bundan tasa çekmemek gerekir, çünkü istediğiniz her şeyi (bir fontu da) çağdaş bir TEX programıyla yapabilirsiniz.

▲ Yıllar geçince TEX daha da kaliteli ve güzel görüntülü çıktı vermektedir. TEX bir belgedeki her paragraf ve onun biçimini özel olarak ayarlar; iki kelime arasına gerekli boşluk koyar; bir kelime (veya formül, ...) hecelemesini de doğru yapmaktadır.

▼ Bir belgede bir resim işleme TEX 'in komutlarıyla yapılamaz. Bir resim işleme yapan özel programlar vardır. Fakat bu programlar sadece bu amaçla üretilmiştir. Dolayısıyla, birkaç resimi (ve/veya şekili) içeren bir belge bu özel programlarla da oluşturulamaz.

▲ TEX bir belgeye dışarından bir resim, şekil ve/veya tablo gibi nesneleri ekleyebilir; hatta bu nesneyi metinde bir kayan nesneye de dönüştürebilir (*bkz.* 2.11.1, 84.sayfa).

▲ Eğer bir belge başka dillerde yazılan birkaç parçayı da içerirse, bu belge en iyisi TEX 'de yazılmalıdır. Çünkü, TEX dünyanın hemen hemen tüm dillerini iyi şeklinde desteklemektedir.

▼ Yukarıda söylendiği gibi, TEX çok kaliteli ve güzel çıktılı bir belge oluşturur. Fakat, bazı dergi ve yayın evi hala TEX dosyasını kabul etmemektedir. Bu dosya, genelde, maalesef başka bir programa (M.S. Word, ...) da çevrilemez. Örneğin, TEX 'de `$\root 4 \of {\beta ^\pi}$` olarak yazılan $\sqrt[4]{\beta^\pi}$ formülünü başka bir program böyle yazdıramaz.

▲ Aşağı yukarı tüm bilimsel dergi ve yayın evi bir kitap ve makale için yazardan sadece TEX dosyasında talepte olmaktadır.

0.4 TEX 'in biçimleri

TEX 'in biçimi bir belgenin giriş ve çıkış dosyasının yapısına göre alınmalıdır. Örneğin, eğer belge metni LATEX 2_ε 'de yazılıp, çıkış dosyasını `*.pdf` uzantılı olmasını istiyorsanız, belge giriş dosyasını TEX 'in PDFLATEX

biçiminde derlemek gerekir.

TEX, genelde, aşağıdaki biçimleri desteklemektedir:

Plain TEX D.Knuth 'un standart makro paketidir.

ε-TEX Plain TEX biçiminin tüm fonksiyonlarını sağladığı halde TEX sürümünü yükseltir. ε-TEX bu biçimi Plain TEX makro paketinin genişletilmesi olan "eplain" biçimiyle karıştırılmayacaktır.

LATEX L.Lamport 'un standart makro paketidir.

PDFTEX Plain TEX ile aynıdır, fakat bu biçim `*.pdf` uzantılı çıkış dosyayı üretir.

Expanded Plain TEX Plain TEX 'i kitap, makale ... gibi belgeyi yazılmasında yararlı olan komutlarla genişletir.

PDFLATEX LATEX ile aynıdır, fakat bu biçim `*.pdf` uzantılı çıkış dosyayı üretir.

CONTEXT H.Hagen 'in makro paketidir. Paketin üç versiyonu vardır: *English*, *German* ve *Dutch*; geçerli yapı için ise *Czech* versiyonu alınır. CONTEXT biçimi ya `*.dvi` ya da `*.pdf` uzantılı çıkış dosyasını üretir.

LATEX2.09 LATEX 'in eski bir sürümüdür; LATEX 2_ε 'in yeni sürümlerinde iyi çalışmayabilir. LATEX 2_ε bu biçimin bazı komutlarını (`\proclaim` ...) desteklememektedir. Dolayısıyla, bu formatın kullanılması önerilmemektedir.

BLUe K. van der Laan tarafından yazılmış Plain TEX tabanlı makro pakettir.

MeX Plain TEX 'in Lehçe versiyonudur.

PLATEX Sadece İngilizce ve Lehçe için iki sözcük arasındaki çizgi örneğini içeren ve Babel paketini desteklemeyen LATEX sürümü.

CSPLAIN `CM` fontu yerine `CS` fontu kullanılan Plain TEX 'in küçük bir genişletmesidir. Çekçe ve Slovakça için iki sözcük arasındaki çizgi örneğini de içerir.

CSLATEX Standart LATEX 'i Çekçe ve Slovakçalaştırır.

TEXinfo TEX 'in help info'sunu genişletir.

FRLATEX LATEX biçimi TEX 'in *French-speaking* seçeneğine ayarlanır.

0.5 TEX 'de hatalar

Bir belgenin metni hatasız yazılmışsa, onun giriş dosyası TEX 'de derlendiğinde TEX `*.dvi` (veya `*.pdf`) uzantılı bir çıkış dosyasını üretir. Eğer metinde bazı hatalar varsa, TEX derlemede bir hatayı şöyle bildirir:

```
! LaTeX Error: Missing \begin{document}
1.415 \begin
            [document
?
```

Görüldüğü gibi, TEX bir `!` ünlem işaretini yazdırıp, bu işaretten hemen sonra hatanın tipini ve ikinci satırda ise hatanın yatay ve dikey yerini göstermektedir. Örnekte, TEX metinin `415`. satırında yazılan `\begin` komutunun `[` parantezi yanlış olduğunu bildirmektedir. Bildirinin son satırında TEX `?` soru işaretiyle sizden bir yönerge istemektedir. Burada TEX 'e birkaç yönerge verilebilir, örneğin, `x` (exit) yönergesi giriş dosyasında hatayı düzeltmek için programı durdurur; `h` (help) yönergesi hata tipine ait bir bilgi yazdırır (bu yönerge yardıma ihtiyaç duyulduğunda kullanılır); `q` (quit) yönergesi ise programı hataları göstermeden çalıştırır. Bundan başka, tüm hataları görmek için her `?` soru işaretine rastladığınızda klavyenin Enter tuşunu basmalısınız.

TEX 'deki hata tipleri ve onlara verilecek tüm yönergeler kitabın ikinci cildinde ayrıntılı şeklinde ele alınacaktır.

0.6 TEX'in kullanılmasında neler gereklidir

TEX'i kullanmanız için en az aşağıdakiler gereklidir.

- (LA)TEX hakkında temel bir bilgiye sahip olmalısınız. Bunun için bu kitaptaki bilgi yeterlidir. Daha da fazla bilgi almanız için kitabın gelecekte basılacak ikinci cildi size yardımcı olacaktır. Eğer İngilizceniz iyi ise, (LA)TEX hakkında sadece çok sayıda basılmış İngilizce kitaplardan değil, Internet'ten de yeterli derecede bilgi alabilirsiniz.
- Bir bilgisayar. Bilgisayara (LA)TEX 'in herhangi bir versiyonu kurulmuş olmalıdır.
- Bilgisayarda bir Düzenleme programını (örneğin, MED, WINEDT PFE, ...) kullanabilmeniz gerekir.

0.7 Kitap nasıl kullanılmalıdır

Kitabın bu cildi üç bölümden oluşmaktadır. Bu cilt'te verilen bilgiler, (LA)TEX 'de herhangi bir belge yazıp, yüksek kaliteli çıktısını almanız için yeterlidir. Kitabın birinci bölümünde bir belgenin giriş dosyasının yapısı ayrıntılı şekilde ele alınmaktadır. Bu yapı hakkında yukarıda kısaca bahis etmiştik. Dosya yapısını oluşturan (LA)TEX 'in tüm komut ve komutlu parantezleri örneklerle verilmektedir. Kitabın ikinci bölümünde bir belgenin metin biçimi ayrıntılı şekilde ele alınmaktadır. Metin biçimini tanıtan tüm komutlar tanımlanmaktadır. Bu komutların parametreleri nasıl değiştirilmesi gerektiği de verilmektedir. Kitabın üçüncü bölümünde bir matematiksel belge oluşturulması ayrıntılı şekilde ele alınmaktadır. Matematikte tanımlı olan tüm formül ve blokları oluşturan komut ve komutlu parantezler çok sayıda örneklerle verilmektedir. Matematikte kullanılabilen tüm font, simge ve harfleri tanımlayan komutlar cetveli, aynı zamanda $\mathcal{AMS}$-LATEX 'in tüm özel komut ve komutlu parantezleri de ele alınmaktadır.

(LA)TEX 'de bir kitap veya büyük bir makale, bildiri, ... yazmanız için bu kitaptaki tüm bilgiye, eksiksiz, sahip olmanız gerekmektedir. Eğer (LA)TEX 'i sadece küçük bildiri ve/veya makale gibi bir şeyler yazmak için öğrenmek istiyorsanız, size en az kitabın 1.1 - 1.4, 1.6.2, 1.10 ve 1.12 paragraflarını ve üçüncü bölümün ise tümünü iyice okumanızı öneririz (*bkz.* 1 - 6, 10, 21, 31. sayfalar).

Bölüm 1

Giriş dosyası

1.1 LaTeX 2$_\varepsilon$ ve LaTeX 2.09 'de belgenin başlık kısmı

Bir giriş dosyası (belge) başlık kısmıyla başlar. LaTeX 2$_\varepsilon$ ve LaTeX2.09 'de bir belgenin başlık kısmı sırasıyla aşağıdaki *tanıtım*la başlamalıdır

```
\documentclass[options]{class}[release-date]
```

```
\documentstyle[options]{class}[release-date]
```

Bu tanıtımlar çalışmanın hangi sınıfa ait olduğunu bildirmektedir. Bu durumda, LaTeX 2$_\varepsilon$ sürümü *class.cls* dosyasından, LaTeX 2.09 sürümü ise *class.sty* dosyasından komutun tanım ve özelliklerini yükleyecektir. Komutların zorunlu olmayan *option* argümanı bu sınıfın öntanımlı bazı parametreleri ve biçimini değiştirmektedir. Argümanın ikiden fazla seçenekleri kendi aralarında virgülle ayrılır. Zorunlu olmayan *release-date* argümanında *class.cls* dosyasının kullanabilir en eski sürümünün tarihi gösterilmektedir. Bu durumda, tarih "yıl/ay/gün" kuralına göre yazılmalıdır:

```
\documentclass[a4paper,12pt]{article}[2000/05/19]
```

Tanıtımın *class* seçeneğinde aşağıdaki *standart* sınıflardan sadece biri alınmalıdır

`article` - bir bilimsel makale yazmak için kullanılır;
`proc` - bir bildiri yazmak için kullanılır;
`book` - bir kitap yazmak için kullanılır;
`report` - bir önerge veya rapor yazmak için kullanılır;
`letter` - bir mektup yazmak için kullanılır;
`slides` - bir bildiri slaytı hazırlamak için kullanılır.

LaTeX 2_ε tabanını daha da genişletmek için `\documentclass` komutundan sonra aşağıdaki *tanıtım* kullanılmalıdır

```
\usepackage[options]{package}[release-date]
```

Bu durumda, LaTeX 2_ε *package.sty* veya *package.cls* dosyasından gerekli komut değişikliklerini ve yeni komut tanımlamalarını yükleyecektir. Tanıtımın *options* ve *release-date* argümanı yukarıdaki gibidir. Bir giriş dosyasında `\usepackage` komutunun sayısı sınırsızdır, yani bu komut istenilen sayıda kullanılabilir. Ancak bunların hepsi belgenin başlık kısmına yazılmalıdır. Öte yandan, bir tanıtımla birkaç paket de yüklenebilir. Fakat, bu durumda, paketler aynı seçenekli olmak zorundadır:

```
\usepackage[dvips]{graphicx, color}
```

Öte yandan, bu paketin `dvips` seçeneği `\documentclass` tanıtımın *options* argümanında da gösterilebilir.

1.2 Belgenin metni

Bir belgenin başlık kısmından hemen sonra belgenin metni başlamaktadır. Belge metni aşağıdaki komutlu parantez içene alınmalıdır

```
\begin{document}    metin    \end{document}
```

NOT: LaTeX `\end{document}` komutundan sonra yazılan her şeyi iptal etmektedir.

Örneğin, LaTeX 2$_\varepsilon$ 'de çok basit bir çalışma şöyle hazırlanabilir

giriş dosyası:

```
\documentclass{article}
\begin{document}
   Ben \LaTeX{} 'i öğrenmek istiyorum.
\end{document}
```

görüntüsü:

Ben LaTeX 'i öğrenmek istiyorum.

Bundan sonra, tüm örneklerde, solda: *giriş dosyası*, sağda: *görüntüsü*, yani *çıkış dosyası* gösterilecektir.

Aynı örnek LaTeX 2.09 'de şöyle yazılmaktadır:

```
\documentstyle{article}
\begin{document}
   Ben \LaTeX{} 'i öğrenmek istiyorum.
\end{document}
```

1.2.1 Başka dosyadan metin eklemek

Giriş dosyasının herhangi bir kısmı (tablo, şekil, ...) başka bir dosyada hazırlanabilir. Giriş dosyasına

```
\input{file}
```

komutuyla *file* adlı bir dosya eklenebilir. Eğer *file*, `tex` uzantılı bir dosya değilse, dosya adı uzantısıyla birlikte yazılmalıdır:

```
\input{kitap.txt}
```

Bu durumda, LaTeX, *file* dosyasını başından sonuna kadar veya

```
\endinput
```

komutuna kadar okur. `\input` komutu belgenin başlık kısmında da yazılabilir. Üstelik, başlık kısmının kendisi de bir *file* dosyasında olabilir. Belgeye

```
\include{file}
```

komutuyla da *file*`.tex` dosyası eklenebilir.

NOT: `\include` komutu `document` bloğunun içine yazılmalıdır, yani bu komut belgenin başlık kısmına yazılamaz.

LaTeX bir dosyayı belgeye eklemeden önce ve sonra yeni bir sayfaya geçmektedir (otomatik olarak `\clearpage` komutu çalışır). Genelde bu komut belgeye önceden hazırlanmış bir bölüm veya kaynakça eklemek için kullanılabilir.

```
\includeonly{file}
```

komutu da, `\include` komutu gibi, *file* dosyasını belgeye eklemektedir. Bu kullanımda, LaTeX birden fazla dosyayı belgeye ekleyebilir. Bunun için komutun *file* argümanında birden fazla dosya adı bir-birinden virgülle ayrılarak yazılmalıdır. Yazılan dosyalardan biri bulunamazsa, LaTeX yeni bir sayfa başlatır.

Giriş dosyası LaTeX 'de derlendikten sonra belgeye dışarıdan eklenen tüm dosyalar hakkındaki bilgi belgenin `log` uzantılı dosyasına yazdırılır. Bu dosyaya *protokol dosyası* denir. Protokol dosyasının kendisi de

```
\listfiles
```

komutuyla belgeye eklenebilir.

1.3 Standart sınıfların seçenekleri

`\documentclass` (veya `\documentstyle`) komutunun zorunlu olmayan *options* argümanı bir sınıfın (`article`, `book`, ...) öntanımlı bazı parametreleri ve biçimleri değiştirmektedir. İki ve ikiden fazla seçenekler kendi aralarında virgülle ayırılır. Aşağıda standart sınıfların tüm seçenekleri verilmektedir.

`10pt|11pt|12pt` - metinde yazı boyutunu tanımlamaktadır (`slides` sınıfı hariç). Seçenek gösterilmediği taktirde, LaTeX bu seçeneği `10pt` olarak alır;

`a4paper|a5paper|b5paper|letterpaper|legalpaper|executivepaper` - çeşitli boyuttaki kağıtlar için sayfanın boyutunu tanımlamaktadır. Seçenek gösterilmediği taktirde, LaTeX bu seçeneği `letterpaper` olarak alır. `proc` sınıfı `a5paper` ve `b5paper` seçenekleri desteklememektedir;

`landscape` - her sayfayı yatay döndürülmüş bir şekilde oluşturmaktadır.

`oneside|twoside` - kağıdın sırasıyla tek ve çift nolu yüzüne yazdırılması için sayfa biçimini oluşturmaktadır. Seçenek gösterilmediği taktirde, LaTeX bu seçeneği `book` sınıfı için `twoside` olarak, diğer sınıflar için ise, `oneside` olarak alır. `slides` sınıfı bu seçenekleri desteklememektedir;

`draft|final` - metini sırasıyla *taslak* ve *son* şeklinde hazırlar. Seçenek gösterilmediği taktirde, LaTeX bu seçeneği `final` olarak alır;

`titlepage|notitlepage` - metin başlığını ve özetini sırasıyla yeni ve geçerli sayfaya yazdırılmasını sağlamaktadır. Seçenek gösterilmediği taktirde, LaTeX bu seçeneği `article` sınıfı için `notitlepage` olarak, diğer sınıflar (`letter` hariç) için ise, `titlepage` olarak alır. `proc` sınıfı `titlepage` seçeneğini desteklememektedir;

`onecolumn|twocolumn` - metini sırasıyla tek ve çift sütunlu yazdırılmasını sağlamaktadır. `proc` sınıfı `onecolumn` seçeneğini, `slide` ve `letter` sınıfları ise, `twocolumn` seçeneğini desteklememektedir;

`leqno` - eklenen işlev numarası satırın soluna yazdırılır; öntanımlı olarak işlev numarası sağa yazılır.

`fleqn` - eklenen işlevleri satırın soluna hizalar; öntanımlı olarak sağa hizalıdır.

`openbib` - kaynakçadaki her maddeyi yeni bir satırdan başlatmaktadır. `slides` ve `letter` sınıfları bu seçeneği desteklememektedir;

`openright|openany` - yeni bölümü sırasıyla tek ve yeni sayfadan başlatmaktadır. Bu seçeneği sadece `report` ve `book` sınıfları destekler. Seçenek gösterilmediği taktirde, LaTeX bu seçeneği `report` sınfı için `openany` olarak, `book` sınfı için ise, `openright` olarak alır.

1.4 LaTeX 'de Türk dili

1999 yılında LaTeX3 projesi üzerinde, arasında Türk dili de olmak üzere, LaTeX 'de birkaç dil desteği gerçekleştirilmiştir. Metinde Türkçe harflerin desteklenmesi için belgenin başlık kısmında inputenc ve babel paketleri sırasıyla `latin5` ve `turkish` seçenekleriyle tanıtılmalıdır:

```
\usepackage[latin5]{inputenc}
\usepackage[turkish]{babel}
```

Bu durumda, LaTeX metinde yazılan Türkçe harfleri anlar ve İngilizce *Part*, *Chapter*, *Section*, *References*, ... başlıkları sırasıyla Türkçe *Kısım*, *Bölüm*, *Paragraf*, *Kaynaklar*, ... başlıklarıyla değiştirir. Bunu bir örnekle gösterelim:

```
\documentclass{article}
\usepackage[latin5]{inputenc}
\usepackage[turkish]{babel}
\begin{document}
   \LaTeXe{} 'de Türkçe metin.
\end{document}
```

LaTeX 2_ε 'de Türkçe metin.

Yurt dışına gönderilecek bir Türkçe metini böyle hazırlanması önerilmemektedir. Sebebi, inputenc ve babel paketlerin sırasıyla `latin5` ve `turkish` seçenekleri LaTeX 'in birkaç paketinin bazı özelliklerini engellemektedir. Üstelik, yurt dışındaki birçok yayın kuruluşu ve basım evinin `\documentclass` tanıtımı için kendi özel "*class.cls*" paketi vardır (1.1, 1.sayfa). Örneğin, "Kluwer Academic Publisher" basım evinin özel dosyası: *crckapb.cls* (veya *crckapb.sty*); "Transformation Groups Journal" yayın kuruluşunun özel dosyası: *tglat2e.cls* (veya *tglat2e.sty*); .... Dolayısıyla, bu

yayın kuruluşu ve basım evi bir yazardan makaleyi (veya kitabı) özel *class.cls* dosyasına göre yazmasını istemektedir. Oysa belgenin başlık kısmında yazılacak tanıtımları sınırlandırmaktadır.

Böyle durumda, bir Türkçe metin

```
\usepackage[latin5]{inputenc}
\usepackage[turkish]{babel}
```

tanıtımları kullanılmadan yazılmalıdır. Bunun için, metinde gerekli İngilizce başlıklar (*Part*, *Section*, ...) `\def` komutuyla Türkçe başlıklarla değiştirilmelidir (*bkz.* 2.13.1, 100. sayfa); Latince'de olmayan Türkçe **ç**, **ğ**, **ı**, **ö**, **ş** ve **ü** harfleri ise aşağıdaki komutlar yardımıyla yazılmalıdır (*yine bkz.* 2.14.2, 103. sayfa, tablo 2.4):

`\c{c}`	ç	‖	`\"{o}`	ö
`\u {g}`	ğ	‖	`\c{s}`	ş
`\i␣`	ı	‖	`\"{u}`	ü

Örneğin, Türkçe: *çıkış*, *ışıkçı*, *öğün*, *şoför* ve *üç* kelimeleri metinde şöyle yazılmalıdır:

```
\"{O}rne\u {g}in, T\"{u}rk\c{c}e: \c{c}\i k\i \c{s}, \i \c{s}\i k\c{c}\i ,
\"{o}\u {g}\"{u}n, \c{s}of\"{o}r ve \"{u}\c{c} kelimeleri metinde
\c{s}\"{o}yle yaz\i lmal\i d\i r.
```

1.5 Açıklama

LaTeX bir satırda [%] simgesinden sonraki her şeyi iptal etmektedir. Bu simge LaTeX 'de bir açıklama yapmak için kullanılabilir. Eğer bir açıklama birkaç satırdan oluşuyorsa, bu taktirde her satırın başına % simgesinin yazılması gerekmektedir. Böyle durumda, belgenin başlık kısmında `tools` sınıfının verbatim paketi tanıtılmalıdır:

```
\usepackage{verbatim}
```

Bu paketin komutlu

`\begin{comment}` ... `\end{comment}`

parantezi içindeki her şeyi LaTeX iptal etmektedir. Bunu bir örnekle gösterelim

```
\documentclass{article}
\usepackage[latin5]{inputenc}
\usepackage[turkish]{babel}
\usepackage{verbatim}
\begin{document}
Bir satırda % simgesinden sonra yazılan
her şey ...
\begin{comment}
iptal edilir. ... birkaç satırdan oluşan bir
açıklama
\end{comment}
için \% simgesi çok sayıda ...
\end{document}
```

Bir satırda her şey ... için
% simgesi çok sayıda ...

Bir açıklama ayrı bir dosya şeklinde de hazırlanabilir. Her açıklama için belgenin başlık kısmında, `\documentclass` komutundan evvel, aşağıdaki komutların herhangi biri kullanılabilir

`\begin{filecontents}{name.ext}` *contents* `\end{filecontents}`
`\begin{filecontents*}{name.ext}` *contents* `\end{filecontents*}`

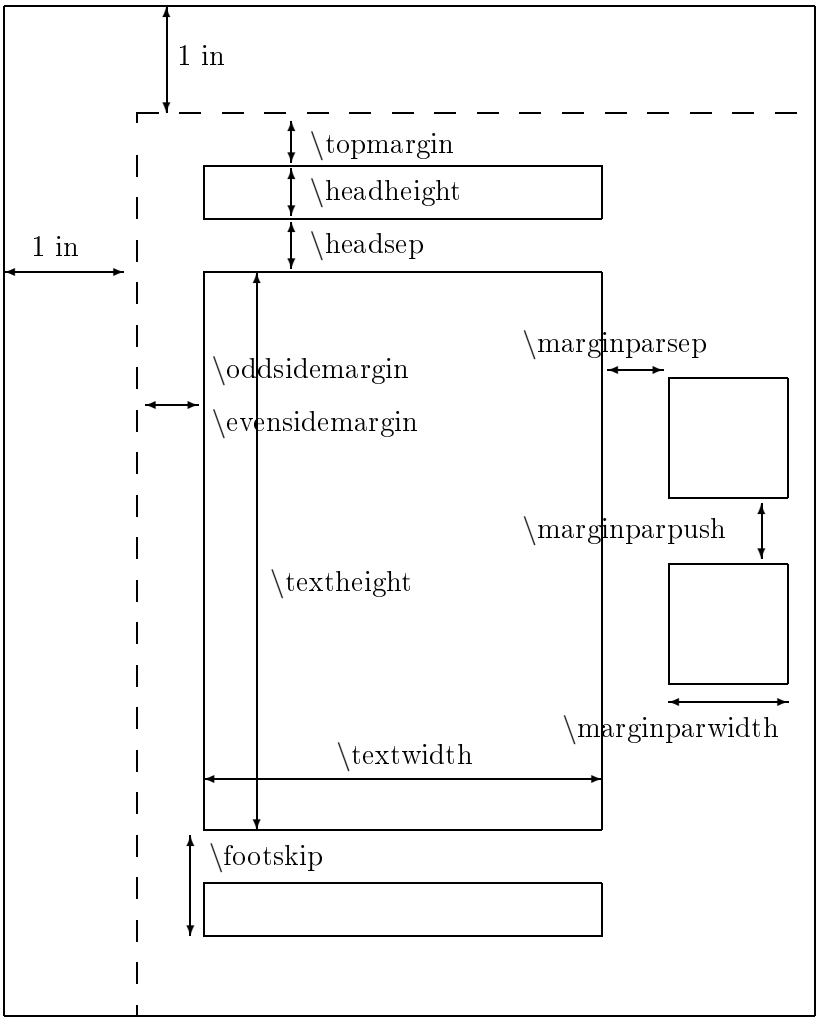
LaTeX her `filecontexts` bloğu için *name.ext* adlı bir dosya üretip, bu dosyaya dosya hakkında kısaca bir açıklama ve *contents* açıklamasını yazdırır. *-yıldızlı komut *name.ext* dosyasına sadece *contents* açıklamasını yazdırır.

1.6 Dosyayı yazdırmak

1.6.1 Sayfa seçenekleri

Bir sayfa üst ve alt bilgi, ve onlar arasına yazılan bir metinden oluşmaktadır. Bundan başka, sayfanın sol ve sağ boşluğuna `\marginpar` komutuyla bir not da yazdırılabilir (*bkz.* 2.11.3. paragraf, 90. sayfa).

TeX bir sayfanın üst ve alt bilgi, sol ve sağ boşluğu gibi tüm parametrelerini otomatik ayarlar. Şekil 1.1 'de bu parametrelerin boyutunu tanımlayan komutlar ve parametrelerin yerleri gösterilmektedir.



Şekil 1.1: Sayfa parametre boyutunu tanımlayan komutlar.

Şekilden görüldüğü gibi, `\oddsidemargin` ve `\evensidemargin` komutları sırasıyla tek ve çift sayfaların sol kenar boşluklarını tanımlamaktadır. Bir parametrenin boyutu belgenin başlık kısmında `\setlength` ve `\addtolength` komutlarıyla değiştirilebilir (*bkz.* 2.1.paragraf, 44.sayfa). Bir parametre değiştirilmeden önce onun geçerli boyutunu öğrenmekte yarar vardır. Bunun için metin başında `tools` sınıfın `layout` paketi tanıtıl-

malıdır:

```
\usepackage{layout}
```

Paketin `\layout` komutu geçerli sayfa parametrelerini ve onun boyutunu bir maket şeklinde oluşturur (bu maket kitabın son sayfasında verilmektedir). Maketten görüldüğü gibi, sol ve sağ sayfanın bazı parametreleri farklıdır. Bundan dolayı, `\layout` komutu sol ve sağ sayfa için ayrı ayrı maket oluşturmaktadır. Maketlerde kullanılan TEX 'in boyut birimleri tablo 2.1 'de verilmektedir (*bkz.* 43.sayfa).

1.6.2 İlk sayfa

Bir ilk sayfa, genelde, metin adı, yazar adı ve özetten oluşmaktadır. Bazen bir ilk sayfada dosyanın yazıldığı tarih de gösterilebilir. Bir metinde ilk sayfa

```
\maketitle
```

komutuyla oluşturulmaktadır. `\maketitle` komutuna aşağıdaki iki komut daha eşlik etmelidir:

```
\title{title}
\author{author(s)}
```

`\title` komutunun *title* argümanında metin başlığının, `\author` komutunun *author(s)* argümanında ise, yazarın (veya yazarların) ismi yazılmalıdır. Komutların argümanı `\title{ }` ve/veya `\author{ }` olarak boş bırakılabilir. `\title` komutu içerisinde uzun adlı bir metin başlığı `\\` komutuyla birkaç satıra ayrılabilir. Çalışmanın yazıldığı tarih

```
\date{date}
```

komutunun *date* argümanında gösterilebilir. Eğer bu komut metinde kullanılmazsa, `\maketitle` komutu metin LATEX 'de okutulduğu andaki tarihi çıkış dosyasına yazdırır. Tarih yazdırılmaması için ise, `\date` komutun *date* argümanı `\date{ }` olarak boş bırakılmalıdır.

`\title` , `\author` ve `\date` komutları belgenin başlık kısmına yazılmalıdır. Bundan başka, bu komutların argümanı

```
\thanks{text}
```

komutunu içerebilir. Komutun *text* argümanında bir (E-mail) adresi, iş yeri ve/veya gerekli bir yere teşekkür yazılabilir.

`\maketitle` komutu `\begin{document}` komutundan sonra yazılmalıdır, yani bu komut belgenin başlık kısmına yazılamaz. `\maketitle` komutu `\documentclass` (veya `\documentstyle`) tanıtımının

> `titlepage` seçeneğinde ilk sayfayı yeni bir sayfayada yazdırır. Bu durumda, baş sayfadan sonradaki ilk sayfa 1.sayfa olarak kabul edilmektedir.
>
> `notitlepage` seçeneğinde baş sayfa 1.sayfaya, normal metinin üst kısmına, yazdırılır.

Seçenek gösterilmediği taktirde ise, LaTeX bu seçeneği `article` sınıfı için `notitlepage` olarak, `book`, `report` ve `slides` sınıfları için `titlepage` olarak alır.

`\author` komutunun *author(s)* argümanında birkaç yazar ismi

```
\and
```

komutu yardımıyla yazılabilir. Bu durumda, LaTeX her yazar ismini ayrı-ayrı olarak yazdırır ve isimler arasına gerekli bir aralık da bırakır.

Bir ilk sayfa

```
\begin{titlepage}  ...  \end{titlepage}
```

bloğuyla da oluşturulabilmektedir. Bu durumda, ilk sayfa yeni bir sayfaya yazdırılır ve bu sayfadan sonradaki ilk sayfa 1.sayfa olarak kabul edilir.

Belgede bir özet

```
\begin{abstract}  ...  \end{abstract}
```

bloğu yardımıyla oluşturulabilir. `abstract` bloğu sadece `article` ve `report` sınıflarında geçerlidir.

Çalışma özeti ayrı bir sayfaya yazdırılması için çalışmanın `titlepage` seçeneği alınmalıdır. Özet İngilizce **Abstract** kelimesiyle başlanır. Metin başında `babel` paketinin *turkish* seçeneği tanıtılmışsa, LaTeX bu İngilizce

kelime yerine Türkçe **Özet** kelimesini yazdırır. Üstelik, bu kelime

```
\abstractname
```

komutunda saklanmaktadır. Böylece, özetin adı `\renewcommand` komutuyla değiştirilebilir (*bkz.* 2.13.1. paragraf, 99.safya).

Çalışmanın `twocolumn` seçeneğinde hem özet hem de metin iki sütunlu olarak yazdırılır. Özet bir sütunlu, metin ise iki sütunlu olması için belgenin başlık kısmında `abstract` paketi tanıtılmalıdır. Bu paket bir özeti her zaman tek sütunlu olarak yazdırır.

Aşağıda yukarıda tanımlanan tüm komutları içeren bir örnek ele alınmaktadır

```
\documentclass{article}
⋮
\title{Türkçe\LaTeX{}\thanks{
Çalışma GKNT'nin N3102 bursuyla
desteklenmiştir}
\author{A.A.Rahimov \thanks{
rakhimov@ktu.edu.tr}
\and O.Kesemen}
\date{}
\maketitle
\begin{abstract}
Bu çalışmada ...
⋮
\end{abstract}
Buradan normal metin başlan-
maktadır
⋮
```

Türkçe LaTeX*

A.A.Rahimov ** O.Kesemen

Özet

Bu çalışmada ...

⋮

Buradan normal metin başlanmaktadır

⋮

* Çalışma GKNT'nin N3102 bursuyla desteklenmiştir

** rakhimov@ktu.edu.tr

1.7 Kısım, bölüm ve paragraf

LaTeX 'de bir metini (`\letter` sınıfı hariç) kısım, bölüm ve paragraflara ayıran özel komutlar vardır. Her komut, birisi zorunlu olmak şartıyla, iki argümandan oluşmaktadır. Zorunlu argümana *bölümün* adı, zorunlu olmayan argümana ise bölümün *İçindekiler* (*bkz.* 17.sayfa) ve sayfa üst bilgisindeki adı yazılmaktadır. Zorunlu olmayan argüman gösterilmediği taktirde, LaTeX bu argümana zorunlu argümandaki değeri alır. Genelde, zorunlu olmayan argümana *bölüm*ün kısa adı yazılır. Bir argüman dayanık-

sız komutları içeriyorsa (böyle argümana *hareketli argüman* denir), bu komutlar, güvenlik amacıyla, `\protect` komutuyla korunması gerekmektedir. Örneğin, `$`, `\hspace`, `\hfill`, ... gibi komutlar bir argüman içinde dayanıksızdır. Dolayısıyla, bu komutlar bir argümanda `\protect` komutuyla şöyle muhafaza edilmelidir:

`{\protect $}`, `\protect\hspace{..}`, `\protect\hfill`, ...

LaTeX her *Kısım*, *Bölüm*, *Paragrafı* otomatik numaralandırır. Bu numaralar

`\secnumdepth`

değişkeninde saklanmaktadır. Onun değeri `\setcounter{}{}` (*bkz.* 2.6. paragraf, 61.sayfa) komutuyla değiştirilebilir. Örneğin,

```
\setcounter{chapter}{5}
```

tanıtımı sıradaki bölüme 5 numarasını vermektedir. Benzer şeklinde, kısım ve paragrafın da numarası değiştirilebilir. Aslında, `\setcounter` komutuyla başka bir şeyin de numarası değiştirilebilir. Örneğin,

```
\setcounter{page}{23}
```

tanıtımı geçerli sayfa numarasını 23 den başlatır (*bkz.* 2.6).

Bir alt düzey değişkeni sadece üst düzey değişkene bağımlıdır. Dolayısıyla, bir bölüm alt düzeyi başka bir bölüm alt düzeyinden bağımsız şeklinde numaralandırılır.

LaTeX 'in *kısım*, *bölüm* ve *paragrafı* oluşturan komutun *-yıldızlı şekli de vardır. Her yıldızlı komutun sadece zorunlu bir argümanı vardır. Bu argümana, yıldızsız komutta olduğu gibi, bölümün adı yazılmalıdır.

1.7.1 Kısım

LaTeX 'de bir metin aşağıdaki komut yardımıyla birkaç kısıma ayrılabilir

`\part[toc]{head}`	`\part*{head}`

`\part` komutu

* `book` ve `report` sınıfında yeni sayfa ortasına İngilizce **Part** kelimesi, kısım numarası ve yeni satıra kısımın *head* adını yazdırır;

* `article` sınıfında **Part** kelimesini *İçindekilerin* üst kısmına yazdırır.

Komutun *toc* argümanı zorunlu değildir. Bu argümana kısımın *İçindekiler* ve sayfa üst bilgisindeki adı yazılmalıdır. *toc* argümanı gösterilmediği taktirde, LaTeX bu argümana zorunlu argümanı alır. *-Yıldızlı `\part*` komutu sadece kısmın adını yazdırır.

Üstelik, `\part` ve `\part*` komutun kendisi de zorunlu değildir. Dolayısıyla, bu komut bir alt düzey numarasıda etkisizdir.

Part kelimesi

```
\partname
```

komutunda saklanmaktadır. Bu kelime `\renewcommand` komutuyla değiştirilebilir. `babel` paketinin *turkish* seçeneği **Part** yerine Türkçe **Kısım** kelimesini yazdırır.

1.7.2 Bölüm

LaTeX 'de bir metin aşağıdaki komut yardımıyla birkaç bölüme ayrılabilir

```
\chapter[toc]{head}        \chapter*{head}
```

`\chapter` bloğu sadece `book` ve `report` sınıfında geçerlidir. Her bölüm yeni bir sayfadan başlamaktadır, şöyle ki, çalışmanın `openright` seçeneğinde yeni sağ sayfadan, `openany` seçeneğinde ise, sıradaki yeni sayfadan başlamaktadır.

Seçenek yazılmadığı taktirde, LaTeX bu seçeneği `book` sınıfı için `openright` olarak, `report` sınıfı için ise `openany` olarak alır.

`\chapter[toc]{head}` komutu İngilizce **Chapter** kelimesini, numarasını, ve bölümün *head* adını yazdırır. Bölümün metni ise, yeni satırdan başlamaktadır. Komutun *toc* argümanı zorunlu değildir. Bu argümana bölümün *İçindekiler* ve sayfa üst bilgisindeki adı yazılmalıdır. *toc* argümanı yazılmadığı taktirde, LaTeX bu argümana zorunlu argümanı alır. *-Yıldızlı `\chapter*` komutu sadece bölümün adını yazdırır.

Chapter kelimesi

```
\chaptername
```

komutunda saklanmaktadır. Bu kelime `\renewcommand` komutuyla değiş-

tirilebilir. babel paketinin *turkish* seçeneği **Chapter** yerine Türkçe **Bölüm** kelimesini yazdırır.

1.7.3 Paragraf

LaTeX 'de bir metin aşağıdaki komutlarla birkaç paragraf ve alt paragrafa ayrılabilir

`\section[`*toc*`]{`*head*`}`	`\section*{`*head*`}`
`\subsection[`*toc*`]{`*head*`}`	`\subsection*{`*head*`}`
`\subsubsection[`*toc*`]{`*head*`}`	`\subsubsection*{`*head*`}`

Komutların *head* ve *toc* argümanı `\chapter` komutunun argümanları gibidir. *toc* argümanı yazılmadığı taktirde, LaTeX bu argümana zorunlu *head* argümanı alır. Komutların *-yıldızlı şekli sadece paragraf ve alt paragraf adını yazdırır.

Satırbaşı

Bir paragraf veya alt paragraf birkaç satırbaşını içerebilir. Bir satırbaşı aşağıdaki komutların herhangi biriyle oluşturulur

`\paragraph[`*toc*`]{`*head*`}`	`\paragraph*{`*head*`}`
`\subparagraph[`*toc*`]{`*head*`}`	`\subparagraph*{`*head*`}`

Bu komutların da *head* ve *toc* argümanı yukarıda ele alınan komutların argümanı gibidir. Bu durumda, satırbaşının *head* adı koyu yazısıyla, devamından ise, onun içereği yazdırılır. *toc* argümanı gösterilmediği taktirde, LaTeX bu argümana zorunlu *head* argümanı alır. Komutların *-yıldızlı şekli sadece satırbaşı adını yazdırır.

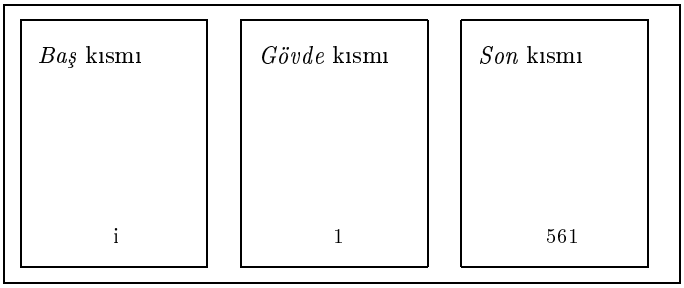
1.7.4 Kitabı kısımlara bölmek

Bir kitap, genelde, *Baş*, *Gövde* ve *Son* gibi kısımlardan oluşur. LaTeX dosyanın `book` seçeneğinde bu kısımları otomatik olarak ayarlayabilir. LaTeX 'in

```
\frontmatter        \mainmatter        \backmatter
```

komutları bir çalışmayı sırasıyla *Baş*, *Gövde* ve *Son* kısımlara bölmektedir. Çalışmanın *Baş* ve *Son* kısmında `\chapter` komutu bölümü numarasiz yazdırmaktadır, ama bölüm adını İçindekilere yazdırır. Bu komutları bir örnekle gösterelim

```
\begin{document}
\frontmatter
Baş kısmı
\mainmatter
Gövde kısmı
\backmatter
Son kısmı
\end{document}
```



Görüldüğü gibi, bir sayfa `\frontmatter` bloğunda, yani çalışmanın *Baş* kısmında, Roma rakamlarıyla, `\mainmatter` ve `\backmatter` bloğunda ise, Arapça rakamlarla numaralandırılmaktadır ve her kısım yeni sayfadan başlatılmaktadır.

Uygulama

Bir çalışmanın "Ek" kısmı LaTeX 'in aşağıdaki komutuyla özel olarak oluşturulabilir

```
\appendix
```

`\appendix` bloğunda `\chapter` komutu **Bölüm** kelimesi yerine **Appendix** kelimesi, bölüm numarası yerine ise, Latince **A, B, C,** ... harfleri yazdırmaktadır. `article` sınıfında `\chapter` komutu desteklenmediği için onun yerine `\section` komutu **Appendix** kelimesini yazdırmaktadır. **Appendix** kelimesi

```
\appendixname
```

komutunda saklanmaktadır. Bu kelime `\renewcommand` komutuyla değiştirilebilir. `babel` paketin *turkish* seçeneği **Appendix** yerine Türkçe **Ek**

kelimesini yazdırmaktadır. `\appendix` bloğunu bir örnekle gösterelim

```
⋮
\appendix
\chapter{Türevin uygulaması}
\section{
Türevin fiziksel uygulaması}
Düzgün hızlanan bir hareketli-
nin t anında aldığı yol
...
\section{
Türevin iktisatta uygulaması}
Bir salça fabrikasında üretilen
salçanın maliyet ...
\chapter{İntegralın uygulaması}
\section{
Hacim hesabına uygulaması}
a yarıçaplı bir kürenin hacmini
hesaplayınız ...
⋮
```

⋮

Ek A

Türevin uygulaması

A.1 Türevin fiziksel uygulaması

Düzgün hızlana bir hareketlinin t anında aldığı yol ...

A.2 Türevin iktisatta uygulaması

Bir salça fabrikasında üretilen salçanın maliyet ...

Ek B

İntegralın uygulaması

B.1 Hacim hesabına uygulaması

a yarıçaplı bir kürenin hacmini hesaplayınız.

⋮

1.8 İçindekiler, Şekil ve Tablo Listesi

1.8.1 İçindekiler

Önceki paragraflarda görüldüğü gibi, bir metin `\part`, `\chapter`, `\section` ve `\subsection` komutuyla sırasıyla *Kısım*, *Bölüm*, *Paragraf* ve *Alt paragrafa* bölünmektedir. Bu durumda, LaTeX bir `*.toc` dosyası üretip, bu dosyaya kısım, bölüm, paragraf ve alt paragrafın adı ve sayfa numarasını yazdırır. `*.toc` dosyası aşağıdaki komut için bir *yardımcı* dosyadır

```
\tableofcontents
```

Bu komut yardımcı `*.toc` dosyasından bir bilgi alıp giriş dosyasında *İçindekileri* oluşturmaktadır. İçindekilere, genelde, kısım, bölüm ve paragrafa ait bilgi yazdırılmaktadır, yani `\tableofcontents` komutu `*.toc`

dosyasından sadece 2. düzeye (`\subsection` düzeyine) kadar bilgi almaktadır. Bu düzey numara

`tocdepth`

değişkeninde saklanmaktadır ve bu `\setcounter` komutuyla değiştirilebilir. Yukarıda söylendiği gibi, `tocdepth` 'e yeni bir değer verilmezse LaTeX bu değeri `2` (`\setcounter{tocdepth}{2}`) olarak alır.

1.8.2 Şekil ve tablo listesi

Metinde bir şekil ve tablo listesi sırasıyla

`\listoffigures` `\listoftables`

komutlarıyla yazdırılmaktadır. Komutların her biri kendisi için LaTeX tarafından üretilen yardımcı dosyadan bilgileri alıp metinde bir şekil ve tablo listesini oluşturmaktadır. Bu iki yardımcı dosya ise sırasıyla `*.lof` ve `*.lot` türünden olup, bunları `\caption` komutu üretmektedir (*bkz.* 83. sayfa). Bir şekil ve tabloda kullanılan bir `\caption` komutu şekil ve tablonun adı ve sayfa numarasını sırasıyla `*.lof` ve `*.lot` dosyasına yazdırmaktadır. `\listoffigures` ve `\listoftables` komutları yardımcı dosyadan gerekli bilgileri alıp, metinde şekil ve tablo listesini yazdırmaktadır.

Bundan başka, `*.toc`, `*.lof` ve `*.lot` dosyasındaki bilgiler aşağıdaki iki komut yardımıyla da hazırlanabilmektedir

`\addcontentsline{`*ext*`}{`*unit*`}{`*text*`}`
`\addtocontents{`*ext*`}{`*text*`}`

Komutun *text* argümanı *file.ext* (=*file*`.toc`, *file*`.lof`, *file*`.lot`) dosyasına eklemektedir [1]; *unit* argümanı *text* argümanın tipini göstermektedir, örneğin `*.toc` tipli dosya için *unit*'e `\part`, `\chapter`, `\section`, ... gibi komutlar yazılmalıdır; `*.lof` ve `*.lot` tipli dosyalar için ise *unit*'e sırasıyla `figure` ve `table` blokları yazılmalıdır.

İçindekiler, şekil ve tablo listesinden önce onların adı sırasıyla İngilizce **Contents**, **List of Figures** ve **List of Tables** kelimeleriyle yazdırılmak-

[1] Aslında, *file.ext* dosyası keyfi olabilir, yani *text* argümanı herhangi bir dosyaya eklenebilir.

tadır. Bu kelimeler sırasıyla

\contentsname	\listfigurename	\listtablename

komutlarında saklanmaktadır ve onlar `\renewcommand` komutuyla değiştirilebilir. `babel` paketinin *turkish* seçeneği bu kelimeleri sırasıyla Türkçe **İçindekiler**, **Şekil Listesi** ve **Tablo Listesi** olarak yazdırır.

1.9 Üst ve alt bilgi

Şekil 1.1 'de bir sayfa üst ve alt bilgi boyutları gösterilmektedir. (*bkz.* 9.sayfa). Bu boyutlar sayfa seçeneğini tanıtan

```
\pagestyle{style}
\thispagestyle{style}
```

komutlarıyla ve *style* argümanından oluşmaktadır. `\thispagestyle` komutu sadece geçerli sayfa için etkilidir.

style argümanının aşağıdaki seçenekleri vardır

plain sayfa numarası alt bilgi ortasına yazdırılır; üst bilgi ise boş kalır. Seçenek yazılmadığı taktirde LaTeX bu seçeneği standart sınıflarda (`book` ve `letter` sınıfları hariç) *plain* olarak alır.

empty sayfa üst ve alt bilgisi boş bırakılır. Seçenek yazılmadığı taktirde LaTeX bu seçeneği `letter` sınıfında *empty* olarak alır.

headings sayfanın üst bilgisine bölüm adı ve sayfa numarası yazdırılır; alt bilgi boş kalır. Seçenek yazılmadığı taktirde LaTeX bu seçeneği `book` sınıfında *headings* olarak alır.

Tek yüzülü yazdırmada bir sayfanın üst bilgisine:

`book` ve `report` sınıflarında bölüm adı (`\chapter` komutundan alınır) ve sayfa numarası;

`article` ve `proc` sınıflarında paragraf adı (`\section` komutundan alınır) ve sayfa numarası yazdırılır.

Böyle biçimler aşağıda tanımlanan `\markright` komutunda da vardır.

İki taraflı yazdırmada:

sol sayfalar üst ve alt bilgisi tek yüzeyli yazdırmadaki gibidir;

sağ sayfalar üst bilgisine:

`book` ve `report` sınıflarında paragraf adı (`\section` komutundan alınır) ve sayfa numarası;

`article` ve `proc` sınıflarında alt paragraf adı (`\subsection` komutundan alınır) ve sayfa numarası yazdırılır.

Alt bilgi boş bırakılır.

Böyle biçimler aşağıda tanımlanan `\markboth` komutunun birinci argümanında da vardır.

myheadings üst ve alt bilgi özel olarak tanımlanmaktadır. Bu seçeneğin `headings` seçeneğinden farkı sayfa üst bilgisinde aşağıda tanımlanan `\markboth` ve/veya `\markright` komutunun argümanı yazdırılır. Alt bilgi boş bırakılır.

`\markboth{`*left*`}{`*right*`}`
`\markright{`*right*`}`

komutlarının

- *myheadings* seçeneğinde bir sayfa üst bilgisi (sayfa numarası hariç) özel olarak tanımlanabilir;

- *headings* seçeneğinde sayfa üst bilgisine bölüm (veya paragraf) adı yerine komutun argümanı yazdırılır.

`\markboth` komutu *left* argümanı sol sayfa üst bilgisine; her iki komut *right* argümanı sağ sayfa üst bilgisine yazdırır. *left* ve *right* argümanında dayanıksız komutlar `\protect` komutuyla korunması gerekmektedir. Örneğin `$` komutu hareketli argümanda dayanıksızdır; dolayısıyla

```
\markright{$ax+b=c$ 'in çözümü}
```

komutu

```
\markright{{\protect $} ax+b=c {\protect $} 'in çözümü}
```

olarak yazılmalıdır.

1.9.1 Standart olmayan üst ve alt bilgi

Bir sayfanın standart olmayan üst ve alt bilgisi `\@oddhead`, `\@evenhead`, `\@oddfoot` ve `\@evenfoot` komutlarını yeniden tanımlayarak elde edilir. Bunu bir örnekle gösterelim:

```
\makeatletter
\renewcommand{\@oddhead}{\hfill\thepage\hfill}
\renewcommand{\@oddfoot}{}
\makeatother
```

tanıtımları tek numaralı sayfa numarasını alt bilgi yerine üst bilgi ortasına yazdırır. Bu tanıtımlar metin başına yazılmalıdır.

Bundan başka, `fancyhdr` paketi de standart olmayan üst ve alt bilgi oluşturur. Bu konuda `fancyhdr` paketin `*.doc` dosyasından gerekli bilgi alınabilir.

1.10 Çapraz gönderme

Bazen metinin bir yerinde metinin başka bir yerindeki ifadeden (bu ifade her şey olabilir) yararlanabilir. Böyle durumda, önce bu ifade işaretlenmelidir, sonra gerekli yerde o işarete bir gönderme yapılabilir.

Metinde bir işaret ve gönderme sırasıyla `\label{`*name*`}` ve `\ref{`*name*`}` komutlarıyla oluşturulmaktadır:

`\label{`*name*`}`

komutu *name* adlı bir işaret oluşturmaktadır. Bu durumda, LaTeX ana dizinde `*.aux` tipli bir dosya üretir. `*.aux` dosyasına `\label` komutu yazılan yerdeki nesne (paragraf, denklem, şekil, tablo, ...) değişkenin numarası ve sayfa numarası yazdırılır. Örneğin, eğer `\label` komutu

`equation` (*bkz* 3.2.1.paragraf, 115.sayfa) bloğunda ise, matematiksel bir denklem veya formül numarası;

`tablo` (*bkz* 2.11.1.paragraf, 84.sayfa) bloğunda `\caption` komutundan (*bkz* 2.10.4.paragraf, 83.sayfa) sonra yazılmışsa, tablo numarası

`*.aux` dosyasına *name* adlı bir işaret olarak yazdırılır. Bir nesne kendi değişkenini `\refstepcounter` komutuyla ilan etmektedir.

`\ref{`*name*`}`

komutu *name* adlı işarete bir gönderme yapmaktadır. Bu komut *name* adlı işaret yerleştiği nesne değişken numarasını yazdırır.

`\pageref{`*name*`}`

komutu da *name* adlı işarete bir gönderme yapmaktadır. Bu komut *name* adlı işaretin sayfa numarasını yazdırmaktadır. `\pageref` komutu işaret ve ona gönderme aynı sayfada veya komşu sayfada olduğu taktirde de sayfa numarasını yazdırır.

NOT: LaTeX 'i ilk öğrenecek öğrenci kitabi ilk okuyuşunda aşağıdaki 1.10.1, varioref paketi, 1.10.2, xr paketi ve 1.10.3, hyperref paketi adlı alt paragrafları okumayabilir.

1.10.1 varioref paketi

`tools` sınıfının varioref paketinde (metin başına `\usepackage{varioref}` tanıtımı eklenmeli) aşağıdaki daha "akıllı" bir komut vardır

`\vpageref[`*text1*`][`*text2*`]{`*name*`}`

`\vpageref` komutunun işareti ve ona gönderme aynı sayfada olduğu taktirde *text1* argümanı, diğer durumda ise *text2* argümanı yazdırır. Bu argümanlar yazılmadığı taktirde `\vpageref` komutu

- işaret ve ona gönderme aynı sayfada ise, "on this page" kelimesini;
- gönderme işarete göre aynı yüzeydeki komşu sayfada ise, "on the facing page" kelimesini;
- işaret ve ona gönderme aynı yüzeyde olmayıp, gönderme işarete göre sonraki sayfada ise, "on the next page" kelimesini;
- gönderme işarete göre en az bir sayfadan sonra (veya önce) gelse, "on page" kelimesi ve sayfa numarasını yazdırır.

varioref paketi tüm Avrupa dillerini desteklediği gibi Türk dilini de desteklemektedir. Dolayısıyla, bu paketin *turkish* seçeneği:

```
\usepackage[turkish]{varioref}
```

İngilizce kelimeleri uygun olan Türkçe kelimelerle değiştirmektedir. Eğer varioref paketinin *turkish* seçeneği çalışmazsa veya İngilizce kelimeleri Türkçe kelimelerle değiştirmezse (LaTeX'in bazı eski sürümlerinde öyle olabilir), bu taktirde belgenin başlık kısmına aşağıdaki tanıtımlar grubu yazılmalıdır:

```
\usepackage{varioref}
\def\reftextfaceafter {\reftextvario{ayn\i y\"{u}zeydeki}{sonraki}
                       sayfa}
\def\reftextfacebefore{\reftextvario{ayn\i y\"{u}zeydeki}{bir
                       \"{o}nceki} sayfa}
\def\reftextafter     {\reftextvario{sonraki}{sonraki} sayfa}
\def\reftextbefore    {\reftextvario{bir \"{o}nceki sayfa}{bir
                       \"{o}nceki sayfa}}
\def\reftextcurrent   {\reftextvario{bu}{ge\c cerli} sayfa}
\def\reftextfaraway#1{\pageref{#1} . sayfa}
\def\reftextpagerange#1#2{\pageref{#1}--\pageref{#2} sayfalar\i n}
\def\reftextlabelrange#1#2{\ref{#1} to~\ref{#2}}
```

1.10.2 xr paketi

tools sınıfın xr paketi (belgenin başlık kısmına `\usepackage{xr}` tanıtımı yazılmalı) başka bir dosyadaki bir işaretten yararlanmasına izin vermektedir. Bunun için, bu dosya belgenin başlık kısmına

`\externaldocument[`*prefix*`]{`*filename*`}`

komutuyla tanıtılmalıdır. Komutun zorunlu olmayan *prefix* argümanı *filename.tex* dosyasından alınacak işaretin giriş dosyasının kendi işaretleriyle karışmaması için gerekmektedir. *filename.tex* dosyada `\label{`*name*`}` ile işaretlenen bir nesneye metinde `\ref{`*prefixname*`}` komutuyla bir gönderme yapılabilir.

NOT: Çıkış dosyasında tüm göndermeler doğru gözükmesi için, LaTeX 'de önce belgenin başlık kısmına `\externaldocument` komutuyla tanıtılan *filename.tex* adlı tüm dosyalar ve en sonda giriş dosyası okutulmalıdır.

1.10.3 hyperref paketi

hyperref, Sebastian Rahtz tarafından yazılmış ve *Acrobat Reader*'in bir pdf-dosyasına dönüştürülmesinde aşırı-gönderme yapılmasını destekleyen ilk pakettir. Bu paketin çok sayıda seçeneği vardır. Seçenekler, hyperref paketinin `\usepackage` komutuyla tanıtımında, komutun zorunlu olmayan argümanları kendi aralarında virgülle ayrılmaktadır:

`\usepackage[`*seçenekler listesi*`]{hyperref}`

veya paket tanıtımından sonra

`\hypersetup{`*seçenekler listesi*`}`

komutun argümanında da verilebilir. Bir seçeneğin sadece iki: `true` ve `false` değeri vardır. Bu değer

seçenek=değer

olarak tanıtılmaktadır. Paketin tanıtımında `true` değerli seçenekler yazılmayabilir. Seçeneklerin tam listesi [8], [9] kaynaklarından bulunabilir [2].

hyperref paketi LaTeX 'in birçok komutlarını yeniden tanımlamaktadır. Dolayısıyla, bu paket belgenin başlık kısmında en son olarak tanıtılması gerekmektedir. Paketin tanıtılmasında onun *sürücüsü* de gösterilmelidir. Çünkü, her sürücü paketin kendi komutlarını farklı tanıtmaktadır. Paketin `pdflatex`, `dvips`, `dvipsone`, `dviwindo`, `vtex`, `hypertex`, `ps2pdf`, `pdftex` ve `latex2html` gibi sürücüleri vardır. Sürücüler hakkında [9] kaynağından bilgi alınabilir.

hyperref paketi metindeki her göndermeyi *aşırı-metinli bir göndermeye* dönüştürmektedir. Bu durumda, İçindekiler, şekil ve tablo listesi, altyazı-göndermesi ve kaynakça-göndermesi aşırı-metinli gönderme şekline dönüşmektedir. Bundan başka, *Dizin*deki bir sayfa numarası da aşırı-

[2] Aslında, kaynaklardaki dosya LaTeX 'de de vardır

metinli gönderme şekline dönüşmektedir. Bu özelliklerin herhangi biri, gerekirse, iptal edilebilir. Örneğin, dizindeki bir sayfa numarasının aşırı-metinli gönderme şeklinde oluşması hyperref paketin seçeneğinde

hyperindex=false

tanıtımıyla iptal edilebilir. Çünkü, bu seçenek böyle gösterilmediği taktirde paket `hyperindex` seçeneğinin değerini otomatik `true` olarak alır.

Aşırı-gönderme metni

Bir aşırı-gönderme metni sadece nesne veya sayfa numarasından oluşmaktadır. Bu ise, bir aşırı-gönderme için çok kısadır. hyperref paketi bir göndermeyi yeterli derecede genişletmektedir. Paketin

`\hyperref[`*name*`]{`*text*`}`

komutu *name* adlı bir etiketi *text* argümanın aşırı-gönderme metni olarak tanıtmaktadır. Bunu bir örnekle gösterelim:

```
\hyperref[sec1]{%
\ref*{sec1}bölümünde} \dots
```

1.10.3 bölümünde ...

hyperref paketin bir başka komutu daha vardır:

`\autoref{`*name*`}`

`\autoref` komutu *name* etiketinin aşırı-gönderme metni yerine doğrudan nesne adını ve değişken numarasını yazdırmaktadır:

```
In \autoref{sec1} \dots
```

In subsection 1.10.3 ...

Görüldüğü gibi, bu komut nesne adını İngilizce yazdırmaktadır. Maalesef, şimdilik `\autoref` komutu, birçok yabancı dili, özelikle Türk dilini desteklememektedir. Türk dilini desteklemesi için nesne adının saklandığı komutlar [3] `\renewcommand` komutuyla yeniden tanımlanması gerekmektedir.

Örneğin, `\sectionautorefname` komutunda saklanan *section* kelimesi

```
\renewcommand{\sectionautorefname}{b\"{o}l\"{u}m}
```

[3] komutların tam listesi hyperref paketin `dtx`-dosyasından bulunabilir.

gibi Türkçe *bölüm* kelimesiyle değiştirildikten sonra `\autoref{ }` komutu şöyle çalışır:

```
Bkz. \autoref{sec1}
```

Bkz. bölüm 1.10.3

Aşırı-gönderme metninde bölüm adı

LaTeX metinde `\ref{`*name*`}` komutuyla yapılan bir gönderme yerine `\label{`*name*`}` etiketi, yazılan bölümün (veya paragrafın) sadece numarasını yazdırmaktadır.

hyperref paketiyle ortak çalışan nameref paketi ise gönderme özelliğini aşağıdaki gibi genişletmektedir. Paketin

`\nameref{`*name*`}`

komutu *name* adlı etiketi bir aşırı-gönderme metni olarak oluşturup, işaretin yerleştirildiği bölüm (veya paragraf) adını yazdırmaktadır. Bunu bir örnekle gösterelim:

```
" \nameref{sec1} " bölümünde
```

" hyperref paketi " bölümünde

Eğer `\chapter[ ]{ }`, `\section[ ]{ }`, ... komutların zorunlu olmayan `[ ]` argümanı varsa, `\nameref` komutu bölümün (veya paragrafın) bu argümanındaki adını yazdırmaktadır.

nameref paketinin `\nameref` komutu da genişletilebilir. Paketin

`\Nameref{`*name*`}`

komutu sadece bölüm (veya paragraf) adı değil, aynı zamanda *name* adlı etiketin sayfa numarasını da yazdırmaktadır. Eğer `\Nameref` komutu metinde ek tanıtımsız doğrudan kullanılırsa, bu komut *sayfa* kelimesini İngilizce *on page* kelimesiyle yazdırmaktadır:

```
\Nameref{sec1} bölümünde
```

‘hyperref paketi’ on page 24 bölümünde

İngilizce kelime Türkçe kelime ile yazdırılması için metinde `\Nameref` ko-

mutu aşağıdaki gibi yeniden tanıtılmalıdır:

```
\renewcommand{\Nameref}[1]{\pageref{#1}.sayfa "\nameref{#1}"}
```

Şu halde, yukarıdaki örnek şöyle görülmektedir

`\Nameref{sec1} bölümünde`	24. sayfa "hyperref paketi" bölümünde

`\nameref` ve `\Nameref` komutunun aşırı-gönderme şeklinde çalışmaması için yapılacak gönderme aşağıdaki komutlu parantez arasına alınmalıdır

```
\begin{NoHyper}    ...    \end{NoHyper}
```

Bunu bir örnekle gösterelim

`\begin{NoHyper}`
`"\nameref{sec1}" bölümünde`
`\end{NoHyper}`
	"hyperref paketi" bölümünde ...

Başka dosyaya aşırı-gönderme

xr-hyper paketi xr paketinin genişletilmiş bir paketidir. Bu paket belgenin başlık kısmında hyperref paketinden önce tanıtılması gerekmektedir. xr-hyper paketi `file.tex` dosyasındaki bir etiketi aşırı-gönderme şeklinde kullanılmasına izin vermektedir. Bunun için, `file.tex` dosyası belgenin başlık kısmında aşağıdaki gibi tanıtılmalıdır

```
\externaldocument[prefix-]{file}[URL]
```

Komutun zorunlu olmayan *prefix* argümanı *file.tex* dosyasındaki bir etiketin giriş dosyasın etiketleriyle karışmaması için gereklidir. *file.tex* dosyasındaki `\label{`*name*`}` gibi işaretlenen bir nesneye giriş dosyasında `\ref{`*prefix-name*`}` olarak bir gönderme yapılmaktadır. Eğer *file.tex* dosyasın bir çıkış dosyası da varsa, bu dosya komutun zorunlu olmayan *URL* argümanında gösterilebilir. Örneğin, eğer *file.tex* dosyasın `*.pdf` tipli bir çıkış dosyası varsa, *URL* argümanda *file*.pdf ifadesi yazılabilir.

Aslında, metinden *file.tex* dosyasına bir gönderme yapıldığında LaTeX gönderme etiketi hakkındaki bilgiyi (yani bölüm ve sayfa numarası...)

`file.aux` dosyasından alır. Dolayısıyla, en az bir dış göndermeli metinin, çıkış dosyasındaki tüm göndermeleri sorunsuz gözükmesi için LaTeX 'de önce *file.tex* dosyasını, sonra ise giriş dosyasını derlemesi gerekir. Şimdi, buna bir örnek verelim.

Varsayılım ki, bir `picture.tex` dosyasında bir şekil `\label{pic:ws}` gibi etiketlenmiş ve `picture.pdf` dosyası `picture.tex` dosyasının `*.pdf` tipli çıkış dosyası olsun. Bu halde, giriş dosyasının başlık kısmında `picture.tex` dosyası şöyle tanıtılmalıdır

```
\externaldocument[xr-]{picture}[picture.pdf]
```

Böylece geçerli metinde `\autoref{xr-pic:ws}` olarak yapılan bir gönderme şöyle sonuç vermektedir

`\autoref{xr-pic:ws} 'e göre \dots` | Figure 1 'e göre ...

Aşırı-gönderme renkleri

Metinde bir göndermenin renkli olması için hyperref paketinin `colorlinks` seçeneğinin de gösterilmesi gerekmektedir. Seçenek gösterilmediği taktirde LaTeX bu seçeneği doğrudan

```
colorlinks=false
```

olarak alır. Dolayısıyla, bir aşırı-gönderme metni normal siyah renkte yazdırılır. Aşağıdaki seçenekler grubu bir gönderme rengini tanımlamaktadır.

Seçenek	vazifesi	false
`linkcolor`	metin içinde aşırı-gönderme rengi	red
`pagecolor`	metin içinde bir sayfaya aşırı-gönderme rengi	red
`filecolor`	bir yerel dosyaya aşırı-gönderme rengi	cyan
`citecolor`	kaynakçaya aşırı-gönderme rengi	green
`urlcolor`	çıkış dosyası URL tipli olan dosyaya aşırı-gönderme rengi	magenta
`anchorcolor`	etiket metninin rengi	black

Seçenek rengi color paketinin herhangi bir rengiyle değiştirilebilir veya

`\definecolor` komutuyla istenilen bir renk tanımlanıp, bu renk seçeneklere eklenebilir:

```
\usepackage{color}
\definecolor{darkgreen}{rgb}{0,.5,0}
\usepackage[colorlinks,filecolor=blue,citecolor=darkgreen]{hyperref}
```

1.11 Sayfa Dipnotu

Metinde bir dipnot LaTeX 'in

`\footnote[`*number*`]{`*text*`}`

komutuyla yazdırılmaktadır. Komutun zorunlu *text* argümanında dipnotun metni yazılmalıdır, zorunlu olmayan *number* argümanında ise dipnotun numarası özel olarak verilebilir. Bu argüman gösterilmediği taktirde LaTeX bu argümana `\footnote` değişkenin sıradaki numarasını verir. LaTeX `\footnote` komutu yazılan yere, bir üst indis olarak, dipnot için bir gönderme etiketini (yani dipnot numarasını) ve dipnot metnini ise geçerli sayfanın altına yazdırmaktadır. Bunu bir örnekle gösterelim

```
Dipnot için "footnote" komutu
kullanılmaktadır \footnote[17]{
bkz. [2]}
```

Dipnot için "footnote" komutu kullanılmaktadır [17]

[17] bkz. [2]

`\footnote` komutu sadece bir metin veya *minipage* (*bkz.* 2.8.4. bölüm, 70. sayfa) bloğunda kullanılabilir. Bir *kutu* (*bkz.* 2.8.1. bölüm, 66.sayfa) içinde bu komut kullanılamaz. *Kutu* bloğunda bir dipnot aşağıdaki komutlarla yazdırılabilir

`\footnotemark[`*number*`]`
`\footnotetext[`*number*`]{`*text*`}`

Bu komutların argümanı `\footnote` komutun argümanı gibidir.

`\footnotemark` komutu bir dipnota sadece bir gönderme etiketini yazdır-

maktadır. Bu komut metinin her yerinde kullanılabilir. Bir dipnotun metni ise `\footnotetext` komutuyla yazdırılabilir. Fakat bu komut *kutu*nun dışında olmalıdır. Bu durumda `\footnotemark` ve `\footnotetext` komutlarının *number* argümanının aynı olması gerekmektedir.

hyperref paketinde `\footnotemark` komutu, maalesef, aşırı-gönderme şeklinde çalışmaz. Dolayısıyla, metinde hyperref paketi tanıtılmışsa, `\footnotemark` komutu yerine `\footnote` komutunun kullanılması önerilmektedir. Çünkü hyperref paketi `\footnote` komutun bir aşırı-gönderme şeklinde çalışmasını desteklemektedir.

Aşağıdaki örnekte bir dipnot komutu `\fbox` komutuyla (*bkz.* 2.8.1. bölüm, 67.sayfa) yapılan bir *kutu* içinde ele alınmaktadır.

```
kutu'da \fbox{"footnote"
\footnote[12]{}}
\footnotetext[12]{bkz. [2]}
```

kutu'da "footnote" [12]

⋮

[12] bkz. [2]

`\footnote` komutu *minipage* bloğunda normal metine göre bağımsız olarak çalışmaktadır. Dolayısıyla, *minipage* bloğunda bir dipnot ayrıca oluşturulmaktadır ve onun numarası da bu bloğun içine yazdırılmaktadır. *minipage* bloğunda bir dipnot numarası da bağımsız olarak Latince harflerle yazdırılır. Bu numara (yani harf) `mpfootnote` değişkeninde saklanmaktadır. Dolayısıyla, bu numara `\setcounter{mpfootnote}{`n`}` ($n = 0, 1, \ldots 26$) komutuyla değiştirilebilir. Ayrıca `\footnote` komutunun normal metindeki numarası da `\setcounter{footnote}{`n`}` komutuyla değiştirilebilir.

minipage bloğunda bir dipnotun sayfanın bir dipnotu olarak yazdırılması için bu blokta `\footnotemark` (bir gönderme etiketi için) komutu ve blok dışında ise `\footnotetext` (dipnot metni için) komutu kullanmalıdır.

Bir dipnot geçerli sayfanın alt kısmında metinden bir yatay çizgiyle ayrılmaktadır. Bu çizgiyi

`\footnoterule`

komutu yazdırır. Çizginin boyutları (yani kalınlığı ve uzunluğu) `\renew-`

`command` komutuyla değiştirilebilir. Yatay çizgiyle birinci dipnot arasındaki düşey aralık

```
\footnotesep
```

komutuyla tanımlanmaktadır. Bu aralığın değeri `\setlength` komutuyla değiştirilebilir.

1.12 Kaynakça

Metinde **Kaynakça** LaTeX 'in aşağıdaki komutlu paranteziyle oluşturulmaktadır

```
\begin{thebibliography}{ text}
    bibitems
\end{thebibliography}
```

`thebibliography` bloğunun *text* argümanı kaynakça öğesinin numarası veya etiketi ile içeriği arasındaki boşluk sayısını göstermektedir. Örneğin,

```
\begin{thebibliography}{9999}
...
```

gibi tanımlanan *thebibliography* bloğunda kaynakça öğesinin numarası (veya etiketi) ile içeriği arasındaki boşluk sayısı: $\underbrace{9999}_{\sim 1cm}$ 'dir.

Kaynakçanın her öğesi

```
\bibitem[label]{id}
```

komutuyla başlamaktadır. Komutun zorunlu *id* argümanı bir gönderme etiketidir, yani metinden kaynakçanın bir öğesine *id* etiketiyle gönderme yapılmaktadır. Komutun zorunlu olmayan *label* argümanı gösterilmediği taktirde, kaynakça öğesi önüne köşeli parantez içine öğenin sıradaki numarası yazdırılır. Bu numara `enumiv` değişkeninde (*bkz.* 2.6. paragraf) saklanmaktadır ve `\setcounter` komutuyla değiştirilebilir. Eğer `\bibitem`

komutunun *label* argümanı varsa, kaynakça öğesinin önüne, numara yerine, *label* argümanı yazdırılır. Bu durumda, `enumiv` değişkenin değeri değişmez. *label* argümanındaki dayanıksız komutlar `\protect` komutuyla korunmalıdır.

Yukarıda söylendiği gibi, *label* argümanı yazılmadığı taktirde, kaynakça öğesinin önüne köşeli parantez içine öğenin sıradaki numarası yazdırılır. Aslında, bu numara da değiştirilebilir. Bunun için `\@biblabel` komutu belgenin başlık kısmında `\renewcommand` komutuyla yeniden tanımlanmalıdır. Bunu bir örnekle gösterelim: eğer belgenin başlık kısmına

```
\makeatletter
\renewcommand{\@biblabel}[1]{#1.}
\makeatother
```

tanıtımları yazılırsa, kaynakça öğesinin numarası parantezsiz olarak yazılır ve numaradan sonra bir nokta konulur.

LaTeX kaynakça için başlık olarak `book` ve `report` sınıfında **Bibliography** kelimesini; `article` sınıfında ise **References** kelimesini yazdırmaktadır. **Bibliography** ve **References** kelimeleri sırasıyla \bibname ve \refname komutlarında saklanmaktadır ve bu kelimeler `\renewcommand` komutuyla değiştirilebilir. `babel` paketinin *turkish* seçeneği, başlıkları sırasıyla **Kaynakça** ve **Kaynaklar** kelimeleriyle değiştirir.

Metinde bir kaynakça öğesine

`\cite[`*text*`]{`*id*`}`

komutuyla bir gönderme yapılabilir. Komutun *id* argümanı zorunludur. LaTeX, `\cite` komutu yazılan yere, köşeli parantez içinde zorunlu argümanı *id* olan kaynakça öğesinin numarası veya *label* (eğer bu argüman kaynakçada gösterilmişse) etiketini yazdırır. `\cite` komutun zorunlu olmayan *text* argümanı, köşeli parantez içerisinde, numaradan sonra virgül koyarak yazdırılır. Bu argüman göndermenin ayrıntısını göstermek için kullanılır. Örneğin, *text* argümanla kaynakça öğesinindeki bir paragraf veya sayfa numarasını belirtmek mümkündür. Aşağıdaki örnek `\cite` ve *thebibliography* bloğunun kullanılmasını göstermektedir.

```
⋮
∫ xdx = x²/2 olduğunu biliyo-
ruz (bkz.\cite[2.4.1.bölüm]{M})
⋮
\begin{thebibliography}{ab}
\bibitem[MB]{M} M.Balcı ,...
⋮
\end{thebibliography}
```

⋮

$\int xdx = \frac{x^2}{2}$ oldugunu biliyoruz (bkz. [MB, 2.4.1. bölüm])

⋮

Kaynakça

[MB] M.Balcı, ...

⋮

`\cite` komutun zorunlu *id* argümanında birkaç gönderme etiketi kendi aralarında virgülle ayrılarak yazılabilir. Dolayısıyla, bir komutla kaynakçaya birkaç gönderme yapılabilir. Bu durumda, göndermeler komutun zorunlu argümanında hangi sırada yazılmışsa, gönderme numaraları da, tek parantez içinde, o sırada yazdırılır. Gönderme numaraları artan sırada yazdırılması için belgenin başlık kısmında `cite` paketi tanıtılmalıdır. O halde, `\cite` komutunda birbirini izleyen sayıları birinci ve sonuncu sayı arasına bir tire işareti konularak yazdırılır. Örneğin, [1,5,3,4] yerine [1,3-5] yazdırılır.

1.13 Sayfa numarası

LaTeX 'de bir sayfa Arapça rakamla numaralandırmaktadır. Baş sayfa numarası 1 rakamıyla başlanır. Sayfa numaralandırması aşağıdaki tanıtımla değiştirilebilir

`\pagenumbering{`*format*`}`

Komutun *format* argümanına aşağıdaki seçeneklerden sadece biri yazılabilir:

`arabic` - sayfa Arapça (1,2,3,4,5, ...) rakamla numaralandırılır;

`roman` - sayfa Roma (i,ii,iii,iv,v,vi, ...) rakamla numaralandırılır;

`Roman` - sayfa büyük Roma (I,II,III,IV, ...) rakamla numaralandırılır;

`alph` - sayfa Latince (a,b,c,d,e,f,h,g, ...) harflerle numaralandırılır;

`Alph` - sayfa büyük Latince (A,B,C,D, ...) harflerle numaralandırılır

Seçenek gösterilmediği taktirde (yani metinde `\pagenumbering` komutu kullanılmazsa) LaTeX bu seçeneği `arabic` olarak alır.

NOT: `alph` ve `Alph` seçeneklerinde sayfa sayısı 26 dan fazla olmamalıdır.

1.14 Birkaç sütunlu metin

`\documentclass` (veya `\documentstyle`) komutun `twocolumn` seçeneğinde LaTeX tüm metini iki sütunlu olarak yazdırmaktadır. Metinin herhangi bir yerinden iki sütunlu metin başlatılması için

`\twocolumn[`*preface*`]`

komutu kullanılır. Bu durumda, yeni sayfadan (otomatik `\clearpage` komutu çalışır) önce zorunlu olmayan *preface* argümanı metin başı olarak tek sütun şeklinde, sonra metin iki sütun olarak yazdırılmaktadır. İki sütunlu metinden tekrar tek sütunlu metine dönmek için \onecolumn komutu kullanılmaktadır. Burada da tek sütunlu metin yeni sayfadan başlatılmaktadır.

İki sütunlu metinde sütunlar arasındaki uzaklık \columnsep komutuyla; sütunlar arasındaki düşey çizgi kalınlığı ise \columnseprule komutuyla tanımlanmaktadır. `\columnseprule` komutuna yeni bir değer verilmezse LaTeX bu değeri 0 (sıfır) olarak alır.

Komutların değeri `\setlength` komutuyla değiştirilebilir:

```
\setlength{\columnsep}{2.0cm}
\setlength{\columnseprule}{2mm}
```

tanıtımından sonra sütunlar arasındaki uzaklık `2.0cm`, sütunlar arası düşey çizgi kalınlığı ise `2mm` olarak belirlenmektedir.

1.14.1 multicol paketi

Önceki paragrafta görüldüğü gibi `\twocolumn` ve `\onecolumn` komutların her ikisi de metini yeni sayfadan başlatmaktadır. Oysa bu komutların kullanma alanını daraltmaktadır. `tools` sınıfın multicol paketinde komutlu

```
\begin{multicols}{n}[preface][skip]
  text
\end{multicols}
```

parantezi tanımlanandır. `multicols` bloğu metini yeni sayfadan başlatmadan gelen yerinden devam ettirmektedir. `multicols` bloğunda önce tek sütun olarak zorunlu olmayan *preface* argümanı, sonra ise *text* argümanı n sütunlu olarak yazdırılır. Sütunların sayısı n, en fazla 10 tane olabilir.

Eğer n sütunlu metin için geçerli sayfada *skip* argümanın değerinden veya \premulticols komutun değerinden az yer kalmışsa, n sütunlu metin, yeni sayfadan başlanır. `\premulticols` komutunun değeri değiştirilmediği taktirde LaTeX bu değeri `50pt` olarak alır.

`50pt≈17mm`

Eğer `multicols` bloğundan sonra geçerli sayfada \postmulticols komutunun değerinden az yer kalmışsa, metin yeni sayfadan başlanır. `\postmulticols` komutun değeri değiştirilmediği taktirde LaTeX bu değeri `20pt` olarak alır.

`20pt≈6.5mm`

Birkaç sütunlu metinden önce ve sonra \multicolsep komutunun değerine eşit olan bir düşey aralık bırakılmaktadır. `\multicolsep` komutunun değeri değiştirilmediği taktirde LaTeX bu değeri `12pt` olarak alır. İki sütunlu metinde olduğu gibi, n sütunlu metinde de sütunlar arasındaki uzaklık `\columnsep` komutuyla; sütunlar arasındaki düşey çizgi kalınlığı ise `\columnseprule` komutuyla tanımlanmaktadır.

LaTeX tüm sütunları aynı yükseklikte oluşturur. Son sütunun dolmasa bile, tüm sütunların yüksekliği eşit olarak arttırılabilir. Bunun için

`unbalance`

değişkeninin değeri arttırılmalıdır. Örneğin,

```
\setcounter{unbalance}{3}
```

tanıtımından sonra tüm sütunların yüksekliği son sütunun sonunda 3

tane boş satır eklenerek eşit olarak arttırılmaktadır.

NOT: `multicols` alanında `figure*` ve `table*` bloğu kullanılabilir.

1.15 Dizin

LaTeX metinde **Dizin**'i otomatik oluşturmaktadır. Bu süreç aşağıdaki birkaç adımdan oluşmaktadır:

- LaTeX2.09 'da `\documentstyle` komutunun `makeidx` seçeneği; LaTeX2$_\varepsilon$ 'de giriş dosyasın başlık kısmında `makeidx` paketi tanıtılır.
- `\makeindex` komutu giriş dosyasının başlık kısmında `\documentstyle` (veya `\documentclass`) ve `\begin{document}` komutları arasına yazılır.
- **Dizin**'de yer alacak her kelime ve simge metinin kullanıldığı yerde `\index` komutuyla aşağıda ele alınan "Dizin tabanı" adlı 1.15.1. alt paragrafında verilen kurallara göre hazırlanır.
- `\printindex` komutu **Dizin**'in yazdırılacağı yere yazılır. Tabii ki bu komut `\end{document}` komutundan önce yazılmalıdır.
- Giriş dosyası LaTeX 'de okutulur. Bu durumda, ana dizinde `idx` dosyası oluşur. Örneğin, eğer giriş dosyası

  ```
  C:\4TEX5.0\BOOK\kitap.tex
  ```

  olarak `kitap.tex` adında ise, `BOOK` klasöründe `kitap.idx` dosyası oluşur.
- **MakeIndex** programı çalıştırılır:

  Utilities → *MakeIndex* → *Run MakeIndex* → *OK*

  Eğer bilgisayarın LaTeX programasında *Utilities* veya *MakeIndex* seçeneği yoksa, bilgisayarın "*komutlar satırı*"na (*Başlat* → *Çalıştır*)

  ```
  makeindex.exe myfile.idx
  ```

  komutu yazılıp, *Tamam* düğmesi tıklanır.
  Bu durumda, LaTeX `idx` tipli dosya yardımıyla `ind` tipli bir dosya oluşturur, yani ana dizinde `kitap.ind` dosyası oluşur.

- Giriş dosyası LaTeX 'de bir daha okutulur. Bu durumda, LaTeX `*.ind` dosyasından **Dizin**'de yer alacak her kelime (veya simge) ve onun sayfa numarasını `\printindex` komutuyla metine yazdırır.

Eğer her şey doğru yapılmışsa, LaTeX hata vermemelidir. Yukarıdaki şıklardan biri yanlış yapılmışsa, program bir hata verir. Özellikle, Dizinde yer alacak her kelime ve simge `\index` komutuyla hatasız hazırlanmalıdır.

NOT: **MakeIndex** programı çalıştırılmadan önce ve sonra giriş dosyası LaTeX 'de okutulması gerekmektedir :
Compile→Utilities→MakeIndex →Run MakeIndex→OK→Main menu→Compile

1.15.1 Dizin tabanı

Dizin'de yer alacak her kelime veya simge metinde kullanıldığı yerde

`\index{`*text*`}`

komutuyla tanıtılır. Bir kelime veya simge komutun *text* argümanına aşağıda verilen kurallara göre yazılmalıdır.

`makeindex` komutunda `!`, `@` ve `|` simgeleri yönlendirici olarak kullanılır. Bu paragrafta, yönlendirici harfi içermeyen Dizinde yer alacak kelime ve simgeler ele alınmaktadır.

`\index` komutunun *text* argümanında yönlendirici harfler yoksa, bu komut şöyle çalışır:

Sayfa	`\index{`*text*`}`	Dizin
iii. sayfa	`\index{Alpha}`	Alpha, iii
viii. sayfa	`\index{alpha}`	alpha, viii, xi, 17
xi. sayfa	`\index{alpha}`	alpha bet, 35
	`\index{Alphabet}`	Alphabet, xi
17. sayfa	`\index{alpha}`	alphabet, 21
	`\index{alphau}`	alphau, 17
21. sayfa	`\index{alphabet}`	
	`\index{alphabet}`	
35. sayfa	`\index{alpha bet}`	

Tablodan görüldüğü gibi, 21. sayfada `\index{alphabet}` komutu iki defa yazılmasına rağmen Dizine 21 rakamı bir defa yazdırılır.

Dizinde birkaç alt girişler de varolabilir. `\index` komutunda her alt giriş yönlendirici `!` simgesiyle işaretlenmektedir:

5. sayfa	`\index{küme!kapalı}`	kültür
22. sayfa	`\index{küme}`	Asya, 47
25. sayfa	`\index{küme!açık}`	Avrupa, 25
	`\index{kültür!Avrupa}`	küme, 22
47. sayfa	`\index{kültür!Asya}`	açık, 25
		kapalı, 5

`\index` komutunda yönlendirici `!` simgesi 2 defa kullanılsa, alt girişin bir alt girişi oluşur:

5. sayfa	`\index{Integral!katlı!iki}`	İntegral
13. sayfa	`\index{Integral!katlı!üç}`	katlı
16. sayfa	`\index{Integral!yüzey}`	iki, 5
		üç, 13
		yüzey, 16

LaTeX ve `makeindex` sadece 2. mertebeden altgirişi destekleyebilir, yani yönlendirici `!` simgesi 3. defa kullanılamaz.

Dizinde yer alacak bir kelime süregelen sayfalarda tekrarlanırsa, bu kelime ilk yazılan yere `\index{...|(}` komutu ve son yazılan yerine `\index{...|)}` komutu yazılmalıdır:

iv. sayfa	`\index{türev|(}`	türev, iv–x, 20
x. sayfa	`\index{türev|)}`	İntegral
20. sayfa	`\index{türev}`	iki katlı,20
	`\index{Integral!iki katlı|(}`	üç katlı,25-32
	`\index{Integral!iki katlı|)}`	
25. sayfa	`\index{Integral!üç katlı|(}`	
30. sayfa	`\index{Integral!üç katlı}`	
32. sayfa	`\index{Integral!üç katlı|)}`	

`\index{...|(}` ve `\index{...|)}` komutları aynı sayfaya düşseler bile `\makeindex` komutu doğru çalışır.

Dizinde bir sayfa numarası yerine bir çapraz gönderme de yapılabilir. Bunun için `\index{...|see{...}}` komutu kullanılır:

5.sayfa	`\index{türev}`	türev, 5
5.sayfa	`\index{türev!kısmı|see{kısmı,türev}}`	kısmi, *bkz* kısmi, türev

Dizinde bir kelime veya simge alfabe sırasıyla değil de belli bir yere yazdırılması için `\index` komutunun *text* argümanında yönlendirici `@` simgesi kullanılır. Örneğin, `\index{`σ`@`τ`}` 'de τ simgesi alfabe sırasıyla σ simgesinin yerine yazdırılır. Aşağıdaki örnekte `@` simgesinin işlevi daha açık görülmektedir:

17. sayfa	`\index{Kitap@{\bf Kitap}}`	alpha, 22
18. sayfa	`\index{Kitap}`	α, 47
22. sayfa	`\index{alpha}`	alphaa, 52
32. sayfa	`\index{otuz}`	Kitap, 18
45. sayfa	`\index{otuz-bir}`	**Kitap**, 17
47. sayfa	`\index{alpha@$\alpha$}`	Kitapa, 64
52. sayfa	`\index{alphaa}`	otuz, 32
63. sayfa	`\index{otuz@xx}`	xx, 63
64. sayfa	`\index{Kitapa}`	otuz-bir, 45

Dizinde bazı sayfa numaraları özel biçimle yazdırılabilir. σ herhangi bir komut olmak üzere, `\index{...|`σ`}` komutu sayfa numarasını `\`σ{`$n$`}` şeklinde yazdırır. Örneğin, `\index{kitap|it}` komutu 5.sayfaya yazılmışsa, bu kelime Dizine şöyle yazdırılır: kitap, *5*, yani sayfa numarası *italik* yazısıyla yazdırılır. Benzer şeklinde, `\index{...|(`$\sigma$`}` komutu sayfa numarasını `\`$\sigma${`n`–`m`}` şeklinde yazdırır. Örneğin, `\index{kitap|(bf}` ve `\index{kitap|)}` komutları sırasıyla 7. ve 12. sayfalara yazılmışsa, bu kelime Dizine şöyle yazdırılır: kitap, **7-12**, yani sayfa numarası **koyu** yazıyla yazdırılır.

Aslında σ komutu da özel olarak tanımlanabilir. Bu halde, belgenin başlık kısmına σ komutunun tanıtımı eklenmelidir. Örneğin, σ şöyle tanımlanmış olsun

```
\newcommand{\aaa}[1]{{\it #1}}
\newcommand{\bbb}[1]{#1b}
```

(1.durumda $\sigma =$`aaa` olarak; 2.durumda $\sigma =$ `bbb` olarak tanımlanıyor). Şu halde, Dizinde sayfa numaraları, bu özel biçimle, şöyle yazdırılır

5. sayfa	`\index{hukuk|aaa}`	hukuk, *5*, 7b
7. sayfa	`\index{hukuk|bbb}`	husus, 8, *21–29*
8. sayfa	`\index{husus}`	
21. sayfa	`\index{husus|(aaa}`	
29. sayfa	`\index{husus|)}`	

1.15.2 Dizinde yönlendirici (`!`, `@` ve `|`) simgeler

Dizinde yer alacak bir kelime veya simge yönlendirici `!`, `@` ve `|` simgelerinden en az birini içeriyorsa, yönlendirici simgenin önüne `"` simgesi yazılmalıdır. Örneğin, **bkz.!** kelimesi Dizine `\index{bkz."!}` şeklinde yazılır. Üstelik `"` simgesinin kendisi de bir yönlendirici simgesine dönüşmektedir. Bu simge `makeindex` komutunda `\verb+"+` olarak yazılabilir:

4.sayfa	`\index{ünlem işareti ("!)}`	ünlem belgisi (!), 4
5.sayfa	`\index{ünlem işareti ("!)!yüksek sesle}`	yüksek sesle, 5
7.sayfa	`\index{tırnak işareti (\verb+""+)}`	tırnak işareti (""), 7

1.15.3 Türkçe Dizin

`\printindex` komutu yeni sayfadan iki sütun olarak **Index** başlığıyla *Dizini* yazdırmaktadır. `\index` komutunun argümanında kullanılan `\see` komutu ise Dizinde İngilizce *see* kelimesini yazdırır. İngilizce **Index** ve *see* kelimeleri sırasıyla

`\indexname` ve `\seename`

komutlarında saklanmaktadır. Bu kelimeler `\renewcommand` komutuyla değiştirilebilir. `babel` paketinin *turkish* seçeneği:

```
\usepackage[latin5]{inputenc}
\usepackage[turkish]{babel}
```

Index ve *see* kelimeleri sırasıyla Türkçe **Dizin** ve *bkz.* olarak yazdırmaktadır. Bu durumda, `\index` komutunun *text* argümanında Türkçe kelimeler de kullanılabilir. Fakat, Latince'de olmayan Türkçe **ç**, **ğ**, **ı**, **ö**, **ş** ve **ü** harfleri **Dizin**de alfa nümerik sırasıyla yazdırılmamaktadır. Örneğin, **ü** harfiyle başlanan Türkçe kelimeler listesi **u** harfiyle başlanan kelimeler listesinden sonra gelmemektedir. Çünkü, `\makeindex` komutu bu harfleri

bir simge olarak anlar, dolayısıyla, onları simgelere göre alfa nümerik sırasıyla yazdırmaktadır. Bu durumda, yönlendirici `@` simgesi aşağıdaki gibi kullanılabilir:

4. sayfa	`\index{cizgi@çizgi}`	cizgi, 3
3. sayfa	`\index{cizgi}`	çizgi, 4
7. sayfa	`\index{cizgiy}`	cizgiy, 7
8. sayfa	`\index{isin@ışın}`	isin, 5
5. sayfa	`\index{isin}`	ışın, 8
9. sayfa	`\index{sutun@sütun}`	ugras, 14
11. sayfa	`\index{sutun}`	uğraş, 14
14. sayfa	`\index{ugras}`	sutun, 11
14. sayfa	`\index{ugras@uğraş}`	sütun, 9

Bölüm 2

Metin biçimi

2.1 TEX 'in ölçü birimleri

TEX 'de **mm**, **cm**, **in**, **pt**, **em** ve **ex** gibi ölçü birimler kullanılmaktadır. TEX ve LATEX 'in bu ölçü birimleri aşağıdaki tabloda verilmektedir

Tablo 2.1: TEX ve LATEX 'in ölçü birimleri

`mm`	millimetre $\approx 1/25$ inch	␣
`cm`	centimetre = 10 mm	␣
`in`	inch = 25.4 mm	␣
`pt`	point $\approx 1/72$ inch $\approx 1/3$ mm	␣
`em`	tahminen geçerli "`M`" harfin eni	␣
`ex`	tahminen geçerli "`x`" harfin yüksekliği	␣

Eğer metinde standart olmayan belli bir ölçü sık-sık kullanılsa, bu ölçü aşağıdaki komutla yeni ölçü birimi olarak tanımlanabilir

`\newlength{`*cmd*`}`

`\newlength` komutu sıfır uzunluklu yeni *cmd* ölçüsünü tanımlamaktadır. Bir ölçü birimin değeri aşağıdaki iki komutun herhangi biriyle değiştirilebilir.

`\setlength{`*cmd*`}{`*lendth*`}`
`\addtolength{`*cmd*`}{`*lendth*`}`

`\setlength` komutu *cmd* ölçü birimine *lendth* değeri vermektedir, `\addtolength` komutu ise *cmd* argümanın değerine *lendth* değeri eklemektedir. Komutların *lendth* argümanına yukarıdaki ölçü birimleri (-2.4cm, 5mm, 1.2ex, ...) gibi metin şeklindeki `1.2\textwidth` ölçü birimleri de yazılabilir. Örneğin, `\newlength{`*uzunluk1*`}` ve `\newlength{`*uzunluk2*`}` komutları sıfır uzunluklu sırasıyla *uzunluk1* ve *uzunluk2* adlı yeni iki ölçü birimini tanımlamaktadır;

`\setlength{`*uzunluk1*`}{`*-0.5pt*`}` ve `\addtolength{`*uzunluk2*`}{1.3\textwidth}`

komutları ise sırasıyla *uzunluk1* ölçüsüne *-0.5pt* değeri vermekte ve *uzunluk2* ölçüsünün değerine ise `1.3\textwidth` değeri eklemektedir.

Bir düşey ölçü birimi, genelde, esnek oluyor. Dolayısıyla, `\setlength` ve `\addtolength` komutlarının *lendth* argümanında bir düşey ölçü biriminin sadece *lendth* değeri değil, onun değişebilen bir aralığını da gösterilmelidir. Örneğin,

```
\setlength{\parskip}{5pt plus 2pt minus 1pt}
```

komutu `\parskip` düşey ölçü birimin değerini `5pt` olarak tanımlamaktadır, fakat gerekirse (sayfadan-sayfaya geçiş, ...) LaTeX bu değeri `2pt` 'e artırabilir veya `1pt` 'e azaltabilir.

2.2 Boş aralık koymak

2.2.1 Yatay aralık

LaTeX 'de aşağıdaki *standart* yatay aralıklar vardır:

`\␣`	`\,`	`\quad`	`\qquad`

Komutlar metinde uzunluğu sırasıyla `␣` , `0.1667em` , `1em` ve `2em` olan yatay aralığı tanımlamaktadır. Bunu bir örnekle gösterelim

```
A \␣ B \, C \quad D \qquad E
```

A B C D E

Standart olmayan *length* uzunluklu bir yatay aralık

`\hspace{`*length*`}`	`\hspace*{`*length*`}`

komutlarıyla tanımlanabilir:

```
A \hspace{2.1cm} B
```

A B
2.1*cm*

`\hspace` komutu hareketli argümanlarda dayanıksızdır, `\hspace*` komutu ise dayanıklıdır. Dolayısıyla, zorunlu olan bir yatay aralık yıldızlı komutla verilmelidir.

NOT: `\hspace` ve `\hspace*` komutu satır sonuna düşse aralık iptal edilir.

`\hfill` komutu bir sınırsız yatay aralık oluşturur, yani komuttan sonraki her şey satırın sağına yaslanır.

`\hrulefill`	`\dotfill`

komutları `\hfill` komutu gibi olup, `\hrulefill` komutu yatay aralığa bir altçizgi yazdırmaktadır, `\dotfill` komutu ise yatay aralığı noktalarla doldurmaktadır:

```
A \hfill B
A \hrulefill BB
A \dotfill BBB
```

A B
A BB
A BBB
textwidth

LaTeX iki cümle arasına iki kelime arasındaki gibi bir yatay aralık koymaktadır. İki cümle arasındaki aralık, genelde, bir az büyük olmalıdır. Bu özel aralık `\@` komutuyla verilebilir. O iki kelime arasına verilen aralıktan bir az uzundur:

```
iki kelime arasında
"␣" dir. \@ Fakat ...
```

iki kelime arasında "␣"dir. Fakat ...

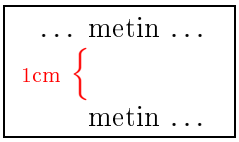
2.2.2 Düşey aralık

Metinde *length* uzunluklu bir düşey aralık

`\vspace{`*length*`}`	`\vspace*{`*length*`}`

komutlarıyla koyulabilir. Eğer bir satırda `\vspace` veya `\vspace*` komutundan sonra metin varsa, düşey aralık satır dolduktan sonra koyulmaktadır. `\vspace` komutu hareketli bir argümanda dayanıksızdır, `\vspace*` komutu ise dayanıklıdır. Dolayısıyla, zorunlu olan bir düşey aralık yıldızlı komutla koyulmalıdır. `\vspace` komutuna bir örnek verelim.

```
... metin\vspace{1.0cm} ... \\
metin ...
```



Eğer `\vspace` komutu sayfa başına veya sonuna düşerse, düşey aralık iptal edilir.

`\vfill` komutu sınırsız bir düşey aralık oluşturur, yani komuttan sonraki her şey sayfanın altına yaslanmaktadır.

`\addvspace{`*length*`}` komutu `\vspace` komutu gibi olup, o iki satır arasındaki düşey aralığına *length* değeri eklemektedir.

Standart yatay aralıklar gibi, standart düşey aralıklar da vardır:

`\smallskip`	`\medskip`	`\bigskip`

Komutların değeri sırasıyla `3pt`, `6pt` ve `12pt` 'dir. Bu değerler sırasıyla

`\smallskipamount`	`\medskipamount`	`\bigskipamount`

komutlarında saklanmaktadır. Bunlar `\setcounter` komutuyla değiştirilebilir.

2.3 Paragraf biçimi

LaTeX geçerli metinde boş bir satırı veya `\par` komutunu bir paragrafın sonu olarak kabul etmektedir. Dolayısıyla, metinde bir boş satır veya `\par`

komutu bir paragrafın bitmesine ve yeni bir paragrafın başlanmasına neden olmaktadır. Sadece bir paragraf biçimini etkiyebilen bir komut etki dairesi paragraf sonunda bir boş satırdan sonra kapatılmalıdır:

```
{\small
Paragraf başı ...
⋮
... burası paragraf sonudur.
boş satır →
}
```

Bir paragraftan önce iki satır arasındaki düşey aralığa `\parskip` komutunun değerinde bir düşey aralık koyulur. `\parskip` komutuna yeni bir değer verilmezse LaTeX bu değeri standart sınıflarda (`letter` sınıfı hariç) 0 (sıfır) olarak alır. Bu değer `\setcounter` komutuyla değiştirilebilir.

2.3.1 Biçim düzeninin değiştirilmesi

LaTeX bir paragrafta iki kelime arasına belli bir kurallara göre bir yatay aralık koymaktadır; her satırı sayfanın soluna yaslamaktadır. Çıkış dosyasında bir satıra sığmayan bir kelime "-" tire simgesiyle ikiye bölünüp, ikinci parçasıyla sıradaki satır başlanmaktadır. Bazen bir uzun kelime ikiye bölünmeyebilir, örneğin "kutu"lar bölünemez. Dolayısıyla, böyle bir kelime bir satır sonuna düşse, bu satır diğer satırlara göre biraz uzun olmaktadır.

Böyle durumlarda, problem yaratan kelime satır sonuna düşmemesi için bu paragraf yeniden yazılmalıdır. Eğer bu mümkün değilse, bu taktirde TeX 'in böyle durumlarda satırı devam ettirmesini tercih eden kriteri zayıflatılmalıdır. Bu kriter TeX 'in `\tolerance` komutunun değeriyle tanımlanmaktadır. Bu komut'a yeni bir değer verilmezse TeX bu değeri `200` olarak alır. Eğer bu değer artırılsa,

- metinin (veya kelimenin) bir satır sonunda kesilebilme olasılığı daha da artır;
- bir satır sonuna yerleşemeyen kelime, önceki satırlarda olan kelimeler arasına biraz uzun aralıklar verilerek, yeni satıra alınmaktadır

`\tolerance` komutun değeri, örneğin

```
\tolerance=500
```

komutuyla değiştirilebilir. Eğer böyle bir komut belgenin başlık kısmına yazılırsa, bu komut tüm metin için geçerli olur. Öte yandan, bu komut etkisi metinin herhangi bir yerinde `\fussy` komutuyla iptal edilebilir.

`\tolerance` komutun en büyük 10000 değeri alındığında, bu komut yerine `\sloppy` komutu kullanılabilir. `\sloppy` komutu metinin (veya kelimenin) bir satır sonunda kesilebilme olasılığı en büyük almaktadır. Metinin sadece bir parçasına ait olan `\sloppy` komutu

```
\begin{sloppypar} ... \end{sloppypar}
```

bloğuyla değiştirilebilir.

2.3.2 Satır başı boşluğu

LaTeX her paragrafın (bölümün ilk paragrafı hariç) ilk satırında bir satır başı boşluğu oluşturmaktadır. Boşluğun boyutu `\parindent` komutunda saklanmaktadır. Bu değer `\setcounter` komutuyla değiştirilebilir.

`\noindent` komutu bir paragrafın satır başı boşluğunu iptal etmektedir. Bunun için, `\noindent` komutu paragraf başına yazılmalıdır.

`\indent` komutu ise `\parindent` komutunun değerine eşit olan bir satır başı boşluğu oluşturmaktadır. Bu komut bir satırda standart bir satır başı boşluğu gerekli olduğu durumda kullanılabilir. Fakat `\indent` komutu bölümün ilk paragrafında çalışmaz. Böyle durumlarda bir satır başı boşluğu `\hspace*{\parindent}` komutuyla oluşturulabilir (*bkz.* 2.2.1.paragraf, 45.sayfa). Metinin tüm bölümlerinin ilk paragrafında bir satır başı boşluğu oluşturulması için `tools` sınıfının `indentfirst` paketi belgenin başlık kısmında tanıtılmalıdır:

```
\usepackage{indentfirst}
```

2.3.3 Satırlar arasındaki boşluk

TeX 'de metin satırları arasındaki boşluk `\baselineskip` komutuyla ayarlanır. Bu komut bazı gerekli durumlarda satırlar arasına otomatik olarak gerekli boşluğu eklemektedir. `\baselineskip` komutuna yeni bir değer

verilmezse TEX bu değeri `12pt` olarak alır. Bu değerin metin başında değiştirilmesi önerilmemektedir. Çünkü, metinde yazı boyutunu değiştiren bir komut kullanıldığı ilk yerde satırlar arası boşluk yine eski durumuna dönmektedir. Böyle durumda, iki komşu satır arasındaki boşluk

`\baselinestretch`

komutuyla değiştirilebilir. Standart sınıflarda bu komutun değeri `1` 'dir, yani standart 1 aralığı tanımlamaktadır. `\baselinestretch` komutunun değeri `\renewcommand` komutuyla değiştirilebilir. Eğer bu değer belgenin başlık kısmında değiştirilmişse, bu tüm metin için geçerli olur, eğer bu değer metin içinde değiştirilmişse, komutun etkisi ancak bir yazı boyutu değiştirildikten sonra başlamaktadır. Dolayısıyla, bu değerin değiştirilmesinden hemen sonra komutun etkisinin başlaması için, örneğin, `\normalsize` komutu kullanılabilir:

```
\renewcommand{\baselinestretch}{2}\normalsize
```
[1]

Bu tanıtımdan hemen sonra metin normal yazı boyutuyla ve standart 2 aralıkla yazdırılır. `\baselinestretch` komutun etki dairesi `{ ... }` küme paranteziyle sınırlandırılabilir. Fakat kapatma parantezi bir boş satır veya `\par` komutundan sonra yazılmalıdır.

İki komşu satır arasındaki boşluk tüm metin için

`\linespread{`*factor*`}`

komutuyla da değiştirilebilir. `\linespread` komutu belgenin başlık kısmına yazılmalıdır. `\baselinestretch` komutunda olduğu gibi, `\linespread` komutunun *factor* argümanın `1.5` (veya `2`) değeri MS Word metinin `1.5` (veya `2`) satır aralığına eşdeğerdir.

2.3.4 TEX 'de sözcük hecelemesi

TEX 'de sözcük hecelemesi otomatik ayarlanır. Fakat TEX Türkçe hecelemeyapamaz. Bu yüzden, eğer satırın sonunda kesme ile ayrılması gereken bir sözcük varsa, `\-` komutunu kullanarak, bu sözcüğü sizin hecelemeniz gerekecektir. Örneğin, *türevlenebilir* sözcüğünün kesilebilme yerleri

```
tü\-rev\-le\-ne\-bi\-lir
```

[1] *bkz.* 99. sayfa.

şeklinde işaretlenebilir. TEX, gerekli durumda, bu hecelemenin uygun bir yerini seçip, sözcüğü o yerinden kesmektedir.

Sık-sık kullanılan sözcüklerin hecelenmesi belgenin başlık kısmında

`\hyphenation{`*words*`}`

komutuyla tanıtılabilir. Komutun *words* argümanında:

- sözcünün hecelemesi sadece bir "-" tire simgesiyle gösterilir;
- sözcünün hecelenmemiş yerinden kesilmesine izin verilmemektedir;
- birkaç sözcük kendi aralarında ␣ aralığıyla ayrılarak tanıtılabilir.

Örneğin

```
\hyphenation{Micro-soft Türkiye}
```

tanıtımı *Microsoft* sözcüğünü sadece *soft* yerinden kesilmesine izin verir ve *Türkiye* sözcüğünün hiçbir yerinden kesilmesine izin vermemektedir.

2.3.5 Satır kesilmesi

Metinde bir satır zorunlu olarak

`\\[`*length*`]`	`\\*[`*length*`]`

komutların herhangi biriyle kesilebilir. Komutların *length* argümanı zorunlu değildir. Bu argümanda ek düşey aralık gösterilebilir. Örneğin, `\\[1.5cm]` komutu geçerli satırı keserek yeni satır için `1.5cm` ek düşey aralık vermektedir. Yıldızlı `\\*` komutu sıradaki satırın yeni sayfaya yazılmasına izin vermemektedir:

```
Türkiye Cumhuriyeti \\*[2mm]
Trabzon, 61080 ...
```

Örnekte yıldızlı `\\*[2mm]` komutu sıradaki satıra `2mm` ek düşey aralık vererek, bu satırı yeni sayfaya yazdırılmasına izin vermemektedir. Dolayısıyla, bu iki satır aynı sayfaya yazdırılır.

`~` komutu iki sözcük arasında satır kesilmesine izin vermemektedir:

```
... bkz~Şekil~1.2 ...
```

Bu durumda "*bkz Şekil 1.2*" sözcükler kesilmeden bir satıra yazdırılmaktadır.

Aşağıdaki komutlar bir metnin birkaç satıra bölünmesinde kullanılır.

`\linebreak[`n`]`	`\nolinebreak[`n`]`

`\linebreak` komutu geçerli satırın bitmesine destek verirken;

`\nolinebreak` komutu tersine, satırın devam etmesine destek verir. Komutların n argümanı zorunlu değildir. Bu argümanın 0,1,2,3 ve 4 değerleri komutun *güçünü* tanımlamaktadır. n büyük oldukça komutun zorunlu olarak çalışma olasılığı da büyür; değerin $n = 0$ olması komutun kayıp olmasına ve $n = 4$ olması ise komutun zorunlu olarak çalışmasına eşdeğerdir. Argüman gösterilmediği taktirde LaTeX bu argümanı `n=4` olarak alır.

2.4 Paragraf biçimini yönetmek

Sayfa biçim düzeni tek yüzlü metinde

```
\raggedbottom
```

komutuyla, iki yüzlü metinde ise,

```
\flushbottom
```

komutuyla ayarlanmaktadır. Bir sayfa düzenin değiştirilmesi için bu sayfaya sadece `\raggedbottom` ve `\flushbottom` komutlarından gerekli olanı yazılmalıdır.

2.4.1 Metinin sayfalara bölünmesi

Aşağıdaki iki komut metinin sayfalara bölünmesinde kullanılır

`\pagebreak[`n`]`	`\nopagebreak[`n`]`

`\pagebreak` komutu geçerli sayfanın bitmesine destek verirken;

`\nopagebreak` komutu tersine, onun devam etmesine destek verir. Komutların n argümanı zorunlu değildir. Bu argümanın 0,1,2,3 ve 4 değer-

leri komutun *güçünü* tanımlamaktadır. n büyük oldukça komutun zorunlu olarak çalışma olasılığı da büyür; değerin $n = 0$ olması komutun kayıp olmasına ve $n = 4$ olması ise komutun zorunlu olarak çalışmasına eşdeğerdir. Argüman gösterilmediği taktirde LaTeX bu argümana `n=4` olarak alır.

NOT: Eğer `\nopagebreak` komutuyla bir sayfa daha da uzunlaştırılmaya çalışılırsa, LaTeX bunu iptal edecektir.

2.4.2 Metin yüksekliğinin değiştirilmesi

Bir sayfada metin alt sınırı

`\enlargethispage{`*length*`}`	`\enlargethispage*{`*length*`}`

komutlarıyla değiştirilebilir. Yıldızlı komut dayanıklıdır. *length* argümanın eksi değerinde metinin alt sınırı yukarıya çekilmektedir.

2.4.3 Yeni sayfa başlatmak

Geçerli sayfa durdurulup yeni sayfa başlatılması için

`\newpage`	`\clearpage`	`\cleardoublepage`

komutlarından herhangi biri kullanılabilir. `\newpage` komutu geçerli sayfayı durdurup, metini yeni sayfadan devam ettirmektedir. `\clearpage` ve `\cleardoublepage` komutları geçerli sayfayı durdurup, yeni sayfaya önce yazdırılmayan tüm kayan nesneleri (tablo, şekil, ... ; *yine bkz.* 2.11.1.paragraf, 84.sayfa) yazdırır, sonra yeni sayfadan metini devam ettirir. `\cleardoublepage` komutunun bir özelliği daha vardır. Bu komut metini tek numaralı sayfadan devam ettirir. Dolayısıyla, yeni sayfa numarası çiftse, bu sayfa boş bırakılıp metin tek numaralı sayfadan başlatılmaktadır.

NOT: Birkaç sütunlu metinde `\newpage` komutu sayfa yerine sadece sütunu kesmektedir.

2.5 Madde işaretleri ve numaralandırma

LaTeX çeşitli madde işaretlerini desteklemektedir. Madde işaretlerinin her öğesine

`\item[`*label*`]`

komutuyla başlanması gerekmektedir. Komutun *label* argümanı zorunlu değildir. *label* argümanı liste öğesi önüne onun bir adı gibi yazdırılır. Bu argüman gösterilmediği taktirde LaTeX öğe önüne onun sıradaki numarasını veya işaretini yazdırır. Maddelerin her öğesi de birkaç alt maddeler içerebilir.

2.5.1 İşaretli listeler

LaTeX 'de bir işaretli liste komutlu

`\begin{itemize}` *items* `\end{itemize}`

paranteziyle oluşturulmaktadır. Listenin her öğesi `\item` komutuyla başlanmalıdır. LaTeX bir öğe önüne onun işaretini veya varsa *label* argümanını yazdırmaktadır. İşaretlenen bir listede dört düzeye kadar alt liste oluşturulmasına izin verilmektedir. Her alt listede bir öğe işaretini sırasıyla

`\labelitemi` `\labelitemii` `\labelitemiii` `\labelitemiv`

komutları yazdırır. Bu işaretler aşağıdaki tabloda ele alınmaktadır

düzey	işaret	komut
1	•	`\textbullet`
2	–	`\bfseries\textendash`

3	$*$	`\textasteriskcentered`
4	$\cdot$	`\textperiodcentered`

Komutların hepsi `\renewcommand` komutuyla değiştirilebilir, yani bir işaret istenilen başka bir işaretle değiştirilebilir. Aşağıda dört alt düzeyli işaretlenen bir liste bir örnekle gösterilmektedir

```
\begin{itemize}
  \item 1.düzey
  \item[31ac] 1.düzey
  (label kullanılmış)
   \begin{itemize}
    \item 2.düzey
    \item 2.düzey
     \begin{itemize}
      \item 3.düzey
      \item 3.düzey
      \item[[13sd]] 3.düzey
       (label kullanılmış)
        \begin{itemize}
         \item 4.düzey
         \item 4.düzey
        \end{itemize}
      \item 3.düzey
     \end{itemize}
    \item 2.düzey
    \item[N2fh] 2.düzey
    (label kullanılmış)
   \end{itemize}
  \item 1.düzey
\end{itemize}
```

- 1.düzey

31ac 1.düzey (*label* kullanılmış)

  - 2.düzey
  - 2.düzey
    - 3.düzey
    - 3.düzey

    [13sd] 3.düzey (*label* kullanılmış)

      - 4.düzey
      - 4.düzey
    - 3.düzey
  - 2.düzey

  N2fh 2.düzey (*label* kullanılmış)

- 1.düzey

2.5.2 Numaralı liste

LaTeX 'de bir numaralanan liste komutlu

`\begin{enumerate}` *items* `\end{enumerate}`

paranteziyle oluşturulmaktadır. Listenin her öğesi `\item` komutuyla başlanmalıdır. LaTeX bir öğe önüne onun sıradaki numarasını veya varsa

label argümanı yazdırmaktadır. Numaralı bir listede, işaretli listede olduğu gibi, dört düzeye kadar alt liste oluşturulmasına izin verilmektedir. Her alt listede bir öğe numarası sırasıyla

```
\enumi      \enumii      \enumiii      \enumiv
```

değişkenlerinde saklanmaktadır. Eğer bir `\item` komutunun *label* argümanı varsa, bu komut kendi düzeyinin değişken değerini değiştirmez. Dolayısıyla, sadece argümansız verilen `\item` komutu kendi düzeyinin değişken değerini bir birim arttırmaktadır.

Her alt düzey numarasını sırasıyla

```
\labelenumi  \labelenumii  \labelenumiii  \labelenumiv
```

komutları yazdırır. Bu komutlar yukarıdaki değişkenlerden her alt düzeyin sıradaki numarasını alarak yazdırır. Dolayısıyla, bir öğe numarasının görüntüsü bu öğeye yerleşen düzeye bağlıdır. Düzeylerdeki öğeler aşağıdaki gibi numaralandırılmaktadır:

1.düzey	1. , 2. , 3. , ...
2.düzey	(a), (b), (c), ...
3.düzey	i. , ii. , iii. , ...
4.düzey	A. , B. , C. , ...

Tablodan görüldüğü gibi, birinci düzey öğeleri Arapça rakamlarla, ikinci düzey öğeleri parantez içine alınan Latince rakamlarla, üçüncü düzey öğeleri Roma rakamlarıyla ve dördüncü düzey öğeleri ise büyük Latince rakamlarla numaralandırılmaktadır.

Yukarıdaki komutların hepsi `\renewcommand` komutuyla değiştirilebilir, yani bir düzeydeki numara tipi istenilen başka bir tipteki numara ile değiştirilebilir.

Numaralanan liste öğesi `\label` (*bkz.* 1.10, 21.sayfa) komutuyla işaretlenip, metinde bu öğeye `\ref` komutuyla bir gönderme yapılabilir. Bu durumda, `\ref` göndermesi `\label` komutuyla işaretlenen öğenin tam numarasını yazdırır. Bunu bir örnekle gösterelim:

```
\begin{enumerate}
 \item birinci düzey
   \begin{enumerate}
    \item 2.düzey\label{lis1}
    \item \dots
      \begin{enumerate}
       \item 3.düzey\label{lis2}
       \item \dots
         \begin{enumerate}
          \item \dots
          \item 4.düzey\label{lis3}
         \end{enumerate}
       \item \dots
      \end{enumerate}
   \item \dots
  \end{enumerate}
 \item \dots \label{lis4}
\end{enumerate}
\ref{lis2} ve \ref{lis4} 'ye göre\dots
\ref{lis3} ve \ref{lis1} 'den\dots
```

1. birinci düzey
   (a) 2.düzey
   (b) ...
      i. 3.düzey
      ii. ...
         A. ...
         B. 4.düzey
      iii. ...
   (c) ...
2. ...

1(b)i ve 2 'ye göre ...
1(b)iiB ve 1a 'den ...

2.5.3 Tanımlama listesi

LaTeX 'de bir tanımlama liste komutlu

`\begin{description}` *items* `\end{description}`

paranteziyle oluşturulmaktadır. Bir tanımlama listesi, genelde, sözlüklü bir makale oluşturmak için kullanılır. Listenin her öğesi `\item` komutuyla başlanmalıdır. Bu komutun zorunlu olmayan *label* argümanı verilmezse LaTeX liste öğesinin önünü boş bırakır. *label* argümanı verildiği taktirde, bu argüman öğe önüne yazdırılır. Aslında, *label* argümanında liste öğesi için bir *konu başlığı* yazılmalıdır. Bu argüman koyu yazıyla (otomatik `\bfseries` çalışır) yazdırılmaktadır. Eğer *label* argümanın kendisi de köşeli bir `[ ]` parantez içinde olması gerekiyorsa, bu argüman `\item` komutunda şöyle yazılmalıdır: `\item[{[`*label*`]}]` . Tanımlama listesine bir örnek verelim:

```
Sözlüklü makale listesi:
\begin{description}
\item[city] (büyük) şehir,(büyük) kent
\item[{[country]}] ülke, memleket,
   yurt, kır, sayfiye,taşra, köy
\item[village] köy, kasaba
\end{description}
```

Sözlüklü makale listesi:

city (büyük) şehir, (büyük) kent

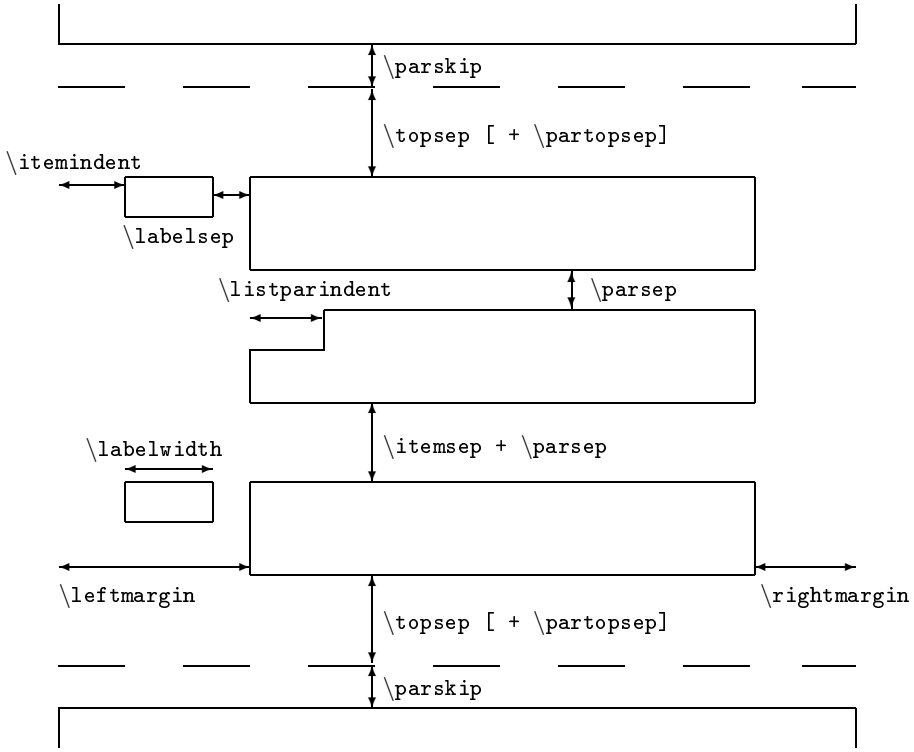
[country] ülke, memleket, yurt, kır, sayfiye, taşra, köy

village köy, kasaba

2.5.4 Özel listeler

Bu paragrafta `itemize`, `enumerate` ve `description` bloklarıyla oluşturulan listelerin parametreleri değiştirilemez olduğu açıklanmaktadır. Daha sonra parametreleri değiştirilebilen bir liste bloğu tanımlanmaktadır. Bu blokta liste parametreleri özel olarak alınabilir. Böyle oluşturulan listeye *özel liste* denir.

İlk önce bir liste parametrelerini tanıtan standart komutları verelim. Bu komutlar liste oluşturan tüm bloklar için aynıdır. Bir listede öğe işareti ve metni, ayrıca onu dış metine göre yerini ayarlayan uzunluk komutları Şekil 2.1 'de verilmektedir:



Şekil 2.1: Bir listede öğe işareti ve metni, ayırca onu dış metine göre yerini ayarlayan komutlar. Modelde birinci öğe iki satırbaşından oluşmaktadır.

Düşey aralıklar: Bir listeden önce ve sonra, `\topsep` komutunun değerine eşit bir düşey aralık koyulmaktadır. Eğer listeden önce bir boş satır bırakılmışsa, veya `\par` komutu varsa, bu taktirde listeden önce ve sonra `\partopsep` komutunun değerine eşit bir düşey aralık koyulmaktadır. Listede bir öğe içinde iki satırbaşı arasına bir ek düşey aralık `\parsep` komutuyla verilir. Normal metinde bu ek düşey aralığı `\parskip` komutuyla verilmesini hatırlatmakta yarar vardır (*bkz.* 2.1, 44.sayfa). Bir liste iki öğe arasına `\parsep` komutunun değerine ek olarak `\itemsep` komutunun değeri de bir düşey aralık olarak verilmektedir. Standart sınıflarda `\itemsep` komutunun değeri sıfırdan farklıdır. Dolayısıyla, bir liste öğeleri arasındaki düşey aralık bir öğe içindeki satırbaşlar arasındaki düşey aralığına göre büyüktür.
Düşey aralığı tanımlayan tüm komutlar esnektir.

Yatay aralıklar: Şekil 2.1 'deki `\leftmargin` ve `\rightmargin` komutları bir liste öğesinin dış metinin sırasıyla sol ve sağ sınırlarından uzaklığını tanımlamaktadır. Liste öğesinin bir satırbaşına (birinci satırbaşı hariç) `\listparindent` komutunun değerine eşit bir satırbaşı boşluğu verilir, fakat standart sınıflarda bu komutun değeri sıfırdır.

`\labelsep` komutu öğe işaretinin sağ sınırıyla metinin sol sınırı arasındaki uzaklığı, `\labelwidth` komutu ise öğe işaretine ayırılan *kutunun* genişliğini tanımlamaktadır. Kutu içinde işaret sola yaslanır. Kutu `\itemindent` komutuyla sol veya sağa çekilebilir. Bu komuta yeni bir değer verilmezse LaTeX bu değeri `0` (sıfır) olarak alır. Standart sınıflarda kutunun sol sınırı dış metinin sol sınırıyla aynı düzey olarak ayarlanmaktadır. Eğer bir öğe işareti kutu enine göre genişse, kutu otomatik sağa genişletilir.

LaTeX standart sınıflarda bir listeye *girişten* önce Şekil 2.1 'deki tüm komutların değerini yeniden onaylamaktadır. Dolayısıyla, bu değerler bir listeden önce değiştirilmesi liste içinde güçünü kayıp etmektedir. Sonuçta, `itemize`, `enumerate` ve `description` bloklarında oluşturulan bir liste parametreleri değiştirilemez.

Parametreleri değiştirilebilen bir listeye *özel liste* denir. Bu liste LaTeX 'in komutlu

```
\begin{list}{def-label}{settings}
  items
\end{list}
```

paranteziyle oluşturulmaktadır. Komutun *def-label* argümanı işaretin görüntüsünü tanımlamaktadır, *settings* argümanı ise liste parametrelerini değiştiren tanıtımları (`\setlength`, `\addtolength`, ...) içermektedir. *settings* argümanında `\usecounter{counter}` tanıtımı da kullanılabilir; burada *counter* liste öğe numarasının değişkenidir.

`list` bloğunda *def-label* ve *settings* argümanları zorunlu değildir. *settings* argümanı gösterilmemesi için bu argüman `{ }` olarak boş kaldırılmalıdır. Bu durumda LaTeX standart parametreleri alır. Aslında `itemize`, `enumerate` ve `description` blokları `liste` bloğunun bir özel durumudur.

`liste` bloğunda hazırlanan listeye bir örnek verelim

```
\newcounter{No}
\begin{list}{kitap:\Alph{No}-
 {\bf \arabic{No}}.\roman{No})}
 {\setlength{\labelsep}{1.5cm}
 \usecounter{No}}
\item liste 1. öğesi
\item liste 2. öğesi
\end{list}
```

... burada dış metin

kitap:A-**1**.i) 

liste 1. öğesi

kitap:B-**2**.ii) 

liste 2. öğesi

dış metin ...

Görüldüğü gibi, bir liste parametreleri ve öğe işareti istenilen şeklinde keyfi oluşturulabilir.

Tüm listelerde (`enumerate` bloğu hariç) `\rightmargin` komutuyla tanımlanan sağ boşluğun eni sıfırdır.

2.5.5 Sade liste

LaTeX 'de bir sade liste komutlu

```
\begin{trivlist}   items   \end{trivlist}
```

paranteziyle oluşturulmaktadır. LaTeX `trivlist` bloğunda sol ve sağ boşluk, işarete ayırtılan kutu eni ve bu kutunun sayfa sol sınırından uzaklığı gibi uzunluk boyutlarını sıfır olarak almaktadır. Bundan başka, `\parsep` komutunun değeri de `\parskip` komutunun değerine eşitlenmektedir. Lis-

tenin diğer parametreleri standart olup, bu parametrelerin listeden önce değiştirilmesi liste içinde geçerli olmaktadır.

2.6 Değişkenler

LaTeX metinde sayfa, bölüm, tablo, şekil, denklem, ... gibi nesneleri otomatik numaralandırmaktadır. Bunun için, numaralandırılacak her nesneye bir değişken tanımlanmıştır. Bir değişkenin adı, genelde, nesnenin adıyla aynıdır. Örneğin, `\section` komutunun değişkeni `section` olup, bu değişken tüm bölüm numaralarını ayarlamaktadır; `table` bloğunun değişkeni, `table` ile, bir sayfa numarasının değişkeni, `page` ile, bir liste öğesinin numarası, her düzeydeki değişken sırasıyla `enumi`, `enumii`, `enumiii` ve `enumiv` ile ve `minipage` bloğunun altbilgi numarasının değişkeni ise `mpfootnote` ile gösterilmektedir. Bir değişken başka bir değişkene içten tanımlanmış olabilir. Bu durumda, eğer dış değişken değeri değişse, iç değişkenin değeri otomatik sıfırlanır. Örneğin, `subsection` değişkeni `section` değişkenine içten tanımlanmıştır. Dolayısıyla, her yeni bölüm başladığında bölümün paragraf ve alt paragraf numarası otomatik sıfırlandırılıp yeniden 1 numarası ile başlanmaktadır.

Metinde bir değişkenin değeri yazdırılması için `\value{`*counter*`}` komutu kullanılır. Komutun zorunlu *counter* argümanına gerekli değişken adı yazılır. `\value` komutu *counter* değişkeninin geçerli değerini yazdırır. Bu komut, genelde, `\setcounter` ve `\addtocounter` komutlarının (*bkz.* 61.sayfa) *integer* argümanında kullanılır. Bunu bir örnekle gösterelim:

```
\setcounter{page}{\value{section}}
```

tanıtımıyla `page` değişkenine `section` değişkeninin değeri verilmektedir, yani geçerli sayfa geçerli bölüm numarasıyla yazdırılır.

Bir değişkenin değerinin yazdırılması için `\the`*counter* komutu da kullanılabilir. `\the` komutu *counter* değişkeninin geçerli değerini yazdırmaktadır. Örneğin, `\thefootnote` komutu kullanılan son alt bilgi numarasını yazdırmaktadır.

Bir değişkende numara biçimi de değiştirilebilir. Bu biçim aşağıdaki komutlarla değiştirilebilir (*bkz.* 1.13.paragraf, 34.sayfa)

`\arabic{`*counter*`}`	`\roman{`*counter*`}`	`\alph{`*counter*`}`
`\fnsymbol{`*counter*`}`	`\Roman{`*counter*`}`	`\Alph{`*counter*`}`

Bunu bir örnekle gösterelim

```
\renewcommand{\theequation}{\thesection,\alph{equation}}
```

tanıtımından sonra bir formül numarası

(\thesection,\alph{equation})

şeklinde yazdırılmaktadır. Örneğin, ikinci bölümün ilk formülü $(2, a)$ şeklinde numaralandırılmaktadır.

Bir metinde, özel olarak, yeni bir değişken tanımlanabilir. Yeni değişken aşağıdaki komutla tanımlanmalıdır

`\newcounter{`*counter*`}[`*out-counter*`]`

Komutun zorunlu *counter* argümanına yeni değişkenin adı yazılmalıdır. Üstelik, bu değişken diğer bir değişkene içten tanımlı olarak oluşturulabilir. Bu durumda, `\newcounter` komutun zorunlu olmayan *out-counter* argümanına dış değişkenin adı yazılmalıdır. Fakat, dış değişken LaTeX 'de önceden varolmalıdır.

Standart sınıflarda yeni değişkenin değeri otomatik `\arabic` komutuyla ayarlanmaktadır, yani değişkenin değerine Arapça numaralar verilmektedir. Bir değişkenin değeri, gerekirse, artırılabilir, azalttırılabilir veya tamamen değiştirilebilir:

`\setcounter{`*counter*`}{`*integer*`}`
`\addtocounter{`*counter*`}{`*integer*`}`

`\setcounter` komutu *counter* değişkenine *integer* değeri vermektedir, `\addtocounter` komutu ise, *counter* değişkenine *integer* değeri eklemektedir. Dolayısıyla, birinci komut bir değişken değerini tamamen değiştirmektedir, ikinci komut ise bir değişken değerini artırır veya azalttırır. Açıkça, *integer* argümanı eksi ise, `\addtocounter` komutu değişkenin değerini azalttırır.

Birkaç düzeyli değişken değeri aşağıdaki komutlarla, özel olarak, değiştirilebilir.

```
\stepcounter{counter}
\refstepcounter{counter}
```

Bu komutlar *counter* adlı değişkenin değerini bir birim arttırıp, bu değişkene içten tanımlı tüm değişkenlerin değerini sıfırlamaktadır. Bunu bir örnekle gösterelim

```
\stepcounter{section}
```

tanıtımı bölüm numarasını bir birim arttırıp, bölümdeki paragraf ve alt paragraf numarasını sıfırlandırmaktadır.

`\refstepcounter` komutu kendisini geçerli değişken olarak ilan etmektedir, yani metinde `\ref` komutuyla bu değişkenin değeri yazdırılabilir.

2.7 Özel paragraflar

2.7.1 Satırda metin yerini belirtmek

`\centering` komutu veya komutlu

```
\begin{center} ... \end{center}
```

parantez bloğundaki tüm satırlar ortalaya yaslar.

`\raggedright` komutu veya komutlu

```
\begin{flushleft} ... \end{flushleft}
```

parantez bloğundaki tüm satırlar sola yaslar.

`\raggedleft` komutu veya komutlu

```
\begin{flushright} ... \end{flushright}
```

parantez bloğundaki tüm satırlar sağa yaslar.

`center`, `flushleft` ve `flushright` bloklarında bir satır sonuna sığmayan bir sözcük heceleme yapılmadan yeni satıra alınır ve bir satır, normal metinde olduğu gibi, `\\` komutuyla kesilmelidir. Bu bloklardan hemen

sonra bir boş satır veya `\par` komutu varsa, bloktan sonraki ilk satır standart düşey aralıkla başlanır.

2.7.2 Ayırtılan metin

Bazen bir metinin (örneğin: *şiir*, *özet*, ...) ayırtılan şeklinde yazdırılması gerekmektedir. Komutlu

```
\begin{quote}      ... \end{quote}
\begin{quotation}  ... \end{quotation}
```

parantez bloğundaki tüm metin sol ve sağdan belli bir ek boşluğuyla ayırtılan şeklinde yazdırılmaktadır.

Aslında, bu iki blok `list` bloğunun tek `\item[ ]` (bkz. 2.5.4, 57.sayfa) öğeli şekli gibi tanımlanmıştır. Dolayısıyla, bloktan önce ve sonra `\topsep` komutunun değerine eşit bir düşey aralık koyulur. Eğer bloktan sonra bir boş satır veya `\par` komutu varsa, `\topsep` komutun değerine `\partopsep` komutunun değeri de eklenir. `quotation` bloğunda bir paragraf başına standart bir satırbaşı boşluğu koyulmaktadır. Bu boşluk `quote` bloğunda yoktur. Dolayısıyla, bu blokta paragraflar arası ek düşey aralık vardır.

2.7.3 Matbaa harfli metin

Aşağıdaki komutlu parantez içine alınan bir metin yazı makinesi harfiyle yazdırılır, yani LaTeX bu metini okumadan dosyaya nasıl yazılmışsa, o şekilde yazdırır.

```
\begin{verbatim}    ... \end{verbatim}
\begin{verbatim*}   ... \end{verbatim*}
```

Yıldızlı `verbatim*` bloğunda yatay boşluklar ␣ ile yazdırılır. Bloğun kapatma `\end{verbatim}` komutunda `\end` ve `{verbatim}` arasında bir boşluk olmaması gerekmektedir. `verbatim` bloğunun daha iyi çalışması için belgenin başlık kısmında `tools` sınıfının `verbatim` paketi tanıtılmalıdır. Bu paket bloğun bazı özelliklerini yeniden tanımlamaktadır, özel halde, o `\end` ve `{verbatim}` arasında boşluklara izin verir; fakat bu satırda kapatmadan sonradaki her şey iptal edilir.

Aslında `verbatim` bloğu, `quote` ve `quotation` bloğu gibi, `list` bloğunun tek `\item[ ]` (bkz. 2.5.4, 57.sayfa) öğeli şekli olarak tanımlanmıştır. Dolayısıyla, `verbatim` bloğunda önce ve sonra `\topsep` komutun değerine eşit bir düşey aralık koyulmaktadır. Eğer bu bloktan sonra bir boş satır veya `\par` komutu varsa, `\topsep` komutunun değerine `\partopsep` komutunun değeri de eklenir. Bu komutların değeri `\setlength` veya `\addtolength` komutuyla değiştirilebilir.

`verbatim` ve `verbatim*` blokları büyük hacimli veya birkaç satırlı metin için kullanılır. Tek satırlı bir metin

`\verb|`*text*`|`	`\verb*|`*text*`|`

komutlarıyla bir yazı makinesi harfiyle yazdırılır. Bunun için, bu metin komutun zorunlu olmayan *text* argümanına yazılmalıdır. Komutun `|` simgesi *text* argümanın içinde olmayan herhangi bir harf veya simge olabilir. Tanımdan görüldüğü gibi, *text* argümanı iki `|` simgesi arasına alınmalıdır. Bu simge yerine, genelde, `|`, `/`, `"`, `!`, ... tipli simgeler alınır. Yıldızlı `\verb*` komutu boşlukları ␣ ile yazdırır.

NOT: `\verb` ve `verbatim` blokları bir komutun argümanında kullanılamaz.

`\verb` komutun `|` simgesi `*` olamaz.

2.7.4 shortvrb paketi

Bir metinde `\verb` komutu çok sayıda kullanılacaksa, kolaylık için, bu komut sadece bir simge veya harfe dönüştürülebilir. Bunun için belgenin başlık kısmında shortvrb paketi tanıtılmalıdır:

```
\usepackage{shortvrb}
```

Bu paket

```
\MakeShortVerb{\*}
```

tanıtımıyla herhangi bir simge veya harfi `\verb` komutunun tüm özelliğine sahip olan *özgü* bir paranteze dönüştürmektedir. Bu simge veya harf `\MakeShortVerb` komutunun zorunlu `*` argümanına yazılmalıdır. Örneğin,

```
\MakeShortVerb{\+}
```

tanıtımından sonra metinde `\verb|`*text*`|` komutu yerine kısaca `+`*text*`+` olarak yazılabilir. shortvrb paketinde paralel olarak `\verb` komutunun kendisi de çalışmaktadır. `\MakeShortVerb` komutu belgenin başlık kısmında veya içinde istenilen sayıda kullanılabilir. Fakat, bu komutla özgü paranteze dönüşen bir simge veya harf metinde başka amaçla kullanılamaz. Böyle bir simge veya harf başka amaçla kullanılması için onun *özgü özelliği* iptal edilmelidir.

shortvrb paketinin

```
\DeleteShortVerb{\*}
```

tanıtımı `*` argümanın `\MakeShortVerb` komutuyla verilen *özgü özelliğini* iptal etmektedir. Bu durumda `*` argümanı (simge veya harf) eski haline dönmektedir.

2.7.5 alltt paketi

LaTeX 'in alltt paketi de verbatim tipli bir bloğu tanımlamaktadır. Paketin komutlu

```
\begin{alltt}  ...  \end{alltt}
```

parantezi aynen `verbatim` bloğu gibi çalışmaktadır (*bkz.* 2.7.3, 63.sayfa). Fakat, alltt bloğunun verbatim bloğuna göre daha iyi bir özelliği vardır. `alltt` bloğu LaTeX 'in yönlendirici `\`, `{` ve `}` simgelerine eskice çalışmasına izin vermektedir, yani bu simgeler `alltt` bloğunda da yönlendirici olarak çalışır. Oysa, bu blokta `\textrm{`*text*`}` gibi komutlar veya font değiştirme ve `\(`*math*`\)` şeklinde bir matematik formül kullanmasına izin verir. `alltt` bloğunda matematik üst ve alt indisler sırasıyla `\sp` ve `\sb` komutlarıyla ve dolar `$` simgesi ise, `\(` ve/veya `\)` komutuyla yazılması gerekmektedir. Çünkü, bu indisleri yazdıran `^` ve `_` simgeler, hemde dolar `$` simgesi (*bkz.* 3.1.paragraf, 109.sayfa) bu blokta yönlendiriciliğini kayıp etmektedir.

2.8 Kutu'lar

2.8.1 Satır kutuları

Bir metin veya formül parçasını ayırtılan şeklinde ayarlanması bu metini *kutu'ya* alınması denir. LaTeX birkaç komutlarla çeşitli kutular oluşturmaktadır. LaTeX 'in

`\makebox[`*width*`][`*position*`]{`*text*`}`

komutu *text* argümanına yazılan metini eni *width* olan bir kutu içine ayırtılan metin şeklinde yazdırır. Ayırtılan metin kutu içinde komutun zorunlu olmayan *position* argümanın `l`, `r` ve `c` seçeneğiyle sırasıyla sola, sağa ve ortaya yaslanır. Bu argümanın yine bir `s` seçeneği de vardır. Bu seçenek ayırtılan metini tüm kutu içine uzatır. Tabii ki bir metin böyle uzatılabilmesi için bu metnin uzatılabilen yatay uzunlukları (örneğin: ␣ , `\hspace`, ...) içermesi gerekmektedir.

`\makebox` komutunun *position* ve *width* argümanları zorunlu değildir. Bu argümanlar verilmezse LaTeX ayırtılan metini kutu içinde ortalar (yani `c` seçeneği alır) ve *width* argümanı metini kendisine göre ayarlar.

Üstelik *width* argümanda

`\width` `\height` `\depth` `\totalheight`

uzunluk komutları da kullanılabilir. Bu komutlar *text* argümanın sırasıyla eni, yüksekliği, derinliği ve toplam yüksekliğine eşittir.

Ayırtılan bir metin çerçeveli kutu şeklinde de yazdırılabilir. LaTeX 'in

`\framebox[`*width*`][`*position*`]{`*text*`}`

komutu `\makebox` komutu gibi olup, ayırtılan metini bir çerçeve içine almaktadır, yani çerçeveli bir kutu oluşturmaktadır. Çerçeve çizgisin kalınlığı ve çizgiyle metin arasındaki uzaklık sırasıyla

`\fboxrule` ve `\fboxsep`

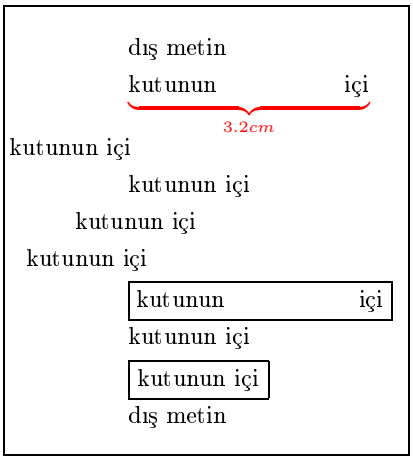
komutlarında saklanmaktadır. Bu komutlara yeni bir değer verilmezse LaTeX bu değerleri sırasıyla `0.4pt` ve `3pt` olarak alır.

`\makebox` ve `\framebox` komutları zorunlu olmayan argümansız kısaca sırasıyla

\mbox{*text*}	\fbox{*text*}

komutlarıyla yazılabilir. Çerçeveli ve çerçevesiz kutuları birkaç örnekle gösterelim

```
dış metin
\makebox[3.2cm][s]{kutunun içi}
\makebox[\depth][r]{kutunun içi}
\makebox[\width][l]{kutunun içi}
\makebox[\height][c]{kutunun içi}
\makebox[\totalheight][r]{kutunun içi}
\framebox[3.5cm][s]{kutunun içi}
\mbox{kutunun içi}
\fbox{kutunun içi}
dış metin
```



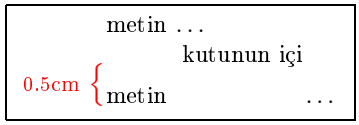
2.8.2 Kutunun düşey kaydırılması

Bir kutu

\raisebox{*offset*}[h][d]{*text*}

komutuyla düşey kaydırılmaktadır. `\raisebox` komutunun *text* argümanı satır taban çizgisinden *offset* uzaklığına düşey kaydırılmaktadır. Komutun `h` ve `d` argümanları metinin kutudaki sırasıyla uydurma yüksekliği ve derinliğini tanımlar. Aslında, TEX kutunun içinde ayırtılan metine bu değerlere göre bir yer ayarlar. `\raisebox` komutunu bir örnekle gösterelim

```
metin ...
metin \raisebox{0.5cm}{kutunun içi} ...
```



Örnekten görüldüğü gibi, *offset* argümanın artı değerinde kutu yukarı yönde, eksi değerinde ise aşağı yönde kaydırılmaktadır.

2.8.3 Kutunun önceden hazırlanması

Eğer metinde çok sayıda kutu varsa, bu kutular metinin bir yerinde veya yardımcı bir dosyada hazırlanıp, sonra onlar metinin gerekli yerinde "çağrılabilir". Bunun için her kutuya bir isim verilmesi gerekmektedir. LaTeX 'in

`\newsavebox{`*cmd*`}`

komutu *cmd* adlı bir kutuyu tanımlamaktadır. Kutunun kendisi ise

`\savebox{`*cmd*`}[`*width*`][`*position*`]{`*text*`}`

`\sbox{`*cmd*`}{`*text*`}`

`\begin{lrbox}{`*cmd*`}` *text* `\end{lrbox}`

komutların herhangi biriyle oluşturulabilir. Bu komutlar hazırlanan bir kutuyu *cmd* adıyla belleğe alır. `\savebox` komutun argümanları `\makebox` komutunun argümanları gibidir. `lrbox` bloğunda *text* argümandan önceki ve sonraki boşluklar iptal edilir. *text* argümanda `\verb` ve `verbatim` bloğu kullanılabilir.

Metinde *cmd* adlı bir kutu

`\usebox{`*cmd*`}`

komutuyla çağrılır.

NOT: Kutunun *cmd* adı `\` komut işaretiyle başlanması gerekmektedir.
İki `\newsavebox` komutu arasında bir boş satır varolmalıdır.
Eğer kutu yardımcı bir dosyada hazırlanmışsa, bu kutu `\usebox` komutuyla çağrılışdan önce yardımcı dosyanın kendisi `\input` komutuyla çağrılmalıdır (eğer o önceden çağrılmamışsa).

Önceden hazırlanan kutuyu birkaç örneklerle gösterelim.

```
\newsavebox{\mm}
\newsavebox{\xx}
\newsavebox{\yy}
\newsavebox{\zz}
\savebox{\mm}[2.0cm][r]{mm adlı kutu}
\sbox{\xx}{ xx adlı kutu}
\begin{lrbox}{\yy} yy adlı kutu\end{lrbox}
\begin{lrbox}{\zz}\fbox{zz adlı kutu}
\end{lrbox}
metin ...
metin \usebox{\mm} ...
metin \usebox{\xx} ...
... \usebox{\zz} ...
... \usebox{\yy} metin ...
```

metin ...

metin **mm** adlı kutu ...

xx adlı kutu ...

... **zz** adlı kutu ...

... **yy** adlı kutu metin ...

Örnekten görüldüğü gibi, `lrbox` bloğunda yine kutu oluşturan bir komut kullanılabilir.

2.8.4 Metin kutusu

2.8.1-2.8.3 paragraflarında ele alınan kutu bloklarıyla birkaç satırlı bir kutu oluşturulamaz, aynı zamanda kutuya alınan metin de istenilen yönde kaydırılamaz. Dolayısıyla, bu blokların kullanma dairesi dardır. Kullanma dairesi daha da geniş olan kutu tipli bir blok LaTeX 'in

`\parbox[`*align*`][`*height*`][`*inner-align*`]{`*width*`}{`*text*`}`

komutuyla oluşturulabilir. `\parbox` komutu eni *width* olan bir alt paragraf oluşturup, onun içine *text* argümanı yazdırır. Komutun zorunlu olmayan *align* argümanında kutunun satırdaki yeri argümanın aşağıdaki seçenekleriyle gösterilmektedir

`t` - kutunun üst taban satır çizgisi (yani üst sınırı) geçerli satırın taban çizgisi düzeyinde;

`b` - kutunun alt taban satır çizgisi (yani alt sınırı) geçerli satırın taban çizgisi düzeyinde;

`c` - kutunun ortası geçerli satırın ortasına yaslanır.

align argümanı gösterilmediği taktirde LaTeX bu seçeneği `c` olarak alır.

Komutun *height* argümanı da zorunlu değildir. Eğer bu argüman verilmişse, *text* argümanın kutu içindeki yeri komutun *inner-align* argümanın aşağıdaki seçenekleriyle gösterilmektedir

`t` - *text* argümanı kutunun üstune yaslanır;

`b` - *text* argümanı kutunun altına yaslanır;

`c` - *text* argümanı kutunun ortasına yaslanır.

`s` - *text* argümanı kutunun tüm yüksekliğine uzatılır.

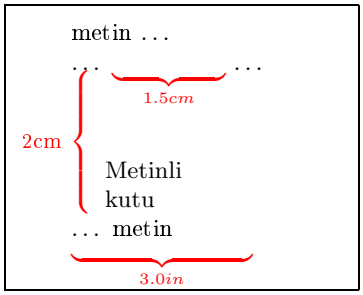
inner-align argümanı gösterilmediği taktirde LaTeX bu seçeneği `c` olarak alır.

Bir şekili veya tabloyu içeren büyük metinli bir kutu LaTeX 'in komutlu

```
\begin{minipage}[ align][height][inner-align]{width}
  text
\end{minipage}
```

paranteziyle oluşturulmaktadır. `minipage` bloğunun argümanları `\parbox` komutunun argümanları gibidir. Aşağıdaki örnekte `\parbox` ve `minipage` blokları ele alınmaktadır

```
\begin{minipage}{3.0in}
metin ...
... \parbox[t][2.0cm][b]{1.5cm}{
Metinli kutu} ...
... metin
\end{minipage}
```



2.8.5 Dolu kutular

Metinde bir dolu kutu, yani içi sadece siyaha boyalı bir kutu LaTeX 'in

```
\rule[offset]{width}{height}
```

komutuyla oluşturulur. `\rule` komutu geçerli satırdan *offset* uzaklıkta eni *width* ve yüksekliği *height* olan bir siyah dikdörtgen kutuyu yazdırır:

```
... metin
metin \rule[3mm]{0.2cm}{0.4cm} ...
```

... metin
metin ...

2.9 Sekmeler

Düşey hizalama yapmak için genelde klavyedeki tab (sekme) tuşu kullanırlır. LaTeX 'de ise, yazıları düşey yönde hizalama yapmak için komutlu

```
\begin{tabbing}  ...  \end{tabbing}
```

parantezi kullanılır. `tabbing` bloğunun temel düşüncesi aşağıdaki örnekte açıkça görülmektedir

```
\begin{tabbing}
123 \= 456 \= 789 \= \kill
\>     Bir \\
\>     Iki \\
\>\>   Üç \\
\>\>   Dört \\
\>\>\> Beş \\
\>\>\> Altı \\
\>\>   Yedi \\
\>\>   Sekiz \\
\>     Dokuz \\
\>     On
\end{tabbing}
```

Bir
İki
Üç
Dört
Beş
Altı
Yedi
Sekiz
Dokuz
On

Örnekte `tabbing` bloğunun birinci satırı sekme duraklarının durumlarını ayarlamaktadır;

`\kill` komutu LaTeX'e bu satırın görüntüsünü yazdırmamasını bildirmektedir;

`\=` komutu sekme duraklarının yerlerinin belirlemektedir;

`\>` komutu metini bir sonraki sekme durağına konumlandırır [2].

[2] Plain TeX 'de \> komutu sadece matematik bir blokta kullanılmalıdır ve bu komut bir karakterlik boşluk bırakır. Bu boşluk, LaTeX 'in matematik bloğunda \: komutuyla yazılır (*bkz.* tablo 3.29).

`tabbing` bloğunda LaTeX, *next-tab-stop* ve *left-margin-tab* adlı iki değişkeni desteklemektedir. Sekme durakları öyle bir gizli kılavuz çizgileri oluşturuyorki, metin, bu çizgilere `\=`, `\>`, ... komutlarıyla yaslandırılır. Kılavuz çizgileri soldan sağa doğru 0,1,2, ... rakamlarıyla numaralandırılmaktadır. Yukarıdaki örnekte 0,1 ve 2 numaralı sekme durakları ele alınmıştır. Başlangıçta, *left-margin-tab* değişkenin değeri sıfıra eşit olup, metinin yaslanması geçerli sol kenar boşluğuna göre ayarlanır ve *next-tab-stop* değişkeninin değeri 1 olarak alınır. Eğer *next-tab-stop* değişkeninin değeri i 'ye eşitse, `\>` komutu metini i. sekme duraklarına yaslayacak ve *next-tab-stop* değişkeninin değerini 1 'e artırır. `\\` komutu geçerli satırı durdurup, yeni bir satırı başlatır ve *next-tab-stop* değişkeninin değerini 1+*left-margin-tab* 'e eşitler, fakat *left-margin-tab* değişkeninin değerini değiştirmez.

`tabbing` bloğunda `\=`, `\>` ve `\kill` komutlarının haricinde aşağıdaki komutlar da kullanılabilir:

`\-`	*left-margin-tab* değişkeninin değerini 1 azaltır;
`\+`	*left-margin-tab* değişkeninin değerini 1 artırır;
`\<`	öndeki bir `\+` komutunu iptal eder;
`\'`	metini sola yanaştırır;
`` \` ``	komutu ve `\\` (veya `\end{tabbing}`) komutu arasındaki metin sağa yanaştırılır ve bu aralıkta `\>` ve/veya `\'` komutları olmaması gerekmektedir;
`\pushtabs`	sekme duraklarını belleğe alır. Gerekli yerlerde `\poptabs` komutuyla sekme durakları geri yerine konulur;
`\poptabs`	son kullanılan `\pushtabs` komutuyla belleğe alınan sekme duraklarını tekrar yerine koyar. `\pushtabs` ve `\poptabs` komutları `tabbing` bloğunda sekme duraklarını sadece bazı satırlarında değiştirmek için kullanılabilir;
`\a`	`\=`, `\'` ve `` \` `` komutlarının normal metindeki görevini, `tabbing` bloğunda yerine getirir. Örneğin, `\=` komutu normal metinde kendisinden sonra yazılan ilk harfin veya simgenin üzerine bir üst çizgi çizer, `tabbing` bloğunda ise bir harfe veya simgeye bir üst çizgi çizdirmek için `\a=` komutu kullanılır: `\a=o` ō. Benzer şekilde, `\'` komutu normal metinde `\'o` ó olarak çalışır, bu komutun `tabbing` bloğunda da bu şekilde çalışması için `\a'` komutu kullanılır.

PASCAL 'daki basit bir program `tabbing` bloğunda aşağıdaki gibi yazıla-

bilir:

```
\begin{tabbing}
function \= fact(n:integer):
integer;\\
begin \> \= \+ \\
\> if \= n $>$ 1 then \+ \\
fact := n * fact(n-1) \- \\
else \+ \\
fact := 1; \-\- \\
end;
\end{tabbing}
```

```
function    fact (n : integer) : integer;
begin
            if n > 1    then
                        fact := n * fact(n-1)
            else
                        fact := 1;
end;
```

Yukarıdaki tüm komutları içeren `tabbing` bloğuna bir örnek verelim.

```
\begin{tabbing}
1234567\=              \kill
23     \> 24 \= 25  \+\+ \\
                156      \\
          342 \' 125 \-   \\
          234 \' 126      \\
                456,765\-\\
123432 \> 21             \\
       \> 37 \` 55       \\
44           \` 33       \\
       \> 22 \` 11  \>99
\end{tabbing}
```

```
23       24 25
            156
         342  125
    234  126
         456, 765
123432   21
         37                    55
44                             33
         22 11                 99
```

NOT: Bir `tabbing` bloğu geçerli sayfaya sığmazsa, LaTeX onun sığmayan parçasını yeni sayfayadan devam ettirir.

Bir `tabbing` bloğunda sekme durakları `\=` komutuyla istenilen şekilde değiştirilebilir. Bunu aşağıda çeşitli örneklerle gösterelim:

```
\begin{tabbing}
Bir küçük \= tablo \\
Bu tabbing bloğunda\> / / / / / / /
\end{tabbing}
```

Bir küçük tablo
Bu tabbing / / / / / / /

```
\begin{tabbing}
eski sütun 1\=eski sütun 2\=eski sütun 3\\
sütun 1     \> sütun 2                  \\
yeni sü.1\= yeni sü. 2    \>eski sütun 3\\
sütun 1  \> sütun 2       \> sütun 3
\end{tabbing}
```

eski sütun 1	eski sütun 2	eski sütun 3
sütun 1	sütun 2	
yeni sü.1	yeni sü. 2	eski sütun 3
sütun 1	sütun 2	sütun 3

2.10 Tablolar

2.9. paragrafında yazıları düşey yönde hizalı tablo benzeri bir yapı oluşturan `tabbing` bloğu ele anlındı. Bu paragrafta normal tablolar ele alınacaktır.

2.10.1 `tabular` bloğu

Metindeki bir paragrafı satır ve sütunlardan oluşan bir tablo şeklinde göstermek için LaTeX 'in komutlu

```
\begin{tabular}[align]{keys}
  strings
\end{tabular}
```

parantezi kullanılır. LaTeX `tabular` bloğunu, bir büyük kutu olarak oluşturur. Dolayısıyla, bu bloğun bir sayfaya sığması gerekmektedir. Aslında, `tabular` bloğu bir `table` bloğuna alınmalıdır (*bkz.* 2.11.1, 84.sayfa). Çünkü, bu durumda, `tabular` bloğu numaralandırılan bir tablo olarak kayan bir nesneye dönüşür.

`tabular` bloğunun *align* argümanı tablonun geçerli sayfadaki düşey yerini tanımlar. Bu argümanın aşağıdaki seçenekleri vardır:

`t` - tablonun ilk satırı geçerli satır düzeyinde hizalanır;

`b` - tablonun son satırı geçerli satır düzeyinde hizalanır;

`c` - tablo geçerli satırda düşey yönde ortalanır.

align argümanı gösterilmediği taktirde LaTeX bu seçeneği `c` olarak alır.

Tablonun ilk ve/veya son satırı bir çizgiden oluşabilir (bu çizgi `\hline` ile yazdırılır). Bu durumda, *align* argümanın `t` ve/veya `b` seçeneğinde bu çizgi geçerli satırın taban çizgisi düzeyinde yazdırılır.

Komutun zorunlu *keys* argümanında tablodaki sütunlar sayısı aşağıdaki, *anahtar* denilen, seçeneklerle gösterilmelidir:

anahtar	*görevi*
`l` -	sütundaki metin sola yaslanır;
`r` -	sütundaki metin sağa yaslanır;
`c` -	sütundaki metin ortalanır;
`p`{*width*} -	eni *width* olan bir sütun oluşturulur.

Bir anahtar bir sütun oluşturur. Bir sütunun eni sütundaki en geniş metine göre ayarlanır. LaTeX bir sütunun soluna ve sağına `\tabcolsep` komutun değerine eşit olan bir yatay boşluk bırakır. Bu komuta yeni bir değer verilmezse LaTeX bu değeri `6pt` olarak alır.

keys argümanın `l`, `r`, `c` ve `p{`*width*`}` anahtarları dışında, önemli olan, iki: `|` ve `@{`*text*`}` anahtarları daha vardır. `|` anahtarı tablonun tüm yüksekliğinde iki sütunu kendi arasında ikiye ayıran düşey bir çizgi çizer; `@{`*text*`}` anahtarı ise iki sütun arasındaki boşluğu iptal ederek, bu boşluğun yerine *text* argümanını yazdırır. Bu anahtarın *text* argümanı `@{ }` olarak boş bırakılabilir. Bu tür argümanla ilk sütundan önceki ve/veya son sütundan sonradaki boşluklar iptal edilebilir. `|` ve `@{..}` anahtarı kendisinden sonra yazılan sütuna aittir (ilk sütun hariç). `@` anahtarının *text* argümanında

`\extracolsep{`*length*`}`

komutu kullanılabilir.

Bu durumda, `@` anahtarının `@{\extracolsep{`*length*`}}` seçeneğinden sonraki tüm sütunların (ilk sütun hariç) soluna *length* uzunluklu bir boşluk koyulmaktadır.

Bir tablonun bazı sütunları aynı tipli ise, bu sütunları *tabular* bloğunun *keys* argümanında tek bir `*{`*n*`}{`*keys*`}` ifadesiyle gösterilebilir, burada *n* aynı tipli sütunların sayısı, *keys* ise `l`, `r`, `c` ve `p{`*width*`}` seçeneklerden biridir.

tabular bloğunda bir satırda iki sütun kendi arasında `&` simgesiyle ayırılır; bir satırın sonunda ise `\\`, `\\[`*length*`]` ve `\tabularnewline[`*length*`]` komutlarından biri yazılmalıdır, burada *length* argümanı satırın ek düşey yükseliğini gösterir (*bkz.* 2.3.5, 50.sayfa). Tablonun son satırında `\\` ve `\tabularnewline` komutları kullanılmayabilir. *tabular* bloğuna bir örnek verelim.

Giriş dosyası:

```
\begin{tabular}{l|@{ sayfa }l||@{\extracolsep{1cm}}rc|*{2}{p{5mm}|}}
 1.sütun   & 23  & 3.sütun & 4.sütun &    &      \\
 AAA       & 37  &         & CCC     & DD & DD   \\
           & 200 & BB      &         & D  &      \\[2mm]
 A         & 184 & B       & C       & DD & D
\end{tabular}
```

LaTeX 'de görüntüsü:

1.sütun	sayfa 23	3.sütun	4.sütun		
		$\underbrace{\quad}_{1cm}$	$\underbrace{\quad}_{1cm}$	$\underbrace{\quad}_{5mm}$	$\underbrace{\quad}_{5mm}$
AAA	sayfa 37		CCC	DD	DD
	sayfa 200	BB		D	
A	sayfa 184	B	C	DD	D

Örnekten görüldüğü gibi, bir sütunun bazı satırları boş bırakılabilir ve *keys* argümanında `|` simgesi istenilen sayıda kullanılabilir. Bir `|` simge düşey bir çizgi çizer.

tabular bloğunda bir yatay çizgi ise

`\hline` `\cline{`$i-j$`}`

komutlarıyla çizdirilir. Bu komutlar bir `\\` veya `\tabularnewline` komutundan sonra ve bir sonraki satırdan önce yazılmalıdır. `\hline` komutu tüm tablo genişliğinde bir yatay doğru çizdirir, `\cline` komutu ise i. sütun sol sınırından j. sütun sağ sınırına kadar bir yatay doğru çizdirir.

Bir sütunun bir satırında, ayrıca `@` komutunun *text* argümanında da `\vline` komutu kullanılabilir. `\vline` komutu, kullandığı yere, bir düşey doğru çizdirir.

Bir satırda birkaç sütun

`\multicolumn{`n`}{`*keys*`}{`*text*`}`

komutuyla bir sütuna özel olarak birleştirilebilir, burada n ($n \in \mathbb{N}$) birleştirilecek sütunların sayısıdır. Bunu bir örnekle gösterelim:

```
\begin{tabular}{lc|rl}
3  & 12 & 53 & 27\\
87 & \multicolumn{3}{r}{3 birleşen
sütun}\\ \cline{1-1}
91 \vline& 436 & 46 & 902\\
\cline{2-3}
\multicolumn{2}{c}{2 birleşen sütun}&
                    38\vline&89
\end{tabular}
```

3	12	53	27
87	3 birleşen sütun		
91	436	46	902
2 birleşen sütun		38	89

`tabular` bloğunda iki satır arasındaki düşey aralık `\arraystretch`

komutunda saklanmaktadır. Bu komuta yeni bir değer verilmezse LaTeX bu değeri 1 olarak alır [3]. `\hline`, `\cline` ve `\vline` komutlarının çizdirdiği bir doğrunun kalınlığı `\arrayrulewidth` uzunluk komutunda saklanmaktadır.

Standart sınıflarda (`slides` sınıfı hariç) bu komutun değeri `0.4pt` 'dir. Süregelen iki doğru arasındaki aralık ise `\doublerulesep` komutunun değerine eşittir (`slides` sınıfı hariç). Bu değer değiştirilmediği taktirde LaTeX bu değeri `2pt` olarak alır.
Aşağıda `tabular` bloğuna bir örnek verilmektedir

Giriş dosyası:

```
\begin{tabular}{|r||r@{--}l|p{1.25in}|}
\hline
\multicolumn{4}{|c|}{2001-2003 yılların \$\$ tablosu}\\ \hline \hline
& \multicolumn{2}{c|}{Fiyat} & \cline{2-3} \multicolumn{1}{|c||}{Yıl}
& \multicolumn{1}{r@{\, \vline\,}}{aşağı} & yukarı
& \multicolumn{1}{c|}{Açıklama}\\ \hline
2001 & 87 & 234 & kötü yıl \\ \hline
02   & 234 & 234 & iyi yıl \\ \hline
03   & 234 & 2034 & çok iyi yıl \\ \hline
\end{tabular}
```

LaTeX 'de görüntüsü:

2001-2003 yılların $$ tablosu			
Yıl	Fiyat		Açıklama
	aşağı	yukarı	
2001	87–234		kötü yıl
02	234–234		iyi yıl
03	234–2034		çok iyi yıl

2.10.2 array paketi

`tools` sınıfın `array` paketi `tabular` bloğunun imkan dairesini daha da genişletmektedir. `array` paketi *keys* argümanına aşağıdaki yeni anahtarları eklemektedir:

[3] Komutun değeri `\renewcommand` komutuyla değiştirilebilir.

anahtar

`!{`*text*`}`	iki sütun arasına *text* argümanı yazdırır. Bu anahtarın `@{..}` anahtarından farkı şu ki, o iki sütun arasındaki standart yatay boşluğu iptal etmeyecektir.
`m{`*width*`}` `b{`*width*`}`	bir sütunun her satırını eni *width* olan bir kutu şeklinde ayarlar. `m{..}` anahtarında kutu ortası geçerli satırı ortasına yaslanır, `b{..}` anahtarında ise kutu son satır taban çizgisi geçerli satırı taban çizgisi düzeyinde ayarlanır.
`>{`*before*`}` `<{`*after*`}`	bir anahtardan sırasıyla önce ve sonra kullanılır. Şu halde, bir sütunun her satırı *before* argümanı ile başlanıp, *after* argümanı ile bitmektedir. Örneğin, eğer bir anahtardan önce `>{\bfseries}` yazılmışsa, anahtarın sütunundaki tüm metin koyu yazısıyla yazdırılır. Çünkü, bu sütunun her satırı `\bfseries` komutuyla başlanır. Başka bir örnek: `>{$}c<{$}` olarak üretilen sütunun her satırı iki dolar `$` simgesi arasına alınmaktadır. Böyle tipli anahtarlar bir matematik formüllü sütunda kullanılabilir.

`array` paketi tanıtılan metinde *keys* argümanın yeni anahtarların çalışmasını bir örnekle gösterelim:

```
\begin{tabular}{
>{\bfseries}l|l!{..}>{$}c<{$}m{1cm}}
a & a & \int f(x) & 1234 567 89\\
b & b & \sum s(x) &
\end{tabular}
```

				1cm
a	a	..	$\int f(x)$	1234 567 89
b	b	..	$\sum s(x)$	

Metinde kullanılan tüm özel anahtarlar ayrı bir dosyaya toplanabilir. `\showcols` komutu giriş dosyasında kullanılan tüm özel anahtarları `log`-dosyasına yazdırır. Eğer metinde bir özel anahtar çok sayıda kullanılsa, bu anahtara aşağıdaki tanıtımla yeni kısa bir ad verilebilir:

`\newcolumntype{`*name*`}[`*n*`]{`*definition*`}`

`\newcolumntype` komutu *n* argümanlı *definition* anahtarı yeni *name* ad ile tanımlamaktadır [4]. Örneğin, yukarıda ele alınan

[4] Bu komutun argümanı `\newcommand` komutun argümanı gibi tanımlanır.

`>{\bfseries}l` ve `>{$}c<{$}`

özel anahtarları sırasıyla yeni `A` ve `B` adıyla şöyle tanımlanabilir:

`\newcolumntype{A}{>{\bfseries}l}` ve `\newcolumntype{B}{>{$}c<{$}}`

`array` paketi `tabular` bloğunda iki satır arasına `\extrarowheight` komutun değerine eşit olan bir ek düşey aralık koymaktadır. Standart sınıflarda bu komutun değeri sıfırdır.

Eğer tablodan önce `\firsthline` komutuyla bir yatay doğrusu yazdırılmışsa, `t` seçenekli tablonun ilk satır taban çizgisi geçerli satır taban çizgisi düzeyinde ayarlanır. Eğer tablodan sonra `\lasthline` komutuyla bir yatay doğrusu yazdırılmışsa, `b` seçenekli tablonun son satır taban çizgisi geçerli satırı taban çizgisi düzeyinde ayarlar. `\firsthline` ve `\lasthline` komutları tablonun sırasıyla ilk ve son satırlarında bir ek düşey aralık oluşturmak için kullanılabilir.

Bu komutların değeri `\extratabsurround` komutunda saklanmaktadır. `\extratabsurround` komutuna yeni bir değer verilmezse LaTeX bu değeri `2pt` olarak alır.

2.10.3 Verilen genişlikteki tablolar

LaTeX 'de verilen genişlikteki bir tablo `tabular` 'in yıldızlı `tabular*` bloğuyla oluşturulabilir:

```
\begin{tabular*}{width}[align]{keys}
    strings
\end{tabular*}
```

`tabular*` bloğunun *width* argümanında tablonun genişliği yazılmalıdır. Tablonun genişliği sabit olduğundan sütunlar arasındaki boşluk esneklidir. Öte yandan, sütunlardaki metin de tablonun bir doğal ini oluşturmaktadır. Eğer tablonun doğal genişliği verilen *width* genişliğinden büyükse, bir sütundaki metin komşu sütuna geçecektir. Dolayısıyla, *width* argümanı tablonun doğal genişliğinden büyük olması gerekir. `tabular*` bloğunun `tabular` bloğundan farkı da budur; hatta `array` paketinde de bu bloklar aynıdır.

`tabular*` bloğunda tablonun genişliğinin *width* olması için sütunlar arası-

na ek boşluklar verilir. Aslında, bu boşluksuz da eni *width* olan bir tablo oluşturulabilir. Bunun için ek boşluk yerine her sütunun eni değiştirilmelidir. Böyle bir tablo `tools` sınıfın tabularx paketiyle oluşturulabilir. tabularx paketin komutlu

```
\begin{tabularx}{width}[align]{keys}
    strings
\end{tabularx}
```

parantezi sütunların eni değiştirilebilen bir tablo oluşturmaktadır. `tabularx` bloğunun *keys* argümanında eni esnekli olan bir sütun `X` anahtarıyla işaretlenmelidir. Bu durumda, işaretlenen her sütun `p{..}` tipli sütuna dönüşmektedir. Eğer metinde tabularx paketiyle array paketi de tanıtılmışsa, işaretlenen bir sütun `m{..}` veya `b{..}` tipli sütuna dönüştürülebilir. Bunun için,

```
\tabularxcolumn{col.width}
```

komutu aşağıdaki gibi yeniden tanımlanması gerekir:

```
\newcommand{\tabularxcolumn}[1]{p{#1}}
\renewcommand{\tabularxcolumn}[1]{m{#1}}
```

Bu tanıtımdan sonra, `tabularx` bloğunda `X` anahtarıyla işaretlenen bir sütun `m{..}` tipli sütune dönüşür.

Bir tabloda birkaç sütun `X` anahtarıyla işaretlenmişse, bu sütunlerin eni aynı değerli olur. Böyle sütunlerin de eni farklı ayarlanabilir. Bunun için, `tabularx` bloğunda `\hsize` komutu kullanılmalıdır. Örneğin, *keys* argümanında iki `XX` anahtarı yerine

```
>{\setlength{\hsize}{.5\hsize}X>{\setlength{\hsize}{1.5\hsize}X
```

ifadesi yazılırsa, ikinci sütun eni birinci sütun eninden 3 defa geniş olur. Böyle tanıtımlarda çok dikkatli olmalıdır. Çünkü, bu değerlerin toplamı değişmemesi gerekir.

NOT: Türlü enli `X` tipli sütunler `\multicolumn` komutuyla birleştirilemez.

`tabularx` bloğu başka bir tablo bloğunda de kullanılabilir. Bu durumda, `tabularx` bloğu `{...}` olarak küme parantezi içine alınmalıdır. `tabularx` bloğunda `\centering`, `\raggedleft` ve `\raggedright` komutları da kullanılabilir (*bkz.* 2.7.1, 62.sayfa). Fakat, bu durumda, bu komuttan hemen

sonra

```
\arraybackslash
```

komutu yazılmalıdır. `tabularx` bloğunun bir sütun öğesinde `\footnote` komutu da kullanılabilir (*bkz.* 1.11, 29.sayfa).

2.10.4 Birkaç sayfalı tablolar

LaTeX 'de birkaç sayfalı uzun bir tablo longtable paketin komutlu

```
\begin{longtable}[position]{keys}
    strings
\end{longtable}
```

paranteziyle oluşturulur. `longtable` bloğunda bir tablo yeni bir satırdan başlanıp, *position* argümanın `l`, `r` ve `c` seçeneklerine göre sayfanın sırasıyla sol, sağ ve ortasına yaslanır. Tablonun *keys* argümanı ise, `tabular` bloğunun *keys* argümanıyla aynıdır. array paketin tüm özellikleri `longtable` bloğunda geçerlidir.

`longtable` bloğunda tablonun ilk satırı, genelde, sütunlerin adını belirlemelidir. Üstelik, bu satır, gerekse, tablonun her sayfasında da ilk satır olarak tekrar yazdırılabilir. Bundan başka, tablonun her sayfasın son satırında, gerekse, tablonun devamı öteki sayfada olduğu hakkındaki bir bilgi de yazdırılabilir. Bu gibi istekler `longtable` bloğunun aşağıdaki *özel gruplarıyla* oluşturulabilir.

`longtable` bloğunun ilk başında her biri birkaç satırdan oluşan dört grup tanımlanabilir. Bu gruplar zorunlu değildir, yani onların herhangi biri yazılmayabilir. Eğer yazılan bir gruptan sonra

```
\endfirsthead
```

tanıtımı varsa, bu grup tablonun başına bir başlık olarak yazdırılır. Bu grupta, genelde, tablo ve her sütunün adı yazılır.

Eğer yazılan bir gruptan sonra

`\endhead`

tanıtımı varsa, bu grup tablonun her sayfasında (birinci sayfa hariç) ilk satır olarak yazdırılır. Bu grupta, genelde, sütunlerin numarası veya adı yazılır.

Eğer yazılan bir gruptan sonra

`\endfoot`

tanıtımı varsa, bu grup tablonun her sayfasın (son sayfa hariç) alt kısmına yazdırılır. Bu grupta, genelde, tablonun devamı öteki sayfada olduğu hakkındaki bir bilgi yazılır. Üstelik, bu grupta sütunlerin numarası veya adı yine ek olarak da gösterilebilir.

Eğer yazılan bir gruptan sonra

`\endlastfoot`

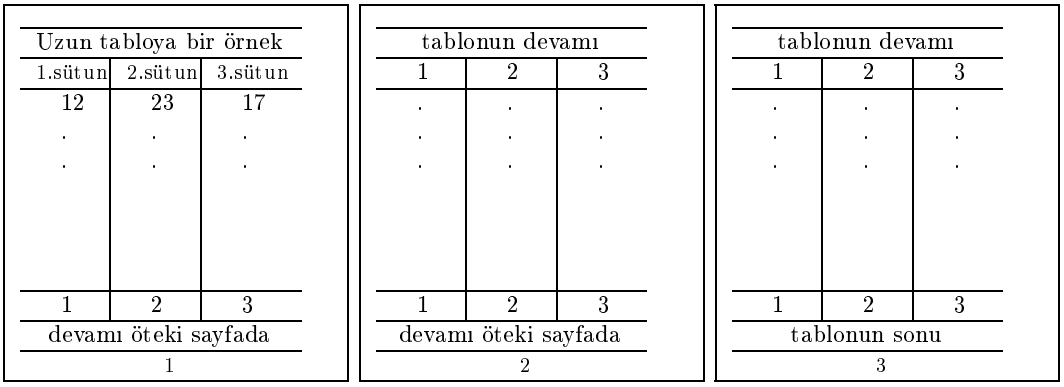
tanıtımı varsa, bu grup tablonun sadece son sayfasın son satırı olarak yazdırılır. Bu grupta, genelde, tablonun sonu olduğu hakkındaki bir bilgi yazılır. Üstelik, bu grupta da sütunlerin numarası veya adı yine ek olarak gösterilebilir.

Bu grupları aşağıdaki bir örnekte izah edelim:

Giriş dosyası:

```
\begin{longtable}{c|r||l}            \hline
      Uzun tabloya bir örnek\\       \hline
      1.sütun & 2.sütun & 3.sütun\\ \hline
  \endfirsthead
                                     \hline
      tablonun devamı\\              \hline
      1 & 2 & 3\\                    \hline
  \endhead
                                     \hline
      1 & 2 & 3\\                    \hline
      devamı öteki sayfada\\         \hline
  \endfoot
                                     \hline
      1 & 2 & 3\\                    \hline
      tablonun sonu\\                \hline
  \endlastfoot
  12 & 23 17 \\
  ...
  ...
\end{longtable}
```

LaTeX 'de görüntüsü:



Uzun tabloya bir örnek		
1.sütun	2.sütun	3.sütun
12	23	17
.	.	.
.	.	.
.	.	.
1	2	3
devamı öteki sayfada		

1

tablonun devamı		
1	2	3
.	.	.
.	.	.
.	.	.
1	2	3
devamı öteki sayfada		

2

tablonun devamı		
1	2	3
.	.	.
.	.	.
.	.	.
1	2	3
tablonun sonu		

3

Örnekten görüldüğü gibi, tablonun başına bir başlık olarak `"Uzun tabloya bir örnek"` sözcüsü ve bir sütun adı olarak ise sırasıyla `1.sütun`, `2.sütun` ve `3.sütun` sözcüsü yazdırılmaktadır. Tablonun, 1.sayfası hariç, her sayfası `"tablonun devamı"` sözcüsü ve her sütun numarasıyla başlanmaktadır. Tablonun, son sayfası hariç, her sayfasın sonu sütun numarası ve `"devamı öteki sayfada"` sözcüsüyle bitmektedir. Tablonun son sayfasın son satırları sütun numarası ve `"tablonun sonu"` sözcüsüyle bitmektedir.
Tablonun kendisi ise üç sütunlü bir metinden oluşan olup, onun normal birinci satırı `12 23 17` rakamlarıyla başlanmaktadır.

`table` bloğunun (*bkz.* 2.11.1)

`\caption[`*entry*`]{`*head*`}`

komutuyla bir tabloya bir `"imza"` yazdırılabilir. Bu durumda, LaTeX bir tabloyu "**Table**" kelimesiyle başlatır, numaralandırır ve iki noktadan sonra `\caption` komutun zorunlu *head* argümanını yazdırır. `babel` paketin *turkish* seçeneği İngilizce **Table** kelimesini Türkçe "**Tablo**" kelimesiyle değiştirir. `\caption` komutun zorunlu olmayan *entry* argümanı *İçindekiler*'in tablolar listesine yazdırılır. Bu argüman gösterilmediği taktirde, LaTeX onun yerine *head* argümanı alır. Tablolar listesinde bir tablo yer almaması için *entry* argümanı `\caption[ ]{`*head*`}` olarak boş kaldırılmalıdır. `\caption` komutun yıldızlı `\caption*` şekli tablonu numaralandırmamaktadır. Bir tabloda `\caption*` komutu kullanılmışsa, bu tablo *tablolar listesinde* yer almayacaktır.

Bir tablo sayfanın sol ve sağ boşluğundan sırasıyla

`\LTleft` ve `\LTright`

komutun değerine eşit olan uzaklıkta ayarlanır. Bu komutların değeri `\setlength` ile değiştirilebilir. Değerler değiştirilmediği taktirde LaTeX bu komutları `\fill` komutuyla eşitlemektedir.

Bir tablodan önce ve sonra koyulacak düşey boşluklar sırasıyla

`\LTpre` ve `\LTpost`

komutlarıyla tanımlanır. Bu komutlar esnekli olup, onların değeri değiştirilmediği taktirde LaTeX bu komutları `\bigskipamount` komutuyla eşitlemektedir. Bir tablo "*imzasının*" eni `\LTcapwidth` komutuyla tanımlanır. Bu komutun değeri değiştirilmediği taktirde LaTeX bu değeri `4in` olarak alır.

NOT: Eğer metinde longtable paketiyle colortbl paketi de tanıtılmışsa, `longtable` bloğu çalışmayabilir, yani colortbl paketi `longtable` bloğunun bazı özelliklerini engellemektedir.

2.11 Kayan nesneler

2.11.1 Şekil ve Tablolar

Bazen metinde bir şekil, tablo veya başka bir nesne geçerli sayfaya sığmadığından dolayı yeni sayfaya alınır. Dolayısıyla, geçerli sayfanın yarısı veya bir kısmı boş kalmaktadır. Böyle durumda, bu nesne kayan bir nesne olarak ayarlanması gerekir. Bu halde, LaTeX metinde sayfaları değiştirerek kayan nesne için uygun bir yer bulur.

LaTeX 'de bir kayan nesne komutlu

```
\begin{figure}[placement]   ...   \end{figure}
\begin{table}[placement]    ...   \end{table}
```

parantezlerin herhangi biriyle oluşturulabilir. İki sütunlu metinde bir kayan nesne sadece bir sütuna yerleştirilir. Bir nesneyi her iki sütuna yerleştirmek için `figure` ve `table` blokların yıldızlı şekli kullanılmalıdır:

```
\begin{figure*}[placement]   ...   \end{figure*}
\begin{table*}[placement]    ...   \end{table*}
```

Bu blokların zorunlu olmayan *placement* argümanı nesnenin sayfadaki yerini tanımlamaktadır. Bu argüman aşağıdaki anahtarların herhangi bir dizisinden oluşuyor olabilir

`h` - geçerli satırdan hemen sonra nesnenin yerleşmesine izin verir. Bu anahtar yıldızlı komutta kullanılamaz;

`t` - nesnenin sayfanın üst kısmına yerleşmesine izin verir;

`b` - nesnenin sayfanın alt kısmına yerleşmesine izin verir;

`p` - nesnenin yeni bir sayfaya yerleşmesine izin verir;

`!` - nesnenin yerleşmesine engel olan anahtarı kaldırır. Bu anahtar kendisinden sonra yazılan anahtarlara aittir ve o değer anahtarların karmaşında kullanılır. Örneğin, argümanın `t!hbp` seçeneğinde `!` anahtarı nesneyi yerleşmesinde engel olan `h`, `b` ve `p` anahtarların herhangi birini kaldırabilir.

placement argümanı gösterilmediği taktirde, LaTeX bu argümanı `tbp` olarak alır.

Eğer LaTeX bir kayan nesneye hiçbir sayfada yer bulamazsa, bu nesneyi çalışmanın son sayfasına (yani metinin sonuna) yazdırır. Aslında metinin herhangi bir yerinde henüz koyulmayan tüm kayan nesneler zorunlu olarak yazdırılabilir. LaTeX 'in

`\clearpage` ve `\cleardoublepage`

komutları metinde henüz koyulmayan tüm kayan nesneleri zorunlu olarak yazdırır. Tersine, bir sayfada bir kayan nesnenin yazdırılması da yasaklanabilir. LaTeX 'in

```
\suppressfloats[key]
```

komutu kendisinden sonra oluşturulan bir kayan nesneyi geçerli sayfaya yazdırılmasına izin vermemektedir.

Komutun zorunlu olmayan *key* argümanında `t` ve/veya `b` anahtarı yazılabilir; yazılan anahtar `\suppressfloats` komutunun sadece bu anahtarlı

kayan nesneyle ilgili olduğunu anlatır. *key* argümanı gösterilmediği taktirde LaTeX bu argümana `t` ve `b` anahtarlarının her ikisini de alır.

Bir şekil veya tabloya

```
\caption[entry]{head}
```

komutuyla bir "imza" yazdırılması 2.10.4. paragrafında ele alınmıştır (*bkz.* 83.sayfa). `\caption` komutunu kısaca hatırlatalım:

LaTeX `\caption` komutu kullanılan `table` bloğunda tabloyu "**Table**" kelimesiyle, `figure` bloğunda ise "**Figure**" kelimesiyle başlatır, numaralandırır ve iki noktadan sonra `\caption` komutun zorunlu *head* argümanını yazdırır. `\caption` komutun zorunlu olmayan *entry* argümanı *İçindekiler*'in sırasıyla tablo ve şekiller listesine yazdırılır. Bu argüman gösterilmediği taktirde, LaTeX onun yerine *head* argümanı alır. Tablo ve/veya şekiller listesinde bir tablo ve/veya şekil yer almaması için *entry* argümanı `\caption[ ]{`*head*`}` olarak boş bırakılmalıdır. `\caption` komutunun yıldızlı `\caption*` şekli tablonu numaralandırmamaktadır. Bir tablo (veya şekilde) `\caption*` komutu kullanılmışsa, bu tablo (veya şekil) *tablolar* (veya *şekiller*) *listesinde* yer almayacaktır.

İngilizce **Figure** ve **Table** kelimeleri sırasıyla

```
\figurename        \tablename
```

komutlarında saklanmaktadır. Bu kelimeler `\renewcommand` komutuyla değiştirilebilir. `babel` paketinin *turkish* seçeneği bu kelimeleri sırasıyla Türkçe **Şekil** ve **Tablo** kelimesiyle yazdırır. Bu kelimelerden sonra iki nokta yerine bir nokta koyulması için metin başında LaTeX 'in `caption2` paketi yüklenip, bu paketin `\captionlabeldelim` komutu

```
\renewcommand{\captionlabeldelim}{.}
```

gibi yeniden tanımlanması gerekir.

Sayfaya kayan nesnenin yerleştirilmesi

Bir sayfanın üst, alt ve herhangi bir yerine yerleşmesi mümkün olan kayan nesnelerin en büyük sayısı sırasıyla

```
topnumber     bottomnumber     totalnumber
```

değişkenlerinde saklanmaktadır. Değişkenlere yeni bir değer verilmezse LaTeX bu değerleri sırasıyla `2`, `1` ve `3` olarak alır. İki sütunlü metinde bu değişkenler sadece bir sütuna yerleşecek kayan nesneleri takip eder. LaTeX 'in

```
\topfraction        \bottomfraction
```

komutları bir sayfanın sırasıyla üstüne ve altına yerleşecek kayan nesnesine ayırtılan sayfanın en büyük kısmını tanımlamaktadır.

`\topfraction` ve `\bottomfraction` komutlarına yeni bir değer verilmezse LaTeX bu değerleri sırasıyla `0.7` ve `0.3` olarak alır. İki sütunlu metinde bu komutlar sadece bir sütuna yerleşecek kayan nesneyi takip eder. LaTeX 'in

```
\textfraction
```

komutu bir sayfada metine ayırılan sayfanın en büyük kısmını tanımlamaktadır. `\textfraction` komutuna yeni bir değer verilmezse LaTeX bu değeri `0.2` olarak alır.

Nesneden önce ve sonra koyulacak düşey aralık

Bir yüzücü nesneden önce ve sonra koyulacak düşey aralık nesneyi yerleştiren *placement* argümanın anahtarına bağlıdır:

anahtar

`h` `\intextsep` uzunluk birimi nesneden önce ve sonra koyulan bir düşey aralığı tanımlar. Bu komuta yeni bir değer verilmezse LaTeX bu değeri `12pt` olarak alır.

`t`, `b` `\textfloatsep` uzunluk birimi nesneyle metin arasındaki bir düşey aralığı tanımlar. İki sütunlü metinde bu aralığı `\dbltextfloatsep` komutu tanımlar. Bu iki komuta yeni

bir değer verilmezse LaTeX bu değeri `20pt` olarak alır. `\floatsep` uzunluk birimi iki nesne arasındaki bir düşey aralığı tanımlar. İki sütunlü metinde bu aralığı `\dblfloatsep` komutu tanımlar. Bu iki komuta yeni bir değer verilmezse LaTeX bu değeri `12pt` olarak alır.

Tüm bu komutlar esnektir.

2.11.2 Akan metinli şekil ve tablolar

Bir `figure` veya `tablo` bloğunda bir kayan nesne yerleştiği her satırın tümünü meşgul etmektedir. Dolayısıyla, eğer kayan nesnenin eni küçükse, bu nesne yerleştiği satırların büyük kısmı boş kalmaktadır. Aslında, böyle bir nesnenin sol ve/veya sağında metinin devam etmesi doğaldır. Bu şekilde ayarlanan kayan nesneye *akan metinli nesne* denir.

Bir `figure` ve `tablo` bloğunda, maalesef, akan metinli bir kayan nesne oluşturulamaz. Akan metinli bir kayan nesne floatflt paketinin

`floatingfigure` ve `floatingtable`

bloklarıyla oluşturulabilir. Bu bloklarda bir nesne sayfanın sol veya sağına yazdırılıp, nesnenin üst sınırı geçerli satır düzeyinde ayarlanır.

floatflt paketinin `rflt`, `lflt` ve `vflt` gibi üç seçeneği vardır:

seçenek	
`lflt`	metindeki tüm akan metinli kayan nesneler sayfanın soluna yazdırılır;
`rflt`	metindeki tüm akan metinli kayan nesneler sayfanın sağına yazdırılır;
`vflt`	akan metinli kayan nesne çift sayfalarda sayfanın soluna, tek sayfalarda ise, sayfanın sağına yazdırılır.

Örneğin, belgenin başlık kısmına

```
\usepackage[lflt]{floatflt}
```

tanıtımı yazılmışsa, `floatingfigure` ve `floatingtable` bloğundaki tüm nesneler sayfanın soluna yazdırılır. floatflt paketinin seçeneği gösterilmediği taktirde, LaTeX bu seçeneği `vflt` olarak alır.

`floatingfigure` bloğu komutlu

```
\begin{floatingfigure}[position]{width}
  figure
\end{floatingfigure}
```

paranteziyle tanımlanmaktadır. Bloğun *width* argümanında *figure* nesnesinin eni gösterilmelidir, *position* argümanında ise nesnenin sayfadaki yeri `r`, `l` ve `p` seçeneklerinin herhangi biriyle gösterilir. Bu seçenekler sırasıyla `rflt`, `lflt` ve `vflt` seçeneklerle aynıdır. *position* argümanın bir `v` seçeneği daha vardır. Bu seçenek nesne için paketin seçeneğini kullanılmasını belirlemektedir.

`floatingtable` bloğu komutlu

```
\begin{floatingtable}[position]{tabular environment}
[\caption{text}]
\end{floatingtable}
```

paranteziyle tanımlanmaktadır. `floatingtable` bloğunda *tabular environment* argümanı zorunludur. Bu argümanda bir `tabular` veya `\parbox` bloğu yazılmalıdır. Akan metinin eni nesnenin enini tanımlamaktadır. Bloğun zorunlu olmayan *position* argümanı `floatingfigure` bloğundaki gibidir.
`figure` ve `table` bloklarında olduğu gibi, `floatingfigure` ve `floatingtable` bloklarında da bir nesneye `\caption` komutuyla bir *imza* yazdırılabilir.

`floatingfigure` ve `floatingtable` bloklarıyla oluşturulan akan metinli nesneyi bir örnekle gösterelim:

Giriş dosyası:

```
\begin{floatingfigure}[r]{5.5cm}
    \begin{tabular}{|lr}
       Türkiye & Istanbul\\
       Trabzon & Izmir\\ \hline
    \end{tabular}
  \caption{\footnotesize{\tt floatingfigure} bloğu}
\end{floatingfigure}
Bu bir {\tt floatingfigure} bloğuna bir örnektir. Bu küçük paragraftaki
tüm metin {\tt floatingfigure} bloğu için akan bir metindir.\\
```

```
\begin{floatingtable}[l]{
    \begin{tabular}{lr|}
       45 & 11 \\
       78 & 292\\ \hline
    \end{tabular} }
  \caption{\footnotesize{\tt floatingtable}}
\end{floatingtable}
Bu bir {\tt floatingtable} bloğuna bir örnektir. Bu küçük paragraftaki
tüm metin {\tt floatingtable} bloğu için akan bir metindir.
```

LaTeX 'de görüntüsü :

Bu bir `floatingfigure` bloğuna bir örnektir. Bu küçük paragraftaki tüm metin `floatingfigure` bloğu için akan bir metindir.

Türkiye	İstanbul
Trabzon	İzmir

Şekil 2.2: `floatingfigure` bloğu

45	11
78	292

Tablo 2.2: `floatingtable`

Bu bir `floatingtable` bloğuna bir örnektir. Bu küçük paragraftaki tüm metin `floatingtable` bloğu için akan bir metindir.

2.11.3 Boşluk notu

LaTeX 'de metinin sol veya sağ boşluğuna bir not

`\marginpar[`*left-text*`]{`*text*`}`

komutuyla yazdırılır. `\marginpar` komutunun zorunlu *text* argümanında notun metni yazılmalıdır. Tek taraflı metinde [5] bir boşluk notu sayfanın sağına; iki taraflı metinde [6], tek sayfalarda sağa, çift sayfalarda ise sola; iki sütunlu metinde ise en yakındaki bir boşluğa yazdırılır. Ayrıca, komu-

[5] `\documentstyle` veya `\documentclass` tanıtımının `oneside` seçeneği alınır

[6] `\documentstyle` veya `\documentclass` tanıtımın `twoside` seçeneği alınır

tun zorunlu olmayan *left-text* argümanıyla sol boşluğa da başka bir not yazdırılabilir. *left-text* argümanın vazifesi tersine dönüştürülebilir, yani bu argümanla sağ boşluğa başka bir not yazdırılabilir: LaTeX 'in

```
\reversemarginpar
```

tanıtımı `\marginpar` komutun *left-text* argümanını *right-text* ile değiştirmektedir. Dolayısıyla, `\reversemarginpar` tanıtımından sonra *left-text* argümanına yazılan bir metin sayfanın sağ boşluğuna bir not olarak yazdırılır. Böyle bir değiştirmeden sonra yine geri dönmek için

```
\normalmarginpar
```

tanıtımı yazılmalıdır. `\normalmarginpar` tanıtımı *left-text* argümanı eski haline getirir.

Bir boşluk notunun ilk satırı `\marginpar` komutu yazılan satır düzeyinde ayarlanır. Bu komut iki paragraf arasına yazılmışsa, boşluk notu birinci paragrafın son satırı düzeyinden başlanır. Eğer bir boşluk notu bir önceki boşluk notunun üstüne çıkıyorsa, LaTeX ikinci boşluk notu birinci boşluk notunun altına yazdırır. Bir boşluk notunun yatay ve düşey boyutu, aynı zamanda iki boşluk notu arasındaki düşey aralık şekil 1.1 'de gösterilmiştir (*bkz.* 9.sayfa).

2.12 Metin yazısı

2.12.1 Yazı özellikleri

LaTeX 'de bir yazının beş özelliği vardır:

encoding – yazının şifre özelliğidir. Standart sınıflarda bu özelliğin `OT1` değeri kullanılır;
family – yazının tip özelliği;
series – yazının doygunluk özelliği;
shape – yazının biçim özelliği;
size – yazının boyut özelliği.

Yazının tüm özellikleri bir-birinden bağımsızdır. Eğer bir yazı için seçilen özellikler uygun değilse, LaTeX bunu bildirerek seçilen özelliklere yakın

olan başka bir özellikleri alır. Seçilen özellikler grubun uygun olup olmadığı yazı özelliklerini tanımlayan `*.fd` tipli dosyalardan öğrenilebilir. Bu dosyaların adı yazı tipi ve şifresinden oluşandır, örneğin, `OT1` şifreli Computer Modern Roman ailesin yazı dosyası `ot1cmr.fd` 'dir.

LaTeX 'de bir yazının farklı özelliklerini tanımlayan çeşitli komutlar vardır. Bu komutlar 2.12.2 - 2.12.5 paragraflarda ele alınmaktadır.

2.12.2 Yazı tipi

Metinde

`\rmfamily`	`\sffamily`	`\ttfamily`

komutları geçerli yazı tipini tanımlamaktadır. Bu komutların değeri sırasıyla

`\rmdefault`	`\sfdefault`	`\ttdefault`

komutlarında saklanmaktadır. Standart sınıflarda bu komutlara Computer Modern ailesin sırasıyla `cmr` (Times tipli), `cmss` (Helvetica veya Arial tipli) ve `cmtt` (matbaa yazısı, Courier tipli) değeri verilmektedir. Bir değer `\renewcommand` komutuyla değiştirilebilir:

```
\renewcommand{\rmdefault}{ptm}
```

tanıtımı yazının Computer Modern Roman tipine Adobe Times Roman yazı tipinin `ptm` değeriyle değiştirir [7].

Kısa metin için `\renewcommand` komutu yerine aşağıdaki komutlar kullanılabilir

`\textrm{`*text*`}`	`\textsf{`*text*`}`	`\texttt{`*text*`}`

Yazı tipini bir örnekle gösterelim:

[7] psnfss paketi ek şeklinde tanıtılması da gerekir.

```
... \textsf{sf tipli metin},
\textrm{rm tipli metin},
\texttt{tt tipli metin},
normal metin. ...
```

... sf tipli metin,
rm tipli metin,
`tt tipli metin`,
normal metin. ...

2.12.6. paragrafında tanımlanan `\fontfamily` ve `\usefont` komutları bir yazının istenilen tipinin seçilebilmesine izin verir.

2.12.3 Yazı doygunluğu

Metinde

`\mdseries`	`\bfseries`

komutları geçerli yazı doygunluğunu tanımlamaktadır. Bu komutların değeri sırasıyla

`\mddefault`	`\bfdefault`

komutlarında saklanmaktadır. Standart sınıflarda bu komutlara bir yazı doygunluğunu sırasıyla `m` (*medium*) normal ve `b` (*bold*) yarı-koyu değeri verilmektedir. Yazı doygunluğu kısa metinde aşağıdaki komutlarla değiştirilebilir

`\textmd{`*text*`}`	`\textbf{`*text*`}`

Yazı doygunluğunu bir örnekle gösterelim:

```
... \textmd{md doygunlu yazı},
\textbf{bf doygunlu yazı},
normal doygunlu yazı. ...
```

... md doygunlu yazı,
bf doygunlu yazı,
normal doygunlu yazı. ...

2.12.6. paragrafında tanımlanan `\fontfamily` ve `\usefont` komutları bir yazının istenilen doygunlukta seçilebilmesine izin verir.

2.12.4 Yazı biçimi

Metinde

\upshape	\itshape	\slshape	\scshape

komutları geçerli yazı biçimini tanımlamaktadır. Bu komutların değeri sırasıyla

\updefault	\itdefault	\sldefault	\scdefault

komutlarında saklanmaktadır. Standart sınıflarda bu komutlara bir yazı biçiminin sırasıyla `n` (*normal*), `it` (*italic*), `sl` (*slanted*) ve `sc` (*small* caps) değeri verilmektedir. Bir metinde yazı biçimi gösterilmemişse, LaTeX yazı biçimin `\updefault` değerini alır. Yazı biçimi kısa metinde aşağıdaki komutlarla değiştirilebilir

\textup{*text*} \textit{*text*} \textsl{*text*} \textsc{*text*}

Yazı biçimini bir örnekle gösterelim:

```
... \textup{up biçimli yazı},
\textit{it biçimli yazı},
\textsl{sl biçimli yazı},
\textsc{sc biçimli yazı},
normal biçimli yazı. ...
```

... up biçimli yazı,
it biçimli yazı,
sl biçimli yazı,
SC BIÇIMLI YAZI,
normal biçimli yazı. ...

2.12.6. paragrafında tanımlanan `\fontfamily` ve `\usefont` komutları bir yazının istenilen biçimli olarak seçilebilmesine izin verir.

LaTeX 'de yazı biçimine ait bir *özgü* komutu daha vardır:

\emph{*text*}

Eğer geçerli metin normal biçimde ise, `\emph` komutu *text* argümanı `it` biçiminde yazdırır; eğer geçerli metin `it` biçiminde ise, *text* argümanı normal biçiminde yazdırır.

2.12.5 Yazı boyutu

Tablo 2.3 'de geçerli metin yazı boyutunu tanımlayan komutlar verilmektedir. Bu komutlar sadece yazı boyutunu değil, satırlar arası uzaklığı da

değiştirir.

Tablo 2.3: Standart sınıfların *size* seçeneğinin `10pt`, `11pt` ve `12pt` değerleri için yazı boyutunu tanımlayan komutlar

komut	*boyut*			komut	*boyut*		
`\tiny`	5pt	6pt	6pt	`\large`	12pt	12pt	14pt
`\scriptsize`	7pt	8pt	8pt	`\Large`	14pt	14pt	17pt
`\footnotesize`	8pt	9pt	10pt	`\LARGE`	17pt	17pt	20pt
`\small`	9pt	10pt	11pt	`\huge`	20pt	20pt	25pt
`\normalsize`	10pt	11pt	12pt	`\Huge`	25pt	25pt	25pt

2.12.6. paragrafında tanımlanan `\fontsize` komutu bir yazının istenilen boyutta seçilebilmesine izin verir. Yazı boyutunu bir örnekle gösterelim:

```
{\tiny tiny                    6pt}
{\scriptsize scriptsize        8pt}
{\footnotesize footnotesize   10pt}
{\small small                 11pt}
{\normalsize normalsize       12pt}
{\large large                 14pt}
{\Large Large                 17pt}
{\LARGE LARGE                 20pt}
{\huge huge                   25pt}
{\Huge Huge                   25pt}
```

tiny 6pt
scriptsize 8pt
footnotesize 10pt
small 11pt
normalsize 12pt
large 14pt
Large 17pt
LARGE 20pt
huge 25pt
Huge 25pt

2.12.6 İstenilen yazı tanımı

Metinde

`\fontencoding{`*encoding*`}`

komutu bir yazının *encoding* şifresini tanımlamaktadır;

`\fontfamily{`*family*`}` `\fontseries{`*series*`}` `\fontshape{`*shape*`}`

komutları bir yazının sırasıyla *family* tipi, *series* doygunluğu ve *shape*

biçimini tanımlamaktadır;

`\fontsize{`*size*`}{`*baselineskip*`}`

komutunun *size* argümanı bir yazının boyutunu, *baselineskip* argümanı ise satırlar arası uzaklığı tanımlamaktadır. *baselineskip* ve *size* argümanlarının değeri değiştirilmezse, LaTeX bu değerleri sırasıyla `12pt` ve `10pt` olarak alır.

Yukarıda tanımlanan `\font ...` komutları sadece yeni yazı özelliklerini tanımlamaktadır. Böyle özellikli bir yazı geçerli metine etkili olması için

`\selectfont`

komutu yazılmalıdır. Bu durumda `\font ...` ve `\selectfont` komutları arasında (boşluklar hariç) bir metin olmaması gerekir. Bunu bir örnekle gösterelim:

```
\fontencoding{U}\fontfamily{psy}\selectfont
```

tanıtımından sonradaki metin Adobe Symbol (`psnfss` paketiyle desteklenmektedir) yazısıyla yazdırılır.

TeX 'in

`\usefont{`*encoding*`}{`*family*`}{`*series*`}{`*shape*`}`

komutu bir yazının tüm özelliklerini (boyutu hariç) yeniden tanımlamaktadır. Bu komut geçerli metinden başlanarak etkili olur. Bunu bir örnekle gösterelim:

```
\usefont{T1}{pzc}{m}{it}
```

tanıtımından sonradaki metin Adobe Zapf Chancery (`psnfss` paketiyle desteklenmektedir) yazısıyla yazdırılır.

TeX 'in

`\symbol{`*code*`}`

komutu geçerli yazının *code* şifre simgesini yazdırır. `\symbol` komutun *code* argümanı `0` ve `255` doğal sayıları arasında değişir. Bu argüman sekizlik veya onaltılık hesap sistemlerinde de verilebilir. Bu durumda, sekizlik hesap sistemin sayısı `'` simgesiyle, onaltılık hesap sistemin sayısı

ise " simgesiyle başlanmalıdır.

Aşağıdaki örnekte şifresi onluk, sekizlik ve onaltılık hesap sistemlerin 14, 16, 17, 21, 51 ve 123 rakamlarında yerleşen bir yazının bazı özellikleri yazdırılmaktadır:

```
\symbol{17}
{\tt \symbol{16}}
{\bf \symbol{123}}
{\it \symbol{14}}
\symbol{51}
\symbol{'51}
\symbol{"51}
\symbol{21}
\symbol{'21}
\symbol{"21}
```

”
“
{
‹
3
)
Q
–
”
!

Üstelik `\usefont` yazısı `\verb!\usefont!` gibi de yazdırılabilir veya yukarıdaki komut yardımıyla `{\tt \symbol{'134}usefont}` gibi de yazdırılabilir.

2.12.7 Metinin taban yazısı

LaTeX 'de

```
\normalfont
```

komutunun etki dairesindeki veya

```
\textnormal{text}
```

komutunun *text* argümanındaki bir metin giriş dosyasının taban yazısıyla yazdırılır. Bunu bir örnekle gösterelim:

```
\textbf{yarı-koyu metin {\normalfont
normal yazılı metin} yine yarı-koyu
metin} ... \textit{italik yazılı metin
\textnormal{normal yazılı metin} yine
italik yazılı metin} ...
```

yarı-koyu metin normal yazılı metin **yine yarı-koyu metin** ...
italik yazılı metin normal yazılı metin *yine italik yazılı metin* ...

2.13 Yeni makro tanımlar

2.13.1 Komutlar

Bir metinde TEX ve LATEX 'in standart komutları dışında istenilen fonksiyonlu yeni bir komut tanımlanabilir:

`\newcommand{`*cmd*`}[`*integer*`][`*default*`]{`*definition*`}`

`\newcommand` tanıtımı *definition* metiniyle tanımlanan *cmd* adlı yeni bir komutu tanıtır. Tanıtımın *integer* seçeneği `1` ila `9` arasındaki bir doğal sayısı olup, tanıtımın argümanları sayısını belirlemektedir. Eğer tanıtımın *default* seçeneği de varsa, birinci argüman zorunlu olmayan argümana dönüşür. Birinci argüman gösterilmediği taktirde LATEX (veya TEX) bu argümana *default* seçeneğinin değerini alır.

`\newcommand` tanıtımın *definition* argümanı seçeneklerin bir argümanını `#`*n* şeklinde içerir, burada *n* argümanın numarasıdır.

NOT: `\newcommand` tanıtımın *cmd* argümanı `\` simgesiyle başlanması lazım

`\newcommand` tanıtımına birkaç örnek verelim:

```
\newcommand{\turkL}{Türkçe \TeX{},
        \LaTeX{},\LaTeXe{},\dots}
Konu adı: \\ \qquad "\turkL ".
⋮
Biz \turkL öğreneceğiz.
```

Konu adı:
"Türkçe TEX, LATEX, LATEX 2_ε, ... ".
⋮
Biz Türkçe TEX, LATEX, LATEX 2_ε, ... öğreneceğiz.

Aşağıdaki örnekte *integer* ve *default* seçenekleri de varolan `\newcommand` komutu ele alınmaktadır:

Giriş dosyası:

```
\newcommand{\be}{\begin{enumerate}}
\newcommand{\ee}{\end{enumerate}}
\newcommand{\gn}{$\gamma _n$}
\newcommand{\Gn}{$\Gamma _n$}
\newcommand{\ii}[2][k]{\item[{[#1]}]}
\newcommand{\F}[2][M]{$\{#2_{-15}, \ldots ,#2_{#1}\}$}
```

```
\be
   \ii[1]{1} eğer \Gn = \gn ise, \Gn + \gn = \F[k]{x} 'dir;
   \ii[2]{2} eğer \Gn $\not=$ \gn ise, \Gn + \gn = \F{y} 'dir.
\ee
```

LaTeX 'de görüntüsü:

[1] eğer $\Gamma_n = \gamma_n$ ise, $\Gamma_n + \gamma_n = \{x_{-15}, \ldots, x_k\}$ 'dir;
[2] eğer $\Gamma_n \neq \gamma_n$ ise, $\Gamma_n + \gamma_n = \{y_{-15}, \ldots, y_M\}$ 'dir.

Bu örnekdeki

```
\newcommand{\F}[2][M]{$\{#2_{-15}, \ldots ,#2_{#1}\}$}
```

tanıtımını açıklayalım.

Bu tanıtım iki argümanlı `\F` adlı bir komutu tanımlamaktadır. `\F` komutunun 1.argümanı gösterilmediği taktirde LaTeX bu argümanı M olarak alır. `\F[k]{x}` komutunun 1.argümanı `k` ve 2.argümanı ise `x` 'dir. Dolayısıyla, `#2_{-15}` ve `#2_{#1}` ifadelerin LaTeX 'de görüntüsü sırasıyla x_{-15} ve x_k olur. Buradan `$\{#2_{-15},\ldots,#2_{#1}\}$` ifadesin LaTeX 'de görüntüsü $\{x_{-15}, \ldots, x_k\}$ olur.

Şimdi `\F{y}` ifadesini ele alalım. Bu komutun 1.argümanı olmadığından, LaTeX bu argümanı M olarak alır; komutun 2.argümanı ise `y` 'dır. Buradan, `#2_{-15}` ve `#2_{#1}` ifadelerin LaTeX 'de görüntüsü sırasıyla y_{-15} ve y_M olur. Dolayısıyla, `$\{#2_{-15}, \ldots ,#2_{#1}\}$` ifadesin görüntüsü $\{y_{-15}, \ldots, y_M\}$ olur.

NOT: `\newcommand` komutuyla LaTeX'de tanımlı bir komut yeniden tanımlanamaz, yani *cmd* argümanında LaTeX'de varolan bir komut yazılamaz.

LaTeX'de varolan bir komut

`\renewcommand{`*cmd*`}[`*integer*`][`*default*`]{`*definition*`}`

tanıtımıyla yeniden tanımlanabilir. `\renewcommand` tanıtımın *integer* ve *default* seçenekleri, aynı zamanda argümanları `\newcommand` tanıtımın sırasıyla seçenek ve argümanlarıyla aynıdır. Önceki paragraflarda `\renewcommand` tanıtımına birkaç örnek ele alınmıştır (*bkz.* 49, 61 ve 80. say-

falar). Daha sonra da böyle örnekler ele alınacaktır.

Bazen yeniden tanımlanacak bir komutun LaTeX 'de varolup olmadığı bilinmemektedir. Böyle durumda, `\renewcommand` tanıtımı yerine

`\providecommand{`*cmd*`}[`*integer*`] [`*default*`]{`*definition*`}`

tanıtımı kullanılmalıdır. Eğer *cmd* adlı bir komut LaTeX 'de varsa, bu tanıtım varolan komutu alır, aksi halde, tanıtımdaki yeni komutu alır.

`\providecommand` tanıtımın *integer* ve *default* seçenekleri, aynı zamanda argümanları `\newcommand` tanıtımın sırasıyla seçenek ve argümanlarıyla aynıdır.

`\providecommand` tanıtımına bir örnek verelim:

`\gamma` komutu matematik bir blokta (örneğin: `$\gamma$`) γ simgesini anlatır (*bkz.* tablo 3.7, 135.sayfa). Dolayısıyla bu komut TeX, LaTeX, ... 'de varolan bir komuttur. Bu durumda,

```
\providecommand{\gamma}{\gamma + \gamma ^2}
```

tanıtımında (`\gamma`, γ) çiftlik [8] yeniden (`\gamma`, $\gamma + \gamma^2$) olarak tanıtılması rağmen metinde yine (`\gamma`, γ) çiftlik çalışmaktadır. Eğer bu tanıtımda `\providecommand` yerine `\renewcommand` yazılırsa, metinde (`\gamma`, $\gamma + \gamma^2$) çiftliği çalışacaktır.

TeX 'de `\newcommand` tanıtımın aşağıdaki basit bir şekli de vardır:

`\def`*cmd* `#1#2...`*i*`{`*def*`}`

`\def` tanıtımı *def* ile tanımlanan *cmd* adlı yeni bir komutu tanımlamaktadır. Bu tanıtımın seçenekleri komutun *cmd* adından hemen sonra `#1#2...`*i* argümanında gösterilmelidir, burada $i \leq 9$ 'dir. Argümanlar arasında boşluklar olmamalıdır. `\newcommand` tanıtımında olduğu gibi, `\def` tanıtımın da *cmd* argümanı \ simgesiyle başlanmalıdır. `\def` tanıtımını bir örnekle gösterelim:

Giriş dosyasının başlık kısmına:

```
\def\R{{\rm I\!\!R}}
\def\N{{\rm I\!\!N}}
\def\e{{\bf 1}\!\!{\rm I}}
\def\ru{\rule{2.0mm}{2.5mm}}
```

[8] çiftlikte: sırasıyla giriş dosyası ve görüntüsü yazılmıştır

```
\def\chaptername{B\"ol\"um}
\def\contentsname{\.I\c cindekiler}
\def\listfigurename{\c Sekil Listesi}
\def\listtablename{Tablo Listesi}
```

tanıtımları yazılırsa, metinde `$\R$ , $\N$ , $\e$ , \ru` olarak yazılan komutların LaTeX 'de görüntüsü sırasıyla şöyle olur: $\mathbb{R}$, $\mathbb{N}$, $\mathbb{1}$, ■

Üstelik, metinde kullanılan `\chapter`, `\contents`, `\listfigure` ve `\listtable` komutları sırasıyla Türkçe *Bölün*, *İçindekiler*, *Sekil Listesi* ve *Tablo listesi* kelimeleri yazdırır. Dolayısıyla, belgenin başlık kısmında

```
\usepackage[turkish]{babel}
```

tanıtımı kullanılmadan da İngilizce başlıklar, bu şekilde elle, Türkçe başlıklarla değiştirilebilir (*bkz.* 1.4, 6. sayfa).

2.13.2 Komutlu parantezler

LaTeX komutlu bir parantez yardımıyla yeni bir komutlu parantez oluşturabilir. LaTeX 'in

`\newenvironment{`*name*`}[`*integer*`][`*default*`]{`*begdef*`}{`*enddef*`}`

tanıtımı *name* adlı yeni bir bloğu tanımlamaktadır. Bu blok normal komutlu bir parantez bloğu gibi

```
\begin{name}
 ...
\end{name}
```

şeklindedir. LaTeX `\begin{`*name*`}` 'yi *begdef* argümanla, `\end{`*name*`}` 'yi ise *enddef* argümanla değiştirir.

`\newenvironment` tanıtımın *integer* seçeneği `1` ve `9` arasındaki bir doğal sayısıdır; bu seçenek *name* bloğunun argümanları sayısını belirlemektedir. Eğer tanıtımın *default* seçeneği varsa, *name* bloğunun 1.argümanı zorunlu olmayan argümana dönüşür. Bu argüman gösterilmediği taktirde LaTeX bu argümana *default* seçeneği alır. *definition* ise bu argümanları `#`*n* şeklinde içerir, burada *n* argümanın numarasıdır.

`\newenvironment` tanıtımına bir örnek verelim:

Giriş dosyası:

```
\newenvironment{ozelquote}[1]{\begin{quote}{\bf #1}:\it}{\end{quote}}
\dots \\
{\it ozelquote} bloğunun bir "Uyarı" için uygulaması
\begin{ozelquote}{Uyarı}
  Uyarı kelimesi yarı-koyu yazısıyla, \textnormal{ozelquote} bloğunun
  içerisi ise italik yazısıyla yazdırılmaktadır.
\end{ozelquote}
```

LaTeX 'de görüntüsü:

...
ozelquote bloğunun bir "Uyarı" için uygulaması

> **Uyarı**: *Uyarı kelimesi yarı-koyu yazısıyla,* ozelquote *bloğunun içerisi ise italik yazısıyla yazdırılmaktadır.*

Bu örnekte komutlu `\begin{quote} ... \end{quote}` parantezi yardımıyla yeni bir komutlu `\begin{ozelquote} ... \end{ozelquote}` parantezi oluşturulmuştur.

LaTeX 'de varolan bir *name* bloğu

`\renewenvironment{`*name*`}[`*integer*`][`*default*`]{`*begdef*`}{`*enddef*`}`

tanıtımıyla yeniden tanımlanabilir. Bu tanıtımın seçenek ve argümanları `\newenvironment` tanıtımın sırasıyla seçenek ve argümanlarıyla aynıdır. Örneğin, yukarıda tanımlanan *ozelquote* bloğu

```
\renewenvironment{ozelquote}[1]{\begin{quote}{\sc #1}:\sf}{\end{quote}}
```

olarak yeniden tanımlanabilir. Şu halde, bu blok giriş dosyasına:

```
\begin{ozelquote}{Uyarı}
   Uyarı kelimesi "sc" yazısıyla ve \textnormal{ozelquote}
   bloğunun içerisi ise "sf" yazısıyla yazdırılmaktadır.
\end{ozelquote}
```

olarak yazılırsa, onun LaTeX 'de görüntüsü şöyle olur:

> UYARI: Uyarı kelimesi "sc" yazısıyla ve ozelquote bloğunun içerisi ise "sf" yazısıyla yazdırılmaktadır.

2.13.3 Makro tanımların *-yıldızlı şekli

2.13.1 ve 2.13.2 paragrafında tanımlanan beş komutun *-yıldızlı şekli de vardır:

`\newcommand*{`*cmd*`}[`*integer*`][`*default*`]{`*definition*`}`

`\renewcommand*{`*cmd*`}[`*integer*`][`*default*`]{`*definition*`}`

`\providecommand*{`*cmd*`}[`*integer*`][`*default*`]{`*definition*`}`

`\newenvironment*{`*name*`}[`*integer*`][`*default*`]{`*begdef*`}{`*enddef*`}`

`\renewenvironment*{`*name*`}[`*integer*`][`*default*`]{`*begdef*`}{`*enddef*`}`

Birkaç argümanlı tanıtımların *-yıldızlı şekli daha da dayanaklıdır. Bu durumda, tanıtımın argümanları *-yıldızlı bir komut veya `\par` komutunu içermemesi gerekmektedir.

2.14 Simgeler

2.14.1 Yardımcı simgeler

TEX ve LATEX 'de `#` , `$` , `&` , `{` , `}` , `_` ve `%` simgeleri yardımcı olarak kullanılır. Metinde bu simgeleri sırasıyla

`\#`	`\$`	`\&`	`\{`	`\}`	`\_`	`\%`

komutları yazdırmaktadır. `^` ve `~` rozetleri de TEX ve LATEX 'in yardımcı simgesidir. Bu simgeleri sırasıyla `\^{ }` ve `\~{ }` komutları yazdırır. `\textasciicircum` ve `\textasciitilde` komutları da bu simgeleri bir az değişik şeklinde yazdırır: ^ ve ~. TEX ve LATEX 'in en önemli yardımcı `\` simgesini \textbackslash komutu yazdırır.

2.14.2 Avrupa alfa-sayısalın ulusal simgeleri

LATEX 'de tabanı Latince olan Avrupa alfa-sayısalın ulusal simgeleri de yazdırılabilir. Latince'de olmayan Türkçe harfler hakkında 1.4. paragra-

fında söz etmiştik (bkz. 6. sayfa). Bu harfler tablo 2.4 'de yine ele alınmaktadır.

Tablo 2.4: Latince'de olmayan Türkçe harfler

komut	görüntüsü	komut	görüntüsü
`\c{c}`	ç	`\"{o}`	ö
`\u {g}`	ğ	`\c{s}`	ş
`\i␣`	ı	`\"{u}`	ü

Bazı Avrupa devletlerin (Fransız, Alman, İspanya, Polonya, ...) alfa-sayısalın Latince'de olmayan ulusal harf ve simgeleri tablo 2.5 ve 2.6 'de verilmektedir.

Tablo 2.5: Latince'de olmayan bazı harf ve simgeleri tanımlayan komutlar

komut	görüntüsü	komut	görüntüsü	komut	görüntüsü
`\‘{o}`	ò	`\.{o}`	ȯ	`\H{o}`	ő
`\’{o}`	ó	`\={o}`	ō	`\u{o}`	ŏ
`\"{o}`	ö	`\b{o}`	o̲	`\v{o}`	ǒ
`\^{o}`	ô	`\c{o}`	ǫ	`\t{oo}`	o͡o
`\~{o}`	õ	`\d{o}`	ọ	`\k{o}` [a]	ǫ

[a] `fontenc` paketi `T1` seçeneğiyle tanıtıldığından sonra geçerlidir.

Tablo 2.6: Avrupa alfa-sayısalın önemli simgeleri

`\aa`	å	`\AA`	Å	`\ae`	æ	`\AE`	Æ	`\o`	ø
`\oe`	œ	`\OE`	Œ	`\l`	ł	`\L`	Ł	`\ss`	ß
`\O`	Ø	`!‘`	¡	`\SS`	SS	`?‘`	¿		

Örneğin, Fransızca "chef-d'œuvre" (şaheser) ve "tête-à-tête" (*gözden göze*) kelimeleri metinde `chef-d’\oe{}uvre` ve `t\^{e}te-\‘{a}-t\^{e}te` gibi

yazılmalıdır.

Tablo 2.7: T1 şifreli Avrupa alfa-sayısalın önemli simgeleri

\dh	ð	\dj	đ	\ng	ŋ	\th	þ
\DH	Ð	\DJ	Đ	\NG	Ŋ	\TH	Þ

2.14.3 Tırnak işareti

İngilizce'de ‘ ... ’ ve " ... " gibi tek ve çift tırnak işaretleri; Fransızca'da « ... » ve "< ... "> gibi tırnak işaretleri; Almanca'da ‚‚ ... “ ve "‘ ... "’ gibi tırnak işaretleri kullanılır. Aşağıdaki tabloda standart tırnak işaretini yazdıran komutlar verilmektedir.

Tablo 2.8: Standart tırnak işaretini yazdıran komutlar

"{\dots}"	" ... "	‘{\dots}‘	‘ ... ‘	‚‚{\dots}‘‘	‚‚ ... “
‘{\dots}’	‘ ... ’	‘‘{\dots}’’	“ ... ”	‚‚{\dots}‚‚	‚‚ ... ‚‚
’{\dots}’	’ ... ’	‘‘{\dots}‘‘	“ ... “	"‘{\dots}"’	"‘ ... "’
’’{\dots}’’	” ... ”	‚‚{\dots}’’	‚‚ ... ”	¨{\dots}¨	¨ ... ¨
<<{\dots}>>	« ... »	"<{\dots}">	"< ... ">		

Avrupa devletlerin T1 şifreli çeşitli tırnak işaretleri aşağıdaki komutlarla yazdırılır.

Tablo 2.9: T1 şifreli tırnak işaretleri

\guillemotleft	«	\guillemotright	»
\guilsinglleft	‹	\guilsinglright	›
\quotedblbase	„	\quotesinglbase	,
\textquotedbl	"		

fontenc paketi T1 seçeneğiyle desteklenmesi gerekir.

2.14.4 Ek simgeler

Aşağıdaki tabloda LaTeX 'in bazı *özgü* simgeleri verilmektedir.

Tablo 2.10: Özgü simgeler

`\dag`	†	`\S`	§	`\copyright`	©	`\pounds`	£
`\ddag`	‡	`\P`	¶	`\textcircled{a}`	ⓐ	`\dots`	...

Metinde bir matematik bloğu kullanılmadan bazı matematiksel simgeler yazdırılabilir. Tablo 2.11 'de bu simgeleri yazdıran komutlar verilmektedir.

Tablo 2.11: Metinde matematiksel simgeler

`\textasteriskcentered`	*	`\textless`	<	`\textgreater`	>
`\textperiodcentered`	·	`\textbar`	|	`\textbullet`	•

2.14.5 textcomp paketi

Bir komutun etki dairesinde bir matematik bloğu kullanılmadan bir matematiksel simge yazdırılabilir. Bunun için belgenin başlık kısmında ek olarak textcomp paketi de tanıtılmalıdır. Bu durumda, tablo 2.11 'deki komutlarla bazı matematik simgeler yazdırılabilir.

Tablo 2.12: textcomp paketinin matematiksel simgeleri

`\textminus`	−	`\textpm`	±	`\textleftarrow`	←
`\texttimes`	×	`\textdiv`	÷	`\textrightarrow`	→
`\textuparrow`	↑	`\textdownarrow`	↓		

\textcelsius	℃	\textdegree	°	\textohm	Ω
\textonehalf	½	\textonequarter	¼	\textthreequarters	¾
\textonesuperior	¹	\texttwosuperior	²	\textthreesuperior	³

Bölüm 3

LaTeX 'de matematiksel formüller

3.1 Matematik blokları

Metinde bir matematik blok TeX 'in özel komutlu paranteziyle ayarlanır. Bir matematik bloğunda

- simgeler arası normal boşluklar TeX tarafından iptal edilerek bu boşluklar duruma uygun olarak yeniden özel olarak koyulur.
- her harf bir değişken adı olarak anlaşılır ve *italik* yazısıyla yazdırılır.
- ^ ve _ yardımcı simgeleri sırasıyla üst ve alt indis yazdırılması için kullanılır (*bkz.*: 2.14.1 ve 3.5.1. paragraflar, 103 ve 130. sayfalar):

```
$a^n + b^{n+m} < x_{12}^2 + y_{p^2_1}^2$
```

$$a^n + b^{n+m} < x_{12}^2 + y_{p_1^2}^2$$

- matematiksel simgeler ve Yunanca harfler belli komutlarla yazdırılır. Bu komutların adı, genelde, simge veya harf adıyla aynıdır (*bkz.* tablo 3.5, 3.6, 3.7, 3.12, ...).
- Latince'de olmayan Türkçe harfler Tablo 2.4 'deki gibi yazılmalıdır.

3.1.1 Metin içinde formül

Metin içinde bir matematiksel formül iki $ dolar veya \(ve \) komutları arasına veya aşağıdaki komutlu parantez içine alınmalıdır:

```
\begin{math}   ...   \end{math}
```

Bir paragrafın tümü bir matematik bloğuna aitse, $ ... $ ve \(... \) komutları yerine komutlu \begin{math} ... \end{math} parantezin kullanılması önerilmektedir. Aşağıda $ ... $ ve \(... \) komutlarıyla yapılan matematik bloğuna bir örnek verilmektedir

```
Bir dik üçgenin $a$ ve $b$ dik kenar
kareleri toplamı $c$ hipotenüs kare-
sine eşittir: \(a^2+b^2=c^2\).
```

Bir dik üçgenin a ve b dik kenar kareleri toplamı c hipotenüs karesine eşittir: $a^2 + b^2 = c^2$.

NOT: 1. Eğer metinde bir $ simgesi varsa, onu kapatıcı yine bir $ simgesi, ve bir \(simgesi varsa, onu kapatıcı bir \) simgesi de varolmalıdır.

2. $ ve \(...\) komutları dayanıksızdır, dolayısıyla, hareketli argümanlarda bu komutlar \protect (*bkz.* 1.7) ile korunmalıdır:

```
\section{\protect\(sinx\protect\),\protect $cosx\protect $ fonksiyonu}
```

Bir formül içinde bir kelime veya simge normal metine aitse, o *math* bloğu dışına alınmalıdır. Bu durumda, formül ve kelime (veya simge) arasına uygun bir boşluk koyulur ve kelime normal metin yazısıyla yazdırılır:

```
Açıkça \(x_k < y_k\) 'dir, bu-
rada $k=1$,~2,$\dots\,$,~$n$
```

Açıkça $x_k < y_k$ 'dir, burada $k = 1, 2, \ldots, n$

Örnekte \dots komutundan sonra \, komutuyla bir ek boşluk koyulmaktadır. Aksi halde, üç noktadan hemen sonra virgül yazdırılır. Tablo 3.29 'de harf veya simgeler arası boşluğu tanımlayan komutlar verilmiştir.

LaTeX bir formülden önce ve sonra belli bir boşluk koymaktadır. Bu boşluk \mathsurround komutun değerine eşit olarak arttırılabilir. Yeni bir değer verilmezse, bu komutun değeri sıfır olarak alınır. Örneğin

```
\setlength{\mathsurround}{2pt}
```

tanıtımından sonra metindeki bir formülün sol ve sağına `2pt` değerli bir ek boşluk koyulacaktır.

Eğer bir formül satır başına düşse, formülün solundaki boşluk; formül satır sonuna düşse, formülün sağındaki boşluk iptal edilir.

Bir formül geçerli satıra sığmadığı taktirde TEX formülü ikiye bölüp, sığmayan ikinci parçasını yeni satıra yazdırır. Bu durumda, bir formül sadece mantıksal ilişki ($=$, $\neq$, $\leq$, $>$, ... ; *bkz.* Tablo:3.4) veya işlecli ($+$, $-$, $\cdot$, ... ; *bkz.* Tablo:3.2) yerinden bölünür. Bölünen yerdeki mantıksal işlem (veya ilişki) formülün birinci parçasının son simgesi olarak yazdırılır ve o yeni satır başına ikinci defa yazdırılmamaktadır. Bir formül mantıksal işlemi (veya ilişkisi) olmayan yerinden bölünmez. Dolayısıyla, parçalanamaz uzun formüllerden mümkün olduğunca kaçınılmalıdır.

Bir kelimeyi parçalanabilir yerleri `\-` komutuyla işaretlendiği gibi (*bkz.* "TEX 'de sözcük hecelemesi" adlı paragraf, 49.sayfa), bir formülün de parçalanabilir yerleri `\allowbreak` komutuyla işaretlenebilir:

```
Alınan vektör şöyledir: $(x_1,
\ldots,x_m,\allowbreak y_1,
{\ldots},y_n)$
```

Alınan vektör şöyledir: $(x_1, \ldots, x_m, y_1, \ldots, y_n)$

`\allowbreak` komutu, genelde, ikili ilişki ve işlemleri çok az veya hiç olmayan uzun formüllerde kullanılır. Tabii ki, eğer formül satır içine yerleşebiliyorsa, `\allowbreak` komutu formülü hiçbir yerinden bölmez:

```
$(x_1,\ldots,x_m,\allowbreak y_1,\ldots,y_n)$
```

$(x_1, \ldots, x_m, y_1, \ldots, y_n)$

Bir formülün parçalanabilir çarpım işlem yerleri özel olarak `\*` komutuyla işaretlenebilir. Eğer formül böyle bir yerinden bölünürse, TEX formülün birinci parçasın sonuna "$\times$" simgesini yazdırır:

```
\* komutuna bir örnek: $(a+b)\*(c+d)$
```

* komutuna bir örnek: $(a+b)\times (c+d)$

Eğer formül bir satıra yerleşebiliyorsa, "$\times$" simgesi yazdırılmamaktadır:

```
bir satırda şöyledir: $(a+b)\*(c+d)$
```

bir satırda şöyledir: $(a+b)(c+d)$.

Yukarıdaki komutlara ters olarak, formülün bir parçası küme parantezi içine `{ ... }` olarak alınırsa, TEX formülün bu parçasının bir yerinden bölünmesine (hatta =, +, ... işlemleri varsa da) izin vermeyecektir.

Bir formülün ikili işlem ve ilişki yerinden bölünme olasılığı TEX 'in sırasıyla `\binoppenalty` ve `\relpenalty` komutlarıyla azaltılabilir veya arttırılabilir. Yeni bir değer verilmezse, TEX bu komutlara sırasıyla `700` ve `500` değeri atar. Aslında `\binoppenalty` ve `\relpenalty` komutlarının değeri TEX 'in bir formülü sırasıyla ikili işlem ve ilişki yerinden böldüğünde alacak cezasını belirlemektedir. Çünkü, ceza ne kadar büyükse (yani bu komutların değeri artarsa), formülün ikili işlem ve ilişki yerinden bölünme olasılığı azalır. Örneğin, belgenin başlik kısmına

```
\binoppenalty=1000
```

tanıtımı yazılırsa [1], bir formülün ikili işlem yerinden bölünme olasılığı daha da azalıp, TEX geçerli satırı mümkün olduğunca metinli yerinden bölmeye çalışır. Komutların `10000` değeri TEX 'e bir formülün hiçbir yerinde bölünmesine izin vermemektedir. Tersine, komutların sıfır değerinde TEX bir formülü bölmeye ihtiyaç duyduğu durumlarda satır sonunda ilk karşılanan ikili işlem veya ilişki yerinden formülü ikiye bölecektir.

3.1.2 Metinde uzun satırların engellenmesi

Bir formül (veya metin) satıra sığmadığı, yani satırın sağ boşluğuna taştığı durumunu ele alalım. Bu durumda TEX formülün tümünü yeni satıra alacak ya da formülü yeni satıra almadan geçerli satırı daha da uzun oluşturur. Her iki durum da iyi değildir, çünkü, eğer formülün tümü yeni satıra alınırsa, geçerli satır dolmadığından dolayı satırdaki kelimeler arasına daha da büyük boşluklar koyulur:

[1] bu komutların değeri sadece pozitif tam sayı olabilir

```
Satırda aşağıdaki formülün
tümü yeni satıra alınmakta-
dır $(a^2+b^2)(c^2+d^2)$
```

formulün tümü yeni satıra alınmaktadır $(a^2+b^2)(c^2+d^2)$

eğer formül yeni satıra alınmazsa, onun bir kısmı sayfanın sağ boşluğuna çıkacaktır:

```
aşağıdaki formül yeni satıra alın-
madığından geçerli satır daha da
uzundur $(a^2+b^2)(c^2+d^2)$
```

... normal satırın sağı
satır daha da uzundur $(a^2+b^2)(c^2+d^2)$
... normal satırın sağı

Böyle durumda, mümkünse, formülü içeren paragraf yeniden öyle yazılmalı ki sonunda bu formül satırın içine düşsün. Eğer bu mümkün değilse, formülün bölünebilme yerleri `\allowbreak` komutuyla işaretlenir ya da TEX 'in uzun satır oluşturma olasılığı `\tolerance` komutuyla (*bkz.* 2.3.1. paragraf, 47.sayfa) zayıflatılmalıdır.

Bir formülün bölünebilme yerlerini işaretlenmesi yukarıda ele alınmıştır. Şimdi metinde uzun satırları engellenmesini ele alalım. Bunun için yukarıda söylendiği gibi `\tolerance` komutun `200` değeri değiştirilmelidir. Bu komutun değeri arttırılsa, bir satırın sağda kesilmesi zorlaşır, yani satırın gereğinden daha da uzun olma olasılığı zayıflar.

Eğer satıra sığmayan formül paragrafın ortasında veya sonunda ise, TEX formülden önceki satırlardaki kelimeler arasına bir az fazla boşluk vererek formül yerleşen satırı doldurmak için kelimeler toplayacak, dolayısıyla, formülü yeni satıra alınabilecektir. `\tolerance` komutun değeri metin başında

```
\tolerance=500
```

tanıtımıyla tüm metin için değiştirilebilir. Sadece bir paragraf için

```
{
\tolerance=700
        ...
boş satır →

}
```

paragraf →

şeklinde değiştirilmelidir. Bu durumda kapatıcı `}` küme parantezinden önce bir boş satır bırakılmalıdır.

Eğer metinde `\sloppy` (*bkz.* 2.3.1) komutu kullanırsa, satırların sağında kesilebilmesi daha da zorlaşır. Çünkü, bu komut `\tolerance` komutuna en büyük `10000` değerini vermektedir. Dolayısıyla, TeX bir satırı bir sorunla karşılaştığı ilk durumda kesecektir. `\sloppy` komutun etki dairesi `\fussy` (*bkz.* 2.3.1) komutuyla iptal edilir. `\sloppy` komutun etki dairesi `{ ... }` küme paranteziyle de sınırlandırılabilir. Bu durumda kapatıcı `}` parantezinden önce bir boş satır bırakılmalıdır. Bir büyük paragrafta `\sloppy` komutu yerine komutlu

```
\begin{sloppypar}   ...   \end{sloppypar}
```

parantezi kullanılabilir (*bkz.* 2.3.1).

3.2 Metinden ayrılan formüller

Bir paragrafta TeX ayrılan formülü, geçerli satırı durdurup, yeni bir satırdan geçici olarak yazdırır. Dolayısıyla, eğer formülden sonra bir boş satır veya `\par` (*bkz.* 2.3) komutu yoksa, bu formülden sonraki ilk satır satır başı olmaksızın yazdırılır.

3.2.1 Tek satırlı formüller

Ayrılan numarasız bir formül iki çift `$$ ... $$` doları veya `\[ ... \]` komutları arasına veya aşağıdaki komutlu parantez içine alınmalıdır:

```
\begin{displaymath}   ...   \end{displaymath}
```

Bir paragrafın tümü bir matematik bloğuna aitse, `$$ ... $$` ve `\[...\]` komutları yerine komutlu `\begin{displaymath} ... \end{displaymath}` parantezin kullanılması önerilmektedir. Aşağıda `$$ ... $$` komutuyla yapılan matematik bloğuna bir örnek verilmektedir

```
Bir dik üçgenin \(a\) ve $b$ dik kenar
kareleri toplamı \(c\) hipotenüs kare-
sine eşittir: $$a^2+b^2=c^2 .$$
```

Bir dik üçgenin a ve b dik kenar kareleri toplamı c hipotenüs karesine eşittir:

$$a^2 + b^2 = c^2.$$

NOT: Bir metinde çift dolar $$ simgesi varsa, onu kapatıcı yine çift dolar $$ simgesi, ve bir `\[` simgesi varsa, onu kapatıcı bir `\]` simgesi de varolmalıdır.

Örnekte görüldüğü gibi, formülün sonundaki nokta (veya virgül, noktalı virgül, iki nokta) kapatıcı çift dolar $$ veya `\]` simgesinden önce yazılmalıdır, aksi halde, bu nokta formülden sonraki ilk satırın başına yazdırılır.

Ayrılan formül içinde bir yatay boşluk `\quad`, `\qquad`, `\hspace` ve `\hspace*` (*bkz.* 2.2.1, 45.sayfa) komutlarıyla koyulur:

```
\[A_{k-1} + A_{k-2} = A_k, \qquad k>2 .\]
```

$$A_{k-1} + A_{k-2} = A_k, \qquad k > 2 .$$

Eğer matematik bir formüle bir gönderme yapılacaksa, bu formül numaralandırılmalıdır. LaTeX bir formül ve ona göndermeyi otomatik oluşturur. Bunun için bu formül komutlu

```
\begin{equation}   ...   \end{equation}
```

parantezi içine alınmalıdır:

```
Bir dik üçgenin \(a\) ve \(b\)  dik
kenar kareleri toplamı $c$ hipotenüs
karesine eşittir:
\begin{equation}
a^2+b^2=c^2 .
\end{equation}
```

Bir dik üçgenin a ve b dik kenar kareleri toplamı c hipotenüs karesine eşittir:

$$a^2 + b^2 = c^2. \qquad (1)$$

Numaralanan bir formüle metinin herhangi bir yerinden gönderme yapılabilmesi için bu formülü içeren komutlu `\begin{equation}` ... `\end{equation}` parantezin herhangi bir yerine formülün *name* adı `\label{`*name*`}` (*bkz.*

1.10, 21.sayfa) komutuyla işaretlenmelidir. Bundan sonra, `\label{`*name*`}` ile işaretlenen formüle metinin herhangi bir yerinden `\ref{`*name*`}` komutuyla bir gönderme yapılabilir (*bkz.* 1.10, 21.sayfa). Bunu bir örnekle açıklayalım:

```
\begin{equation}\label{eq-sinx}
 \sin(x)=x
\end{equation}
(\ref{eq-sinx}) denklemi sadece
sonsuz küçük $x$'ler için doğrudur.
```

$$\sin(x) = x \tag{3.1}$$

(3.1) denklemi sadece sonsuz küçük x 'ler için doğrudur.

Örnekte görüldüğü gibi `\ref` komutu formül numarasını parantez içine almamaktadır. $\mathcal{AMS}$-LaTeX 'in `amsmath` paketin `\eqref` komutu formül numarasını bir parantez içine alır. Yukarıdaki örnekte `\eqref{eq-sinx}` komutu (3.1) 'i yazdırır.

Standart sınıflarda, genellikle, bir formül numarası sayfanın sağına yazdırılır. Eğer belgenin başlık kısmında `\documentclass` (veya `\documentstyle`) komutun `leqno` seçeneği seçilirse:

```
\documentclass[leqno]{article}
```

metindeki tüm numaralı formüllerin numarası satırın soluna yazdırılır. Öte yandan, `amsmath` paketinin `reqno` seçeneği formül numarasın sayfanın soluna yazdıracak herhangi bir kuralı iptal ederek formül numarasını sayfanın sağına yazdırır.

Bir formül numarasının görüntüsü metinin hangi sınıfa ait olduğuna bağlıdır. Örneğin, `article` sınıfında formüller metinin başından sonuna kadar düz olarak (1), (2), (3), ... gibi numaralandırılır; `book` sınıfında ise formül numarası her bölümde yeniden başlanıp $(\cdot\,,\cdot)$ şeklinde çift sayılıdır, burada birinci sayı bölüm numarası, ikinci sayı ise formülün bu bölümdeki sıra numarasıdır, mesela 7.bölümün dördüncü formülünün numarası (7.4) olur.

Bir formül numarası `equation` değişkeninde saklanır. Bazı paketlerde formül numarasının biçimini özel olarak tanımlayan özel bir komut vardır. Örneğin, `amsmath` paketinin `\numberwithin` komutuyla bir formül numarasının biçimi şöyle değiştirilebilir: belgenin başlık kısmına

```
\numberwithin{equation}{section}
```

tanıtımı yazılırsa, formüller her bölümde bağımsız olarak numaralandırılır

ve formül biçimi (3.8) şeklinde çift sayılı olur. Eğer tanıtımda `section` değişkeni yerine `subsection` değişkeni alınırsa, formül biçimi (3.1.4) şeklinde üç sayılı olur.

amsmath paketinde `equation*` bloğu da çalışır:

```
\begin{equation*}   ...   \end{equation*}
```

`equation*` bloğunda formüle numara verilmemektedir. `equation` bloğuna sadece bir yıldız ekleyerek veya çıkartarak bir formül numarasızlandırılır veya numaralandırılır.

Standart sınıflarda bir formül satıra ortalanır. Eğer belgenin başlık kısmında `\documentclass` (veya `\documentstyle`) komutun `fleqn` seçeneği de tanıtılsa:

```
\documentclass[fleqn]{article}
```

metinde bir formül satırın soluna yaslanacaktır. Bu durumda, satırın sol sınırıyla formül arasındaki uzaklık `\mathindent` komutun değerine eşittir. Bu değer değiştirilmediği taktirde LaTeX bu değeri `2.5em` olarak alır. Bu değer `\setlength` veya `\addtolength` komutuyla değiştirerek formülü satırın sol sınırına istediğimiz kadar yakınlaştıra veya uzaklaştırabiliriz.

3.2.2 Denklem sistemleri

Bir denklem sistemi LaTeX 'in komutlu

```
\begin{eqnarray}    ...   \end{eqnarray}
\begin{eqnarray*}   ...   \end{eqnarray*}
```

parantezlerin herhangi biriyle oluşturulur. `eqnarray*` bloğunda denklem sistemi numaralandırılmamaktadır.

`eqnarray` (veya `eqnarray*`) bloğunda:

> değişik satırlara yerleşen iki denklem kendi arasında `\\` komutuyla ayrılır;
>
> bir satır zorunlu olmayan *üç* kısımdan oluşur. Bu kısımlar kendi arasında & simgesiyle ayırılırlar. Her kısım kendi sütununu oluşturur. Sol sütun formülü sağa, orta sütun formülü ortaya, sağ sütun formülü ise sola yaslanır;

her satır ayrıca numaralandırılır. Bir denklem numarası `\nonumber` komutuyla iptal edilebilir;

son satıra \\ komutu yazılmayabilir, aksi halde, bu satıra ek olarak `\nonumber` komutu da yazılmalıdır.

`eqnarray` ve `eqnarray*` bloklarını birkaç örnekle gösterelim.

```
\begin{eqnarray}
         2x_3 & = & 12 \\
      x_2-x_3 & = & 0  \nonumber\\
4x_1-3x_2+x_3 & = & 415\\ \nonumber
\end{eqnarray}
```

$$
\begin{aligned}
2x_3 &= 12 && (3.3)\\
x_2 - x_3 &= 0 &&\\
4x_1 - 3x_2 + x_3 &= 415 && (3.4)
\end{aligned}
$$

```
\begin{eqnarray*}
         2x_3 & = & 12  \\
      x_2-x_3 & = & 0   \\
4x_1-3x_2+x_3 & = & 415
\end{eqnarray*}
```

$$
\begin{aligned}
2x_3 &= 12\\
x_2 - x_3 &= 0\\
4x_1 - 3x_2 + x_3 &= 415
\end{aligned}
$$

```
\begin{eqnarray}
   x^2+2z^3 & < & a^2+b^2\nonumber\\
     2x-y^2 & = & a^4   \\
x^3-y^2+z^2 & > & b^3-1 \nonumber
\end{eqnarray}
```

$$
\begin{aligned}
x^2 + 2z^3 &< a^2 + b^2 &&\\
2x - y^2 &= a^4 && (3.5)\\
x^3 - y^2 + z^2 &> b^3 - 1 &&
\end{aligned}
$$

`eqnarray` (veya `eqnarray*`) bloğunda iki komşu satır arasındaki düşey aralık `\jot` komutunda saklanır. Komutun değeri değiştirilmediği taktirde LaTeX bu değeri `3pt` olarak alır. Aslında bir satırdan sonra koyulacak düşey aralık `\\[...]` komutuyla da değiştirilebilir (*bkz.* 2.3.5, 50.sayfa).

Bir formül numarası için `equation` ve `eqnarray` blokları aynı *equation* değişkenini kullanır. Dolayısıyla, her matematik bloğunda *equation* değişkenin geçerli numarası yazdırılmaktadır.

Yukarıda LaTeX 'in `equation` ve `eqnarray` matematik blokları ele alındı. Şimdi $\mathcal{AMS}$-LaTeX 'in matematik bloklarını ele alalım.

$\mathcal{AMS}$-LaTeX 'in `amsmath` paketinde [2] `eqnarray` bloğun daha da basit

[2] Aslında, `amsmath` paketiyle metine $\mathcal{AMS}$-LaTeX 'in aşağı yukarı tümü yüklenmiş olur. Çünkü bu paket $\mathcal{AMS}$-LaTeX 'in tüm önemli matematik komut ve bloklarını içermektedir.

`gather` şekli vardır:

```
\begin{gather}    ...   \end{gather}
```

`gather` bloğun `equation` bloğundan farkı şu ki `gather` bloğu birkaç satırlı formülü içerebilir. Bu blokta her satır ayrıca numaralandırılır. Bu numara `\nonumber` komutuyla iptal edilebilir. Tüm satırlar numaralandırılmaması için yıldızlı `gather*` bloğu kullanılmalıdır:

```
\begin{gather*}    ...   \end{gather*}
```

Aşağıda `gather` ve `gather*` bloklarına birer tane örnek verelim.

```
\begin{gather}
L_x + L_y = - L_{xy} \\[2mm]
- L_x - L_z > 0 \nonumber \\
100 < 3L_x - L_y + 2L_z
\end{gather}
```

$$L_x + L_y = -L_{xy} \tag{3.6}$$
$$-L_x - L_z > 0$$
$$100 < 3L_x - L_y + 2L_z \tag{3.7}$$

```
\begin{gather*}
3A + B < - 2C  \\[2mm]
A > 0
\end{gather*}
```

$$3A + B < -2C$$
$$A > 0$$

`amsmath` paketin aşağıdaki komutlu parantezi `eqnarray` bloğunun bir genişletmesidir

```
\begin{align}    ...    \end{align}
```

`align` bloğunda bir satır (yani matematik formül) & simgesiyle birkaç sütuna bölünebilir ve satır sonu \\ ile gösterilir. Bir satırdaki birinci, üçüncü, ... & simgesi geçerli sütunda düşey yaslanma noktasını tanımlar; ikinci, dördüncü, ... & simgesi ise sütunların ayırılmasını tanımlar:

```
\begin{align}
a    &=b   &x    &=y-5\nonumber\\
1+a^2&=3b^2&x^2-x&=y^2
\end{align}
```

$$\begin{aligned} a &= b & x &= y - 5 \\ 1 + a^2 &= 3b^2 & x^2 - x &= y^2 \end{aligned} \tag{3.8}$$

Tek numaralı & simgesi sol ve sağ sütunu "birleştirerek" bir tek matematik bloğunu (örnekteki denklem) oluşturmaktadır; çift numaradaki &

simgesi ise sol ve sağ sütunu bir-birinden "ayırarak" bunlar arasına bir boşluk koymaktadır. $\mathcal{AMS}$-LaTeX bu boşlukları, aynı zamanda birinci sütundan önce ve son sütundan sonraki boşluğu da kendi aralarında eşit olarak ayarlar. Dolayısıyla, `align` bloğunda her satırı birkaç denklemden oluşan bir matematik formülü yazılabilir.

`align` bloğunda satırlar numaralandırılmaması için bu bloğun yıldızlı `align*` şekli kullanılmalıdır:

```
\begin{align*}    ...    \end{align*}
```

amsmath paketinde `align` bloğun benzeri olan `alignat` bloğu vardır:

```
\begin{alignat}    ...    \end{alignat}
```

Bu blok sadece birinci sütundan önce ve son sütundan sonra koyulan boşlukları eşitler, iki sütun arasındaki boşluk ise boşlukları tanımlayan komutlarla elle koyulmalıdır. `alignat` bloğunda tek ve çift numaralı & simgesinin görevi de aynen `align` bloğundaki gibidir.

Öte yandan, `alignat` bloğu `tabular` bloğuna da çok benzerdir (*bkz.* 2.10.1. paragraf, 74.sayfa). Çünkü `alignat` bloğunun, `tabular` bloğu gibi, sütunların sayısını tanımlayan zorunlu argümanı vardır. Dolayısıyla, bir satırda & simgesinin sayısı gösterilen sütunlar sayısından büyük olmamalıdır:

```
\begin{alignat}{2}
a    &=b   &\quad  x&=y-5\\
1+a^2&=3b^2&   x^2-x&=y^2
\end{alignat}
```

$$a = b \qquad x = y - 5 \tag{3.9}$$
$$1 + a^2 = 3b^2 \quad x^2 - x = y^2 \tag{3.10}$$

`alignat` bloğunda satırlar numaralandırılmaması için bu bloğun yıldızlı `alignat*` şekli kullanılmalıdır:

```
\begin{alignat*}    ...    \end{alignat*}
```

amsmath paketinde `align` bloğunun yine benzeri olan `flalign` bloğu da vardır:

```
\begin{flalign}    ...    \end{flalign}
```

`flalign` bloğunda `align` bloğundaki gibi sütunlar arası boşluklar aynıdır,

fakat birinci sütundan önce ve son sütundan sonra boşluk koyulmamaktadır. Birinci sütun sayfa soluna, son sütun ise sağına (formül numarasına) yaslanır:

```
\begin{flalign}
a  &=b   &    x&=y-5& x&>0\\
1+a&=3b^2&x^2-x&=y^2& x&<0
\end{flalign}
```

$$a = b \qquad x = y - 5 \qquad x > 0 \quad (3.11)$$
$$1 + a = 3b^2 \qquad x^2 - x = y^2 \qquad x < 0 \quad (3.12)$$

`flalign` bloğunda satırlar numaralandırılmaması için bu bloğun yıldızlı `flalign*` şekli kullanılmalıdır:

```
\begin{flalign*}    ...    \end{flalign*}
```

Yukarıdaki örneği `flalign*` bloğunda ele alalım:

```
\begin{flalign*}
a  &=b   &    x&=y-5& x&>0\\
1+a&=3b^2&x^2-x&=y^2& x&<0
\end{flalign*}
```

$$a = b \qquad x = y - 5 \qquad x > 0$$
$$1 + a = 3b^2 \qquad x^2 - x = y^2 \qquad x < 0$$

$\mathcal{AMS}$-LaTeX 'de bir formül numarası `\nonumber` komutuyla iptal edildiği gibi `\notag` komutuyla da iptal edilebilir.

3.2.3 Uzun formüllerin bölünmesi

Metinde iki $ dolar veya `\( . \)` arasına alınan bir formül satıra sığmasa, TeX bu formülü ikiye bölüp, ikinci parçasını yeni satıra alışı ve, bu durumda, formülün parçalanabilme yerlerin `\allowbreak` komutuyla işaretlenebilmesi 3.1.1. paragrafında ele alınmıştır (*bkz.* 111.sayfa). Bu paragrafta, bir satıra sığmayan ayırtılan formülleri ele alacağız.

TeX bir ayırtılan formülü hiç bölmeden bir satıra yazdırır. Dolayısıyla, ayırtılan bir formül geçerli satıra sığmasa, onun bir kısmı satırın sağ boşluğuna çıkmaktadır. Böyle durumda, TeX bu hakkında `Overfull...` yazılımıyla bir bilgi yazdırır. Ayırılan formülün bölünebilme yerleri `\allowbreak` gibi birer bir komutla işaretlenemez. Böyle bir formül birkaç satıra elde bölünmelidir. Ayırılan uzun bir formül `eqnarray` (veya `eqnarray*`) bloğunda birkaç kısıma şöyle bölünebilir:

```
\begin{eqnarray*}
&\sum (a_n+b_n)(c_n-d_n) = & \\
&\qquad =\sum a_nc_n-\sum a_nd_n+
         \sum b_nc_n-\sum b_nd_n&
\end{eqnarray*}
```

$$\begin{aligned}\sum(a_n+b_n)(c_n-d_n) &= \\ =\sum a_nc_n-\sum a_nd_n+\sum b_nc_n-\sum b_nd_n&\end{aligned}$$

($\sum$ simgesi için tablo 3.12 'e bakınız). `eqnarray` bloğunda her satır ayrıca numaralandırılır. Fakat formül birkaç satırlı olması rağmen o tektir. Dolayısıyla, `eqnarray` bloğunda bir satır hariç tüm satırlar sonuna `\nonumber` komutu yazılmalıdır. Yukarıdaki örnekte görüldüğü gibi, bir satıra `\quad`, `\qquad` ve `\hspace` komutlarıyla boşluklar koyarak o satır sol veya sağa yaslanabilir. Üstelik bir satır sola veya sağa & simgesiyle de yaslanabilir:

```
\begin{eqnarray}
x - y & = & a + b + c -\nonumber\\
      &   & d - e - f - g .
\end{eqnarray}
```

$$\begin{aligned}x-y &= a+b+c- \\ &\qquad d-e-f-g. \end{aligned} \tag{3.13}$$

Birkaç satırlı bir uzun formül, özel olarak, `amsmath` paketin `multline` bloğunda da yazılabilir:

```
\begin{multline}   ...   \end{multline}
```

`multline` bloğunda bir satır sonu `\\` ile gösterilip, birinci satır sayfa soluna, son satır sayfa sağına ve aradaki satırlar ise sayfanın ortasına yaslanır (`\documentclass` tanıtımın `fleqn` seçeneği hariç):

```
\begin{multline}
A = a_{11}x^2_1 + \\
+a_{12}x_1x_2+\dots+a_{1n}x_1x_n+\dots\\
\dots+a_{n1}x_nx_1+\dots+a_{nn}x_nx_n=\\
= \sum _{i,j=1}^n a_{ij} x_ix_j
\end{multline}
```

$$\begin{aligned}A = a_{11}x_1^2+ \\ +a_{12}x_1x_2+\ldots+a_{1n}x_1x_n+\ldots \\ \ldots+a_{n1}x_nx_1+\ldots+a_{nn}x_nx_n= \\ =\sum_{i,j=1}^{n} a_{ij}x_ix_j\end{aligned} \tag{3.14}$$

Birkaç satırlı bir uzun formül `amsmath` paketin `split` bloğunda da yazılabilir:

```
\begin{split}   ...   \end{split}
```

Fakat `split` bloğun kendisi de başka bir ayırtılan matematik bloğu içine alınmalıdır:

```
\begin{equation}
\begin{split}
\phi = &  \ln\Phi - \cos\psi +\\
       & + \sin\psi\left(\frac{\pi}{2}\\
         + \tan\Phi\right)
\end{split}
\end{equation}
```

$$\begin{aligned}\phi = &\ln\Phi - \cos\psi + \\ &+ \sin\psi\left(\frac{\pi}{2}\right. \\ &\left. + \tan\Phi\right)\end{aligned} \tag{3.15}$$

Örnekte kullanılan `\ln` (ln), `\cos` (cos), `\sin` (sin), `\tan` (tan), `\psi` (ψ), `\phi` (ϕ), `\Phi` (Φ), `\pi` (π), `\frac`, `\left` ve `\right` komut ve simgeleri tablo 3.23 ve 3.7 'de verilmektedir (*bkz.* 157 ve 135.sayfalar). Bir satırdaki formül parçasının yaslanması için & simgesi kullanılır. `amsmath` paketin tanıtımında, paketin seçeneği gösterilmemişse, LaTeX bu seçeneği `centertags` olarak alır. Dolayısıyla, örnekte de görüldüğü gibi, formül numarası onun yüksekliğine göre ortalanır. Eğer `amsmath` paketin `tbtags` ("Top-or-bottom tags") seçeneği alınırsa:

```
\usepackage[tbtags]{amsmath}
```

formül numarası tek numaralı sayfada son satır düzeyinde, çift numaralı sayfada ise birinci satır düzeyinde ayarlanır.

3.2.4 Bir satırda bağımsız denklemler

Bu paragrafta `amsmath` paketinin `aligned`, `gathered` ve `alignedat` blokları ele alınmaktadır:

```
\begin{aligned}    ...  \end{aligned}
\begin{gathered}   ...  \end{gathered}
\begin{alignedat}  ...  \end{alignedat}
```

Bu bloklarda bir formül sırasıyla `align`, `gather` ve `alignat` bloklarındaki kurallara göre ayarlanır. Fakat yeni bloklardaki bir satırda formüle sadece gerekli yer ayrılır. Bu ise bir satıra birkaç bağımsız denklem (yani formül) veya metini yerleştirilmesine izin vermektedir. Bir bağımsız denklem

sisteminin düşey yaslanması bloğun zorunlu olmayan `t`, `c` ve `b` seçenekleriyle tanımlanır. Bloğun seçeneği gösterilmediği taktirde, $\mathcal{AMS}$-LaTeX bu seçeneği `c` olarak alır. Üstelik bu bloklar, `split` bloğu gibi, başka bir ayırtılan matematik bloğu içine alınmalıdır. Böyle bir matematik bloğunda denklem sisteminin parantezi (`[`, `]`, `(`, `)`, `{` ve `}`) `\left` ve `\right` komutlarıyla yazdırılabilir. Bir blokta `\left` ve `\right` komutların biri varsa, diğeri de varolmalıdır. Tek parantezli denklem sisteminde bir parantezin olmadığından dolayı `\left` ve `\right` komutların birinde parantez yerine nokta yazılmalıdır:

`\left\{` *formül* `\righ.` veya `\left.` *formül* `\righ]`.

`aligned` bloğuna birkaç örnek verelim (bu örnekler `gathered` ve `aligned` blokları için benzerdir).

```
\begin{equation*}
\begin{aligned}
      z^2 & = x^2 + y^2\\
x^2 - y^2 & = z^2
\end{aligned}
\qquad \text{Pisagor teoremi}
\end{equation*}
```

$$\begin{aligned} z^2 &= x^2 + y^2 \\ x^2 - y^2 &= z^2 \end{aligned} \qquad \text{Pisagor teoremi}$$

Bir satırda birkaç bağımsız denklem sistemi `\left` ve `\right` komutlarıyla şöyle yazılabilir

Giriş dosyası:

```
\begin{equation*}
\left.
   \begin{aligned}
            z^2 & = x^2 + y^2\\
      x^2 - y^2 & = z^2
   \end{aligned}
\right\} \quad \text{Pisagor teoremi} \quad \left.
   \begin{aligned}
      B' & = -\partial\times E\\
      E' & = -\partial\times B - 4\pi j
   \end{aligned}
\right] \quad \text{Maxwell denklemleri}
\end{equation*}
```

LaTeX 'de görüntüsü:

$$\left.\begin{aligned} z^2 &= x^2 + y^2 \\ x^2 - y^2 &= z^2 \end{aligned}\right\} \text{ Pisagor teoremi} \qquad \left.\begin{aligned} B' &= -\partial \times E \\ E' &= -\partial \times B - 4\pi j \end{aligned}\right] \text{ Maxwell denklemleri}$$

Matris tipli ifadeler de bağımsız formüller sistemi olarak `\left` ve `\right` komutlarıyla şöyle yazılabilir

Giriş dosyası:

```
\begin{equation*}
A=\left(
   \begin{aligned}
                a_{11} & a_{12} + a_{21}\\
      a_{12} - a_{21} & a_{22}
   \end{aligned}
\right) \ \text{matris ve} \ B=\left[
   \begin{aligned}
      b_1 & b_2\\
      b_3 & b_4
   \end{aligned}
\right] = Cx + D \ \text{denklem}
\end{equation*}
```

LaTeX 'de görüntüsü:

$$A = \left.\begin{aligned} a_{11}a_{12} + a_{21} \\ a_{12} - a_{21}a_{22} \end{aligned}\right) \text{ matris ve } B = \left[\begin{matrix} b_1 b_2 \\ b_3 b_4 \end{matrix}\right] = Cx + D \text{ denklem}$$

Aslında matrisler `matrix` bloğunda yazılmalıdır (*bkz.* 3.10.2. paragraf, 165. sayfa), `eqnarray` bloğunda ise `array` ile (*bkz.* 3.10.1. paragraf, 160. sayfa), yazılmalıdır.

NOT: Yukarıda ele alınan tüm matematik bloklarda satırlar arası boşluk `\\` komutun zorunlu olmayan *length* argümanıyla arttırıla veya azalttırılabilir (*bkz.* 2.3.5. paragraf, 50. sayfa):

`\\[2mm]`, `\\[-1.2cm]`, `\\[5pt]`, ...

$\mathcal{AMS}$-LaTeX 'de bir formül tüm satırı örtüyorsa formül numarası bu satıra yazdırılmadan onun altına yazdırılır:

```
\begin{align}
a = 1 + 2 + 3 + 4 + \ldots + n
\end{align}
```

$$a = 1 + 2 + 3 + 4 + \ldots + n \tag{3.16}$$

Böyle durumda formül numarası `\raisetag` komutuyla yukarı veya aşağıya kaydırılabilir. Örneğin, `\raisetag{6pt}` komutu formül numarasını `6pt` 'e yukarı çeker. Fakat bu komutun kullanılması her zaman işimize yararlı olmuyor. Çünkü formül numarası yukarı çekildiğinde formül üstüne çıkabilir:

```
\begin{align}
a = 1+2+3+4+\ldots +n\raisetag{5mm}
\end{align}
```

$$a = 1 + 2 + 3 + 4 + \ldots + n \tag{3.17}$$

$\mathcal{AMS}$-LaTeX formül sağında, az olsa da, boş yer bulabilse, numarayı formül sağına sıkıştırarak şöyle yazdırır:

```
\begin{align}
a = 1+2+3+4\ldots +n\raisetag{5mm}
\end{align}
```

$$a = 1 + 2 + 3 + 4 \ldots + n \tag{3.18}$$

3.2.5 Çok satırlı formülün bölünmesi

`amsmath` paketinin bir komutlu parantezinde çok satırlı bir formül veya denklem sisteminin bir parçası geçerli sayfaya sığmazsa, $\mathcal{AMS}$-LaTeX bu parçayı yeni sayfaya yazdırılmasına izin vermemektedir, yani geçerli sayfada bir formül belli bir yerinden kesilerek yeni sayfadan devam ettirilmemektedir. Eğer metin birkaç denklem sistemlerinden oluşan bir büyük formülü içeriyorsa, bu formül geçerli sayfaya sığmadığından dolayı formülün tümü yeni sayfaya alınır. Oysa geçerli sayfanın dolmamasına sebep olur.

$\mathcal{AMS}$-LaTeX 'in bu özelliği belgenin başlık kısmında `\allowdisplaybreaks` tanıtımıyla tüm matematik bloklar için iptal edilebilir. Bu tanıtımın zorunlu olmayan argümanı `1`, ... ,`4` değerleri alabilir. Argüman değeri büyük oldukça $\mathcal{AMS}$-LaTeX 'in formül kesilmesine izin vermemesi de zayıflar. Dolayısıyla, argümanın `4` değeri izin vermemesini tamamen kaldırır, yani her zaman izin verir demektir; argümanın `1` değerinde ise

formülün mümkün olduğunca kesilmemesi önerilmektedir. Argüman değeri gösterilmediği taktirde $\mathcal{AMS}$-LaTeX bu değeri `4`, yani `\allowdisplaybreaks[4]` olarak alır.

Eğer `\allowdisplaybreaks` tanıtımıyla tüm matematik bloklarda bir formülün gerekli durumlarda kesilmesine izin verilmişse, bir matematik bloğunda bu kuralı kaldırılması (yani gerekli durumda izin verilmemesi) için bir satırda `\\` komutu yerine `\\*` komutu kullanılmalıdır (*bkz.* 2.3.5. paragraf, 50. sayfa).

Üstelik, `\allowdisplaybreaks` komutu kullanılmadan da bir formül kesilmesine sadece yerel bir yerde izin verilebilir. Bunun için bir formülün olası kesilebilme satırı sonuna `\\` komutundan önce `\displaybreak` komutu yazılmalıdır. `\displaybreak` komutunun da zorunlu olmayan argümanı `0`, ... ,`4` değerleri alır. Argüman değeri büyük oldukça, formülün kesilme olasılığı da artar; argümanın `0` değeri formülün kesilmesine izin verir, fakat onu zorlamayacaktır. Tersine, `\displaybreak[4]` ise kesilmesini zorlaştıracaktır. Argüman değeri gösterilmediği taktirde $\mathcal{AMS}$-LaTeX bu değeri `4` olarak alır.

NOT: Birkaç satırlı formülün kesilmesini zorlayacak komutlar `split`, `aligned`, `gathered` ve `alignedat` bloklarında geçerli değildir.

3.3 Simgeler arasındaki boşluk

TeX bir matematik blokta simgeler arası boşluğu kendisi ayarlar. Bir boşluğun büyük veya küçüklüğü simgelerin hangi tipe ait olduğuna bağlıdır:

```
$a-3.1=b \qquad (+a)-3,1{=}b$
```

$$a-3.1=b \qquad (+a)-3,1{=}b$$

TeX birinci denklemde `"-"` ve `"="` simgelerini sırasıyla ikili işlem ve ilişki olarak anlar. Dolayısıyla, bu simgelerin soluna ve sağına gerekli bir boşluk verilmektedir. İkinci denklemde ise, bu boşluklar verilmemektedir. Çünkü, TeX a simgesin önündeki `"+"` simgeyi bir ikili işlemi olarak anlamaz; `"="` simgesi de küme parantezi içine alındığından dolayı onu

da bir ikili ilişkisi olarak anlamaz. Virgülden sonra da bir boşluk verilir, fakat noktadan sonra ise bu boşluk verilmemektedir.

TEX tüm simgeleri sekiz tipe sınıflandırmaktadır. Aşağıda kendi argümanını belli bir sınıfa ait olduğunu ilan eden komutlar verilmektedir:

komut		*argüman*
`\mathord`	-	A, B, ... , 0, Φ, ∞, ...
`\mathopen`	-	$\langle$, $\lceil$, $\lfloor$, $\vert$, ...
`\mathclose`	-	$\rangle$, $\rceil$, $\rfloor$, $\Vert$, ...
`\mathpunct`	-	, ; . ! ? ...
`\mathbin`	-	ikili işlemler: $+$, $-$, $\times$, ...
`\mathrel`	-	ikili ilişkiler: $=$, $>$, $<$, $\not\leq$, $\subset$, ...
`\mathop`	-	matematik işlemler: $\sum$, $\int$, sin , log , ...
`\mathalpha`	-	simgeler alfa nümerik olduğunu ilan edilmektedir. Bu ise bir simgeye 3.11.2. paragrada tanımlanan komutların kullanılabilmesi için gerekmektedir (*bkz.* 170. sayfa).

Nasıl "doğru aralık koyuluşunu" aşağıdaki örnekte `\mathop` komutuyla gösterelim:

```
\( \mathrm{dim} \tilde{\Delta}_m = n \)
```

$$\mathrm{dim}\tilde{\Delta}_m = n$$

Görüldüğü gibi, "dim" ve "$\tilde{\Delta}_m$" arasına bir boşluk gerekmektedir. Tabii ki bu boşluk tablo 3.29 'deki komutlar yardımıyla da koyulabilir (*bkz.* 173. sayfa). Ama bu işi TEX 'in kendisine verelim: açıkça, o "doğru" boşluk yazdırır. Bunun için "dim" ifadesin matematik bloğunda `\mathop` tipli sınıfa ait olduğu şöyle gösterilmelidir:

```
\( \mathop{\mathrm{dim}} \tilde{\Delta}_m = n \)
```

$$\mathrm{dim}\,\tilde{\Delta}_m = n$$

Görüldüğü gibi, TEX "dim" ve "$\tilde{\Delta}_m$" arasına gerekli boşluğu yazdırmak-

tadır.

3.4 Simge boyutu

Bir matematik blokta boyutu üç tipli olan simgeler kullanılabilir:
- metin yazı boyutu;
- indis yazı boyutu;
- indisin indis yazı boyutu.

Eğer metin yazı boyutu `10pt` ise, bu halde indis yazı boyutu `7pt`, her düzeydeki indisin indis yazı boyutu ise `5pt` olarak alınır. Bir formül her parçasın boyutu belli bir kurallara göre ayarlanır. Örneğin, TEX bir kesirin pay ve paydasın boyutunu kendisi seçer. Bir formülün yazı boyutu aşağıdaki tanıtımlarla değiştirilebilir:

`\displaystyle` - ayırtılan formül yazı boyutunu tanımlar; normal metin boyutunda kullanılır.

`\textstyle` - metin içinde ve `array` bloğunda (*bkz.* 3.10.1) formül boyutunu tanımlar; normal metin boyutunda kullanılır.

`\scriptstyle` - birinci düzey indis boyutunu tanımlar; indis yazı boyutunda kullanılır.

`\scriptscriptstyle` - yukarı düzey indis boyutunu tanımlar; indisin indis yazı boyutunda kullanılır.

TEX ayırtılan bir formülü `\displaystyle` yazı boyutuyla; normal metinde bir formülü `\textstyle` yazı boyutuyla ve `\frac`, `^`, `_`, ... gibi komutun argümanı ise bir alt düzey yazı boyutuyla yazdırır:

```
$a + \frac{b}{c}$
$$a + \frac{b}{c}$$
```

$$a + \frac{b}{c}$$
$$a + \frac{b}{c}$$

```
$a + \frac{b^{n+m^2}}{c^2}$
$$a + \frac{b^{n+m^2}}{c^2}$$
```

$$a + \frac{b^{n+m^2}}{c^2}$$
$$a + \frac{b^{n+m^2}}{c^2}$$

3.5 Matematiksel simgeler

3.5.1 Üst-alt indis ve ayraçlar

Bir formülün üst ve alt indisi sırasıyla `^{ }` ve `_{ }` komutların zorunlu argümanlarına yazılmalıdır (*bkz.*: 2.14.1 ve 3.1. paragraflar, 103 ve 109. sayfalar):

... `^{`*simgeler*`}` ... `_{`*simgeler*`}` ...

Eğer bir simge veya harfin hem üst hem de alt indisi varsa, onlar istenilen sırada yazılabilir:

`a^{ghf}_{ij} = a_{ij}^{ghf}` $= a_{ij}^{ghf}$

Alt indisin üst indis altında olmaması için `^` ve `_` simgeleri arasında içi boş olan küme parantezi kullanılmalıdır:

`a^{ghf}{ }_{ij}` $= a^{ghf}{}_{ij}$, `a_{ij}{ }^{ghf}` $= a_{ij}{}^{ghf}$,

`a_{i}{ }^{gh}{ }_{j}{ }^{f}` $= a_{i}{}^{gh}{}_{j}{}^{f}$

$Fe_2^{+2}Cr_2O_4$ tipli bir formül iyi gözükmemektedir. Çünkü formülün alt indisleri türlü düzeylerdedir. İndekslerin aynı düzeyde yazdırılması 3.12.4. paragrafında ele alınmaktadır (*bkz.* 178.sayfa).

Bir simge veya harfe bir ayraç `\prime` komutu veya `'` simgesiyle yazdırılır:

```
$(fg)''=f''g+2f'g'+fg^{\prime\prime}$
```

$(fg)''=f''g+2f'g'+fg^{\prime\prime}$

3.5.2 Üç nokta

Üç noktanın çeşitli şekilleri aşağıdaki tabloda verilmektedir.

Tablo 3.1: Üç noktalar

`\cdots`	$\cdots$	`\ddots`	$\ddots$	`\ldots`	$\ldots$	`\vdots`	$\vdots$

Bir yatay üç nokta, genelde, ikili işlemlerde `\cdots` komutuyla ($1+2+3+\cdots+n$), diğer durumlarda `\ldots` komutuyla ($1,2,3,\ldots,n$) yazdırılır.

amsmath paketin \dots komutu hangi tipli üç nokta koyuluşunu komuttan öncedeki simgeye göre kendisi ayarlar. \dots komutun, her durumda, üç noktayı yazdıracak aşağıdaki özel komutları vardır:

özel komut	üç nokta tipi
\dotsc	virgülden sonra yazdırır: $a_1, a_2, \ldots, a_n$
\dotsb	ikili işlemden sonra yazdırır: $a_1 + a_2 + \cdots + a_n$
\dotsm	çarpımdan sonra yazdırır: $a_1 a_2 a_3 \cdots$
\dotsi	integraldan sonra yazdırır: $\int_{G_1} \int_{G_2} \cdots$
\dotso	diğer durumlarda

3.5.3 Aritmetik işlem simgeleri

TEX 'in +, -, · gibi ikili işlemleri yukarıda ele alınmıştır. Aşağıdaki tabloda bir ikili işlemi tanımlayan tüm komutlar verilmektedir

Tablo 3.2: Aritmetik işlem simgeleri

$\pm$	\pm	$\mp$	\mp	$\times$	\times	$\div$	\div
$*$	\ast	$\star$	\star	$\circ$	\circ	$\bullet$	\bullet
$\cap$	\cap	$\cup$	\cup	$\uplus$	\uplus	$\cdot$	\cdot
$\sqcap$	\sqcap	$\sqcup$	\sqcup	$\vee$	\vee	$\wedge$	\wedge
$\oplus$	\oplus	$\ominus$	\ominus	$\otimes$	\otimes	$\oslash$	\oslash
$\triangle$	\bigtriangleup	$\odot$	\odot	$\lhd$	\lhd	$\dagger$	\dagger
$\bigtriangledown$	\bigtriangledown	$\bigcirc$	\bigcirc	$\rhd$	\rhd	$\ddagger$	\ddagger
$\triangleleft$	\triangleleft	$\diamond$	\diamond	$\unlhd$	\unlhd	$\setminus$	\setminus
$\triangleright$	\triangleright	$\wr$	\wr	$\unrhd$	\unrhd	$\amalg$	\amalg

Eğer \lhd, \rhd, \unlhd ve \unrhd komutları programda çalışmazsa, belgenin başlık kısmında latexsym paketi tanıtılmalıdır.

$\mathcal{AMS}$-LaTeX 'in amssymb paketi aşağıdaki özel ikili işlemleri tanımlamaktadır

Tablo 3.3: $\mathcal{AMS}$'ın aritmetik işlem simgeleri (amssymb paketi)

$\dotplus$ \dotplus	$\ltimes$ \ltimes	$\smallsetminus$ \smallsetminus
$\rtimes$ \rtimes	$\barwedge$ \barwedge	$\curlywedge$ \curlywedge
$\veebar$ \veebar	$\curlyvee$ \curlyvee	$\doublebarwedge$ \doublebarwedge
$\Cap$ \Cap,\doublecap	$\leftthreetimes$ \leftthreetimes	$\Cup$ \Cup,\doublecup
$\rightthreetimes$ \rightthreetimes	$\boxtimes$ \boxtimes	$\circledast$ \circledast
$\boxminus$ \boxminus	$\circleddash$ \circleddash	$\boxplus$ \boxplus
$\centerdot$ \centerdot	$\boxdot$ \boxdot	$\circledcirc$ \circledcirc
$\divideontimes$ \divideontimes	$\intercal$ \intercal	

3.5.4 Mantıksal ilişki simgeleri

Aşağıdaki tablo 3.4'de ikili ilişki komutları verilmektedir.

Tablo 3.4: Mantıksal ilişki simgeleri

$\leq$ \leq	$\geq$ \geq	$\ll$ \ll	$\gg$ \gg
$\equiv$ \equiv	$\asymp$ \asymp	$\neq$ \neq	$\doteq$ \doteq
$\subset$ \subset	$\supset$ \supset	$\subseteq$ \subseteq	$\supseteq$ \supseteq
$\sqsubset$ \sqsubset	$\sqsupset$ \sqsupset	$\sqsubseteq$ \sqsubseteq	$\sqsupseteq$ \sqsupseteq
$\models$ \models	$\perp$ \perp	$\mid$ \mid	$\parallel$ \parallel
$\prec$ \prec	$\succ$ \succ	$\preceq$ \preceq	$\succeq$ \succeq
$\sim$ \sim	$\simeq$ \simeq	$\approx$ \approx	$\cong$ \cong
$\bowtie$ \bowtie	$\Join$ \Join	$\smile$ \smile	$\frown$ \frown
$\in$ \in	$\ni$ \ni	$\vdash$ \vdash	$\dashv$ \dashv
$\propto$ \propto			

Eğer \Join, \sqsubset ve \sqsupset komutları programda çalışmazsa, belgenin başlık kısmında latexsym paketi tanıtılmalıdır.

NOT: Tablo 3.3 ve 3.4 'deki ikili işlem ve ilişkiler sadece bir matematik blokta geçerlidir: `$\leq$`, `$\sqsubseteq$`, `$\doublebarwedge$`, ...

$\mathcal{AMS}$-LaTeX 'in amssymb paketi aşağıdaki özel ikili ilişkileri tanımlar

Tablo 3.5: $\mathcal{AMS}$'ın mantıksal ilişki simgeleri (amssymb paketi)

$\leqq$ \leqq	$\geqq$ \geqq	$\leqslant$ \leqslant
$\geqslant$ \geqslant	$\eqslantless$ \eqslantless	$\eqslantgtr$ \eqslantgtr
$\lesssim$ \lesssim	$\gtrsim$ \gtrsim	$\lessapprox$ \lessapprox
$\gtrapprox$ \gtrapprox	$\approxeq$ \approxeq	$\lessdot$ \lessdot
$\gtrdot$ \gtrdot	$\lll$ \lll,\llless	$\ggg$ \ggg,\gggtr
$\lessgtr$ \lessgtr	$\gtrless$ \gtrless	$\lesseqgtr$ \lesseqgtr
$\gtreqless$ \gtreqless	$\lesseqqgtr$ \lesseqqgtr	$\gtreqqless$ \gtreqqless
$\preccurlyeq$ \preccurlyeq	$\succcurlyeq$ \succcurlyeq	$\curlyeqprec$ \curlyeqprec
$\curlyeqsucc$ \curlyeqsucc	$\precsim$ \precsim	$\succsim$ \succsim
$\precapprox$ \precapprox	$\succapprox$ \succapprox	$\subseteqq$ \subseteqq
$\supseteqq$ \supseteqq	$\Subset$ \Subset	$\Supset$ \Supset
$\sqsubset$ \sqsubset	$\sqsupset$ \sqsupset	$\backsim$ \backsim
$\thicksim$ \thicksim	$\backsimeq$ \backsimeq	$\thickapprox$ \thickapprox
$\doteqdot$ \doteqdot,\Doteq	$\eqcirc$ \eqcirc	$\risingdotseq$ \risingdotseq
$\circeq$ \circeq	$\fallingdotseq$ \fallingdotseq	$\triangleq$ \triangleq
$\vartriangleleft$ \vartriangleleft	$\vartriangleright$ \vartriangleright	$\trianglelefteq$ \trianglelefteq
$\trianglerighteq$ \trianglerighteq	$\vDash$ \vDash	$\Vdash$ \Vdash
$\Vvdash$ \Vvdash	$\smallsmile$ \smallsmile	$\smallfrown$ \smallfrown
$\shortmid$ \shortmid	$\shortparallel$ \shortparallel	$\bumpeq$ \bumpeq
$\Bumpeq$ \Bumpeq	$\between$ \between	$\pitchfork$ \pitchfork

Bir ikili ilişkinin karşıtı `\not` komutuyla şöyle oluşturulabilir:

```
$A\not=B, A\not\subsetB, A\not\supsetB$
$A\not\geqB, A\not\leqB, A\not\precsim B$
```

$A \neq B$, $A \not\subset B$, $A \not\supset B$
$A \not\geq B$, $A \not\leq B$, $A \not\precsim B$

Üstelik amssymb paketinin de karşıtı ikili ilişkileri tanımlayan komutları vardır:

Tablo 3.6: $\mathcal{AMS}$'ın karşıt mantıksal ilişki simgeleri (amssymb paketi)

$\nsim$ \nsim	$\ncong$ \ncong	$\nless$ \nless
$\ngtr$ \ngtr	$\nleq$ \nleq	$\ngeq$ \ngeq
$\nleqslant$ \nleqslant	$\ngeqslant$ \ngeqslant	$\nleqq$ \nleqq
$\ngeqq$ \ngeqq	$\lneq$ \lneq	$\gneq$ \gneq
$\lneqq$ \lneqq	$\gneqq$ \gneqq	$\lvertneqq$ \lvertneqq
$\gvertneqq$ \gvertneqq	$\lnsim$ \lnsim	$\gnsim$ \gnsim
$\lnapprox$ \lnapprox	$\gnapprox$ \gnapprox	$\nprec$ \nprec
$\nsucc$ \nsucc	$\npreceq$ \npreceq	$\nsucceq$ \nsucceq
$\precneqq$ \precneqq	$\succneqq$ \succneqq	$\precnsim$ \precnsim
$\succnsim$ \succnsim	$\precnapprox$ \precnapprox	$\succnapprox$ \succnapprox
$\ntriangleleft$ \ntriangleleft	$\ntriangleright$ \ntriangleright	$\ntrianglelefteq$ \ntrianglelefteq
$\ntrianglerighteq$ \ntrianglerighteq	$\nshortmid$ \nshortmid	$\nmid$ \nmid
$\nshortparallel$ \nshortparallel	$\nparallel$ \nparallel	$\nvdash$ \nvdash
$\nvDash$ \nvDash	$\nVdash$ \nVdash	$\nVDash$ \nVDash
$\nsubseteq$ \nsubseteq	$\nsupseteq$ \nsupseteq	$\nsubseteqq$ \nsubseteqq
$\nsupseteqq$ \nsupseteqq	$\varsubsetneq$ \varsubsetneq	$\varsupsetneq$ \varsupsetneq
$\subsetneq$ \subsetneq	$\supsetneq$ \supsetneq	$\varsubsetneqq$ \varsubsetneqq
$\varsupsetneqq$ \varsupsetneqq	$\subsetneqq$ \subsetneqq	$\supsetneqq$ \supsetneqq

3.5.5 Yunanca harfler

Metinde bir Yunanca harf belli bir komutla yazdırılır. Bir komut adı Yunanca harfin İngilizce adıyla aynıdır. Örneğin, Yunanca α harfi \alpha komutuyla yazdırılır. Ayrıca, Yunanca o (omikron) harfi Latince o harfin italik yazısıyla aynı olduğundan bu harfe bir özel komut verilmemiştir.

Bu harf bir matematik bloğunda Latince `o` harfiyle yazdırılır.

Bazı Yunanca harfler `amssymb` paketi yardımıyla yazdırılır. Örneğin, $\varkappa$ (kappa) harfi `amssymb` paketin `\varkappa` komutuyla yazdırılır.

Aşağıdaki tabloda Yunanca harfleri tanımlayan komutlar verilmektedir.

Tablo 3.7: Yunanca harfler

α	`\alpha`	β	`\beta`	γ	`\gamma`	δ	`\delta`
ϵ	`\epsilon`	ε	`\varepsilon`	ζ	`\zeta`	η	`\eta`
θ	`\theta`	ϑ	`\vartheta`	ι	`\iota`	κ	`\kappa`
λ	`\lambda`	μ	`\mu`	ν	`\nu`	ξ	`\xi`
o	`o`	π	`\pi`	ϖ	`\varpi`	ρ	`\rho`
ϱ	`\varrho`	σ	`\sigma`	ς	`\varsigma`	τ	`\tau`
υ	`\upsilon`	ϕ	`\phi`	φ	`\varphi`	χ	`\chi`
ψ	`\psi`	ω	`\omega`				

Yunanca baş (büyük) harfler komutlarda da baş harflerle başlanır: `\Psi`, `\Phi`, `\Sigma`, ... Bazı Yunanca baş harfler (örneğin, *alfa*, *beta*, ...) Latince'de varolan benzer baş harflerle aynıdır. Dolayısıyla, bu harfler için özel bir komut yoktur. Böyle bir harf yazdırılması için bu harfin Latince'deki benzeri `\mathrm` komutun etki dairesine alınmalıdır:

`$\mathrm{A}$` (A), `$\mathrm{B}$` (B), ...

Latince'de benzeri olmayan Yunanca baş harfler aşağıdaki tabloda verilmektedir.

Tablo 3.8: Yunanca büyük harfler

Γ	`\Gamma`	Δ	`\Delta`	Θ	`\Theta`	Λ	`\Lambda`
Ξ	`\Xi`	Π	`\Pi`	Σ	`\Sigma`	Υ	`\Upsilon`
Φ	`\Phi`	Ψ	`\Psi`	Ω	`\Omega`		

Görüldüğü gibi, küçük harfler italik yazıyla ve baş harfler ise normal yazıyla yazdırılmaktadır. İstenilen bir harf veya simge `\mathrm` ve `\mathit` komutlarıyla sırasıyla `normal` ve *italik* yazıyla yazdırılabilir. Örneğin, `${\mathit\Delta}$` komutun görüntüsü Δ 'dir. Üstelik `amssymb` paketin

aşağıdaki özel komutları bazı baş harfleri *italik* yazıyla yazdırır:

Tablo 3.9: $\mathcal{AMS}$'ın Yunanca harfleri (amssymb paketi)

Γ	\varGamma	Δ	\varDelta	Θ	\varTheta
Λ	\varLambda	Ξ	\varXi	Π	\varPi
Σ	\varSigma	Υ	\varUpsilon	Φ	\varPhi
Ψ	\varPsi	Ω	\varOmega	$\varkappa$	\varkappa
$\digamma$	\digamma				

NOT: Tablo 3.7 - 3.9 'deki Yunanca harfler sadece matematik bloklarda geçerlidir: `$\alpha$`, `$\beta$`, `$\gamma$`, ...

3.5.6 Noktalama işaretleri

Aşağıdaki simgeler TeX 'de noktalı-virgül simgesi olarak kabul edilmiştir. Metinde bu simgelerden sonra bir küçük boşluk koyulur.

Tablo 3.10: Noktalama işaretleri

,	,	;	;	$\cdotp$	\cdotp	:	$\colon$	.	$\ldotp$

`$\colon$` komutuyla yazdırılan iki nokta klavye tuşundaki iki noktadan farklıdır. TeX klavye tuşundaki iki noktayı bir ikili ilişki olarak anlar:

giriş dosyası: `$f:A \to B$ \quad $f\colon A \to B$`

görüntüsü: $f : A \to B$ $f\colon A \to B$.

Görüldüğü gibi, TeX birinci formülün iki noktasını bir ikili ilişki olarak anlamaktadır. Dolayısıyla, onun hem soluna hem de sağına belli bir boşluk koyulmaktadır.

3.5.7 Vurgular

Bir ifade üstüne bir vurgu (rozet) tablo 3.11 'deki komutlarla yazdırılır (bu ifade tabloda kısaca a ile gösterilmektedir). Üç ve dört noktalı vur-

gular amsmath paketin komutlarıyla yazdırılır. Bu komutlar da tablo 3.11 'de verilmektedir.

Tablo 3.11: Matematiksel blokta vurgular

$\acute{a}$	`\acute{a}`	$\check{a}$	`\check{a}`	$\grave{a}$	`\grave{a}`	$\vec{a}$	`\vec{a}`
$\bar{a}$	`\bar{a}`	$\dot{a}$	`\dot{a}`	$\hat{a}$	`\hat{a}`	$\dddot{a}$	`\dddot{a}` [a]
$\breve{a}$	`\breve{a}`	$\ddot{a}$	`\ddot{a}`	$\tilde{a}$	`\tilde{a}`	$\ddddot{a}$	`\ddddot{a}` [a]

[a] amsmath paketin tanıtılması gerekir.

amsmath paketinde tablodaki komutların baş harfleriyle başlanan aynı adlı komutlar da tanımlanan olup, bu komutlar çift vurgularda kullanılmaktadır:

`\Hat{\Vec{a}}`, `\Tilde{\Ddot{a}}`, ...

Gerçi, $\mathcal{AMS}$-LaTeX 'in son sürümünde bu pek fazla fark etmez:

```
$\Hat{\Vec{A}} \qquad \hat{\vec{A}}$
```

$$\hat{\vec{A}} \qquad \hat{\vec{A}}$$

3.5.8 Köklü ifadeler

Bir ifadenin kökü aşağıdaki komutla yazdırılır

`\sqrt[`n`]{`a`}`

`\sqrt` komutunun zorunlu olmayan n argümanı gösterilmediği taktirde LaTeX bu komutun zorunlu a argümanının kare kökünü yazdırır, aksi halde, a argümanının n. kökünü yazdırır:

```
$$\sqrt{a^2}=|a|,\sqrt[3]{a^3}=a$$
```

$$\sqrt{a^2}=|a|,\sqrt[3]{a^3}=a$$

`\sqrt` komutu kök altındaki formüle (yani a argümanına) göre kökün boyutunu ayarlamaktadır:

```
$$\sqrt[7]{\sum f_n(x)},\sqrt[7]{a}$$
```

$$\sqrt[7]{\sum f_n(x)}, \quad \sqrt[7]{a}$$

Bazen böyle kökler iyi gözükmeyebilir:

```
$\sqrt{x}-\sqrt{y}+\sqrt{d}+\sqrt{g}$
```

$\sqrt{x}-\sqrt{y}+\sqrt{d}+\sqrt{g}$

Görüldüğü gibi, x, y, d ve g elemanları farklı boyutlu olduğundan onları örten köklerin de boyutları farklı olup, aynı düzeyde değildir. Köklerin aynı düzeyde olması için her kök altında `\mathstrut` komutu kullanılmalıdır. Aslında bu komut bir görünmez öğedir, çünkü bu görüntüde gözükmemektedir. Bu öğe birkaç kök altında sadece aynı düzey oluşturmak için yazılabilir. Dolayısıyla, her kök altındaki ifade boyutu aynı olup, tüm kökler aynı düzeyde gözükmektedir:

```
$\sqrt{\mathstrut d}-\sqrt{\mathstrut y}$
```

$\sqrt{\mathstrut d}-\sqrt{\mathstrut y}$

Fakat `\mathstrut` komutu her zaman da yararlı değildir. Çünkü görünmez bu öğenin boyutu (yani yüksekliği) bir sol `(` parantezin yüksekliğiyle aynıdır (tabii ki, eni ise sıfırdır). Bu öğe 3.12.4. paragrafta ele alınacak görünmez simgelerin özel bir durumudur (*bkz.* 177.sayfa).

Bazen bir kökün n argümanı da iyi gözükmeyebilir: $\sqrt[\beta]{a}$. Böyle durumda amsmath paketinin `\leftroot` ve `\uproot` komutlarıyla kökün n argümanı sırasıyla yatay ve düşey yönde sürülebilir:

```
$\sqrt[\beta]{a}\quad
\sqrt[\leftroot{2}\uproot{3}\beta]{a}\quad
\sqrt[\leftroot{4}\uproot{-1}\beta]{a}$
```

$\sqrt[\beta]{a}\quad \sqrt[\leftroot{2}\uproot{3}\beta]{a}\quad \sqrt[\leftroot{4}\uproot{-1}\beta]{a}$

Görüldüğü gibi, `\leftroot{ }` ve `\uproot{ }` komutları `\sqrt` komutunun zorunlu olmayan argümanında kullanılır ve bu komutların kendi argümanları ise negatif sayı da olabilir.

Bir ifadenin kökünü aşağıdaki komut da yazdırır

\root {n} \of {a}

`\root` komutunun zorunlu olmayan n argümanı gösterilmediği taktirde LaTeX bu komutun zorunlu a argümanının kare kökünü yazdırır, aksi halde, a argümanının n. kökünü yazdırır:

```
$\root\of 4=2, \root 3\of 8=2,
\root{17}\of{1}=1$
```

$\root\of 4=2, \root 3\of 8=2, \root{17}\of{1}=1$

`\sqrt` ve `\root` komutları oluşturduğu kökler aynıdır. Dolayısıyla, bu komutların hangisinin kullanması, genelde, fark etmez.

3.5.9 Kesirli ifadeler

Metin içinde bir kesir bazen bölü `/` simgesiyle şöyle yazılır: `$x^2/2$` ($x^2/2$). Normal pay ve paydalı kesirler aşağıdaki komutla yazdırılır

\frac{*numerator*}{*denominator*}

Komutun her iki *numerator* ve *denominator* argümanı zorunludur. *numerator* argümanı kesirin payı ve *denominator* argümanı ise paydasını tanımlar:

```
\[
   \frac{a-1}{b}=\frac ab - \frac 1 b
\]
```

$$\frac{a-1}{b}=\frac ab - \frac 1 b$$

Görüldüğü gibi, eğer kesirin pay ve/veya paydası tek harf veya simgeliyse, komutun argümanı küme parantezi içine alınmayabilir.

TeX bir kesir pay ve paydasının yazı boyutunu kendisi ayarlar. Fakat zincirli kesirlerde bu iyi gözükmemektedir:

```
\[
\frac{1}{\sqrt{2}+\frac{1}{\sqrt{2}+
\frac{1}{\sqrt{2}+\dots }}}
\]
```

$$\frac{1}{\sqrt{2}+\frac{1}{\sqrt{2}+\frac{1}{\sqrt{2}+\dots}}}$$

Böyle durumda her düzeydeki yazı boyutunu göstermekte yarar vardır:

```
$$\frac{1}{\sqrt{2} +
\displaystyle\frac{1}{\sqrt{2}+
\displaystyle\frac{1}{\sqrt{2}+
\cdots}}}$$
```

$$\frac{1}{\sqrt{2}+\displaystyle\frac{1}{\sqrt{2}+\displaystyle\frac{1}{\sqrt{2}+\cdots}}}$$

`\frac` komutundan önce yazılan `\displaystyle` komutu ikinci ve üçüncü kesir paydada olması rağmen onları `displaystyle` yazı boyutuyla yazdırılması için gerekmektedir.

amsmath paketinin `\cfrac` komutu bir kesiri her zaman `displaystyle` yazı boyutuyla yazdırır:

```
$$
\cfrac{1}{\sqrt{2} +
\cfrac{1}{\sqrt{2} + \cdots}}
$$
```

$$\cfrac{1}{\sqrt{2}+\cfrac{1}{\sqrt{2}+\cdots}}$$

`\cfrac` komutunun zorunlu olmayan bir argümanı da vardır. Bu argümanın sadece iki değeri var olup, onun `l` değeri kesirin payını sola, `r` değeri ise sağa yaslar:

```
$$
\cfrac[l]{a^2}{a^2+b^2} +
\cfrac[r]{b^2}{a^2+b^2} = 1
$$
```

$$\cfrac[l]{a^2}{a^2+b^2}+\cfrac[r]{b^2}{a^2+b^2}=1$$

Üstelik amsmath paketinin `\dfrac` (`\displaystyle\frac` komutunun kısaltması) ve `\tfrac` (`\textstyle\frac` komutunun kısaltması) komutları kesir pay ve paydasını sırasıyla `displaystyle` ve `textstyle` yazı

boyutuyla yazdırır. `\dfrac` komutu, genelde, metin içindeki bir formülde, `\tfrac` komutu ise ayırtılan bir formülde kullanılır:

```
$\frac{1}{n}\log_2f(x) \quad
\dfrac{1}{n}\log _2f(x)$
```

$\frac{1}{n}\log_2 f(x) \quad \dfrac{1}{n}\log_2 f(x)$

```
$$
\sqrt{\frac{1}{n}\log _2f(x)}\quad
\sqrt{\tfrac{1}{n}\log _2f(x)}
$$
```

$$\sqrt{\frac{1}{n}\log_2 f(x)} \quad \sqrt{\tfrac{1}{n}\log_2 f(x)}$$

3.5.10 Limitlerle operatörler

Aşağıdaki formülü ele alalım

$$\lim_{n\to\infty} \sum_{k=0}^{n} \frac{1}{2^k} = 2.$$

Toplam $\sum$ simgesi bir operatördür (yani dönüşümdür). Bazen bu operatöre "toplam limiti" de denir; işlemin üst ve alt kısmı ise, genelde, bir *limit* denir. Bir limit üst "`^`" ve/veya alt "`_`" indis yardımıyla yazılır. Dolayısıyla, yukarıdaki formül şöyle yazılabilir:

```
$$\lim_{n \to \infty} \sum_{k=0}^n \frac{1}{2^k} = 2 .$$
```

Burada formülün ayırtılan olması da önemlidir; aksi halde, limitler indis gibi yazdırılır:

```
$\lim_{n\to\infty}\sum_{k=0}^n\frac{1}{2^k}=2$
```

$\lim_{n\to\infty}\sum_{k=0}^{n}\frac{1}{2^k}=2$

Metin içinde $\sum$ simgesin de boyutu değişir. Tablo 3.12 'de limitli operatörlerin sırasıyla ayırtılan ve metin içi formülündeki görüntüleri verilmektedir. Bu operatörlerin indisleri $\sum$ operatörün limiti gibi gözükmektedir.

Tablo 3.12: Limitli operatörler

\bigcap	$\bigcap$	$\cap$	\bigotimes	$\bigotimes$	$\otimes$	\bigwedge	$\bigwedge$	$\wedge$
\bigcup	$\bigcup$	$\cup$	\bigsqcup	$\bigsqcup$	$\sqcup$	\coprod	$\coprod$	$\amalg$
\bigodot	$\bigodot$	$\odot$	\biguplus	$\biguplus$	$\uplus$	\int	$\int$	$\int$
\bigoplus	$\bigoplus$	$\oplus$	\bigvee	$\bigvee$	$\vee$	\oint	$\oint$	$\oint$
\prod	$\prod$	Π	\sum	$\sum$	Σ			

Yukarıda görüldüğü gibi, ayırtılan bir formülde limitli operatörün limiti operatörün (yani simgenin) üstüne ve/veya altına yazdırılır. Bu limitin bir indis gibi yazdırılması için operatör ile limit arasına `\nolimits` komutu yazılmalıdır:

```
$$
\prod\nolimits_{k=1}^n k=n!
$$
```

$$\prod_{k=1}^n k = n!$$

amsmath paketinin `sumlimits` ve `nosumlimits` seçenekleri de var olup, bunun `nosumlimits` seçeneği metindeki tüm limitli operatörlerde (integral operatörü hariç) `\nolimits` komutun kullanmasına denktir. Seçenek gösterilmediği taktirde LaTeX bu seçeneği `sumlimits` olarak alır.

`\nolimits` komutuna karşıt olarak `\limits` komutu kullanılabilir, yani bir limitli operatörün limiti indis gibi değil, operatörün üst ve/veya altına yazdırılması için `\limits` komutu kullanılır:

```
$\prod _{k=1}^n k = n!$
$\prod\limits _{k=1}^n k = n!$
```

$$\prod_{k=1}^n k = n!$$
$$\prod\limits_{k=1}^n k = n!$$

İntegraller

Bir integral simgesi `\int` veya `\oint` komutuyla yazdırılır. Bir integral limiti hem metin içi, hem de ayırtılan formülde bir indis gibi yan tarafa yazdırılır:

```
$\int_a^b f(x)dx , \quad \oint_D fd\gamma$

$$\int_a^b f(x)dx , \quad \oint_D fd\gamma$$
```

$\int_a^b f(x)dx, \quad \oint_D fd\gamma$

$$\int_a^b f(x)dx, \quad \oint_D fd\gamma$$

Eğer integral limiti birkaç harf veya simgeden oluşuyorsa, bu limit küme parantezi içine alınmalıdır:

```
$\int _{-\infty}^{100} f(x)dx , \quad
 \oint _{\partial D} fd\gamma $
```

$\int _{-\infty}^{100} f(x)dx, \quad \oint _{\partial D} fd\gamma$

İntegral limitin integral simgesin üst ve altına yazdırılması için `\int` (veya `\oint`) komutundan hemen sonra `\limits` komutu yazılmalıdır:

```
$$\int\limits_a^b f(x)dx , \quad
  \oint\limits_D fd\gamma $$
```

$$\int\limits_a^b f(x)dx, \quad \oint\limits_D fd\gamma$$

amsmath paketin `intlimits` ve `nointlimits` seçenekleri de var olup, bunun `intlimits` seçeneği metindeki tüm integrallerde `\limits` komutun kullanmasına denktir. Seçenek gösterilmediği taktirde LaTeX bu seçeneği `nointlimits` olarak alır. Örneğin, belgenin başlık kısmında

```
\usepackage[nosumlimits,intlimits]{amsmath}
```

tanıtımı yazılırsa, metindeki tüm limitli operatörlerin limiti indis gibi yan tarafa, tüm integrallerin limiti ise integral simgesin üst ve altına yazdırılır.

amsmath paketin `\iint`, `\iiint` ve `\iiiint` komutları sırasıyla iki, üç ve dört katlı integrali yazdırır:

```
$\iint fdxdy \iiint fdxdydz
 \ \iiiint fdxdydzdt$
```

$$\iint fdxdy \quad \iiint fdxdydz \quad \iiiint fdxdydzdt$$

Çok katlı integral ise amsmath paketin `\idotsint` komutuyla yazdırılır:

```
$$\idotsint\limits_V d\vec{a}$$
```

$$\idotsint\limits_V d\vec{a}$$

Çok satırlı indis

Bir limitli operatörün limiti birkaç satırdan oluşuyor olabilir. Bir limiti birkaç satıra bölmek için LaTeX 'in `\atop` komutu kullanılır. Bu durumda, limitin her satırı, \\ komutu yerine, `\atop` komutuyla kesilmelidir:

```
$$
\sum_{\scriptstyle i=0 \atop
\scriptstyle j=1}^n a_{i,j}$, \quad
\bigcup_{\scriptstyle\sqrt{x^2+y^2}<1
\atop\scriptstyle x,y > 0} \{(x,y)\}
$$
```

$$\sum_{\scriptstyle i=0 \atop \scriptstyle j=1}^n a_{i,j}\ , \quad \bigcup_{\scriptstyle\sqrt{x^2+y^2}<1 \atop\scriptstyle x,y > 0} \{(x,y)\}$$

Burada `\scriptstyle` komutu indisi `scriptstyle` yazı boyutuyla yazdırılması için kullanılmaktadır.

amsmath paketinin `\substack` komutu da birkaç satırlı limitin yazılmasında kullanılabilir. Bu durumda, operatörün limiti `\substack` komutuyla başlanıp, her satır, her zaman olduğu gibi, \\ komutuyla kesilmelidir:

```
$\bigoplus_{\substack{1<i<n \\
                      1<j<m} } P_{i,j}$

$$\sum_{\substack{1<i<n \\
                  1<j<m} } P_{i,j}$$
```

$$\bigoplus_{\substack{1<i<n \\ 1<j<m} } P_{i,j}$$

$$\sum_{\substack{1<i<n \\ 1<j<m} } P_{i,j}$$

Görüldüğü gibi, `\substack` komutu indisi otomatik `scriptstyle` yazı boyutuyla yazdırmaktadır. `\substack` komutu aşağıdaki komutlu paran-

tezine genişletilebilir

```
\begin{subarray}{ }  ...  \end{subarray}
```

Komutun zorunlu argümanında `l`, `c` ve `r` seçeneklerinden biri alınarak indisin sırasıyla sola, ortaya ve sağa yaslanması gösterilebilir:

```
$$\bigotimes_{\begin{subarray}{c}
      n\in\Gamma \\ m\not=n \\ 1<m<10
      \end{subarray}} (n+m) \qquad
  \sum_{\begin{subarray}{l}
        1<i<n \\ j\in\Lambda
      \end{subarray}} P_{i,j}$$
```

$$\bigotimes_{\substack{n\in\Gamma\\ m\neq n\\ 1<m<10}} (n+m) \qquad \sum_{\substack{1<i<n\\ j\in\Lambda}} P_{i,j}$$

amsmath paketinin `\sideset{ }{ }` komutu bir limitli operatörün dört köşesine bir indis yazdırır. Birinci küme parantezi içinde operatörün sol üst ve sol alt indisleri, ikinci küme parantezi içinde ise operatörün sağ üst ve sağ alt indisleri gösterilmektedir (küme parantezin içi boş da olabilir):

```
\[
\sideset{_1^2}{_3^4}\prod _n \qquad
\sideset{_{\alpha}^{\beta}}{_a^b}\sum _{k=1}^n
\]
```

$$\sideset{_1^2}{_3^4}\prod_n \qquad \sideset{^\beta_\alpha}{^b_a}\sum_{k=1}^n$$

Ayırtılan formülde limitli bir operatörün limiti operatörün üst ve/veya altına yazdırılır: $\sum\limits_{k=1}^{n}$, $\prod\limits_{k=1}^{n}$, $\bigotimes\limits^{\infty}$ .... Eğer bu blokta `\nolimits` komutu kullanılsa, operatörün tüm limiti operatöre indis olarak yazdırılır: $\sum_{k=1}^{n}$, $\prod_{k=1}^{n}$, ...

Fakat limitin bir kısmı indis olarak, diğer kısmı ise operatörün üst ve/veya altına $\sum\limits_{x\in X}^{n}$ gibi yazdırılması için `\limits` ve `\nolimits` komutları kullanılamaz. Böyle durumda, LaTeX 'in `\mathop` komutu aşağıdaki gibi kullanılabilir:

```
$$
\mathop{{\sum}^n}_{x\in X} f(x)\qquad
\mathop{{\sum}_{x\in X}}^n f(x)$
$$
```

$$\mathop{{\sum}^n}_{x\in X} f(x) \qquad \mathop{{\sum}_{x\in X}}^n f(x)$$

3.5.11 Parantez ve başka ayraçlar

Metinde `(`, `[`, `)` ve `]` parantezleri doğrudan klavye tuşlarıyla; küme parantezi `{ }` sırasıyla `\{` ve `\}` komutlarıyla ve köşe parantezi $\langle\ \rangle$ ise sırasıyla `\langle` ve `\rangle` komutlarıyla yazdırılır. Küme ve köşe parantezleri sadece bir matematik blokta geçerlidir.

Değişen boyutlu parantezler

Bir parantez içine alınan formülün bir kısmı büyük yüksekliyse (örneğin, kesir veya üstel fonksiyonu), bu parantez normale göre bir az büyük boyutlu olmalıdır. Böyle bir parantezin sol ve sağı TeX 'in sırasıyla `\left` ve `\right` komutlarıyla şöyle yazılmalıdır:

```
$$
\lim_{n\to\infty}(1-\frac{1}{n})^{
-n}=\lim_{n\to\infty} \left(1 +
       \frac{1}{n}\right)^n = e
$$
```

$$\lim_{n\to\infty}(1-\frac{1}{n})^{-n}=\lim_{n\to\infty}\left(1+\frac{1}{n}\right)^{n}=e$$

Görüldüğü gibi, ilk formül parantezi iyi gözükmüyor, sağdaki formülün parantezi ise `\left` ve `\right` komutlarıyla ayarlanmaktadır. Aslında bu komutlar herhangi tipli parantezlere uygulanabilir:

```
$$
\left|\left[\frac{\cos 1}{\sin 1} -
\frac{\sin 1}{\cos 1}\right]\right|=-2cot2
$$
```

$$\left|\left[\frac{\cos 1}{\sin 1}-\frac{\sin 1}{\cos 1}\right]\right|=-2cot2$$

NOT: Bir formülde `\left` ve `\right` komutları çift olarak yazılması lazım, yani bu komutların biri varsa, diğeri de varolmalıdır. Bu komutlara eşlik eden ayraçlar keyfi olarak alınabilir, mesele sol parantez `\left(` olarak verilip, sağ parantez ise `\right]` veya `\right\}` olarak verilebilir; hatta bunlar `\left)` ... `\right(` veya `\left]` ... `\right)` olarak da tanımlanabilir.

Ayraçlar

Yukarıda `\left` ve `\right` komutlarının `( )`, `[ ]` ve `| |` parantezlerine bazı uygulamaları ele alınmıştır. Tablo 3.13 'de `\left` ve `\right` komutları yardımıyla gerekli boyuta değişebilen *ayraçlar* verilmektedir.

Tablo 3.13: Ayraçlar

`(`	(	`)`	)	`\uparrow`	$\uparrow$	`\Uparrow`	$\Uparrow$
`[`, `\lbrack`	[	`]`, `\rbrack`	]	`\downarrow`	$\downarrow$	`\Downarrow`	$\Downarrow$
`\{`, `\lbrace`	{	`\}`, `\rbrace`	}	`\updownarrow`	$\updownarrow$	`\Updownarrow`	$\Updownarrow$
`\lfloor`	$\lfloor$	`\rfloor`	$\rfloor$	`\lceil`	$\lceil$	`\rceil`	$\rceil$
`\langle`	$\langle$	`\rangle`	$\rangle$	`/`	/	`\backslash`	\
`|`, `\vert`	|	`||`, `\Vert`	$\|$				

$|x|$ ve $\|A\|$ tipli bir formül `|` ve `\|` komutlarıyla değil, `amsmath` paketin `\lvert`, `\rvert`, `\lVert` ve `\rVert` komutlarıyla yazılırsa daha iyi olur:

```
$$
|x|,\|A\|\qquad \lvert x\rvert,\lVert A\rVert
$$
```

$$|x|, \|A\| \qquad |x|, \|A\|$$

Tablo 3.14 'de LaTeX 'in büyük ayraçları verilmektedir. Bu ayraçlara `\left` ve `\right` veya `\Big...`, `\bigg...` ve `\Bigg` komutların eşlik etmesi gerekir. Bu ayraçlar 148.sayfada da ele alınmaktadır.

Tablo 3.14: Büyük ayraçlar

`\lgroup`	$\lgroup$	`\rgroup`	$\rgroup$	`\lmoustache`	$\lmoustache$
`\arrowvert`	$\arrowvert$	`\Arrowvert`	$\Arrowvert$	`\rmoustache`	$\rmoustache$
`\bracevert`	$\bracevert$				

Yukarıda söylendiği gibi, bir formülde `\left` ve `\right` komutları çift

olarak yazılmalıdır. Diğer taraftan, sadece sol veya sağ ayraçlı formüller de vardır. Bu durumda, ayraç yerine "." (nokta) yazılmalıdır. Örneğin, ayraç simgesi [ ] olmak üzere, sadece sol ayraçlı bir formül \left[komutuyla başlamalı, \right. komutuyla bitmelidir. LaTeX nokta yerinde hiçbir şey yazdırmayacaktır:

```
$
f(x) = \left\{ \begin{aligned}
1 & x \not\in \mathbb{Q} \\
0 & x \in \mathbb{Q}
\end{aligned} \right.
$
```

$$f(x) = \begin{cases} 1 & x \notin \mathbb{Q} \\ 0 & x \in \mathbb{Q} \end{cases}$$

```
$$
\int\limits_a^b\frac{F(x)'}{F(x)^2}dx =
\left. \frac{-1}{F(x)} \right|_a^b
$$
```

$$\int\limits_a^b \frac{F(x)'}{F(x)^2}dx = \left. \frac{-1}{F(x)} \right|_a^b$$

Tablo 3.15 'de amssymb paketin özel ayraçlar verilmektedir.

Tablo 3.15: $\mathcal{AMS}$'ın ayraçları (amssymb paketi)

\ulcorner	⌜	\urcorner	⌝
\llcorner	⌞	\lrcorner	⌟

Bu ayraçlara \left ve \right komutlarının eşlik etmesi gerekmez. \left ve \right komutlarının uygulamasına benzer örnekler 3.10.1. paragrafında da ele alınmaktadır (*bkz.* 160.sayfa).

Ayraç boyutunu tanımlayan komutlar

\left ve \right komutlarının ürettiği ayraçlar bazen gereğinden aşırı büyük olmaktadır. Sebebi, TeX bir ayraçı, üst ve alt indis dahil olmak üzere, tüm formülün yüksekliğine göre üretir:

$$\left| a + b - \frac{a_1^2}{b_1^2} + c - d \right| \cdot |a + b| = \dots .$$

Böyle durumda, ayraç boyutu `\left` ve `\right` komutlarıyla değil, tablo 3.16 'de tanımlanan TEX 'in özel ayraçlarıyla verilmelidir.

Tablo 3.16: Özel ayraçlar

	Özel ayraçlar				
sol ayraç	`\bigl`	`\Bigl`	`\biggl`	ve	`\Biggl`
sağ ayraç	`\bigr`	`\Bigr`	`\biggr`	ve	`\Biggr`
orta ayraç	`\bigm`	`\Bigm`	`\biggm`	ve	`\Biggm`
keyfi ayraç	`\big`	`\Big`	`\bigg`	ve	`\Bigg`

Orta ayraç komutu kullanıldığında ayraçın soluna ve sağına küçük bir ek boşluk koyulmaktadır; *keyfi ayraç* komutu kullanıldığında ise ayraçın sadece yüksekliği değişmektedir:

```
$\bigl(a\in A\bigm| b\in B\bigr)$\\
$\bigl(a\in A\big| b\in B\bigr)$
```

$$a \in A \mid b \in B$$
$$a \in A \, b \in B$$

Özel ayaraç boyutlarını bir örnekle gösterelim:

```
$$
\Biggl(\biggl(\Bigl(\bigl((a)\bigr)\Bigr)\biggr)\Biggr)
$$
```

$$\Biggl(\biggl(\Bigl(\bigl((a)\bigr)\Bigr)\biggr)\Biggr)$$

Aşağıdaki örnekte `\left` ve `\right` komutlarıyla oluşturulan parantez boyutunun özel ayraçlar yardımıyla oluşturulan parantez boyutundan farkı daha iyi görülmektedir.

```
$$
\left( \sum_{k=1}^n x^k \right)^2 \qquad
\Bigl( \sum_{k=1}^n x^k \Bigr)^2
$$
```

$$\left(\sum_{k=1}^{n} x^k \right)^2 \qquad \Bigl(\sum_{k=1}^{n} x^k \Bigr)^2$$

Tablo 3.16 'deki bir özel ayraç tek olarak da kullanılabilir:

```
$\frac{1}{\sin x}\bigr|_{x=\pi /2} = 1$
```

$\frac{1}{\sin x}\bigr|_{x=\pi /2} = 1$

Yüksekliği büyük olan bir formülde ayraçlar `\left` ve `\right` komutlarıyla oluşturulsa, geçerli satırda satırlar arası düşey aralık bir az büyür. Düşey aralığın büyümemesi için ayraçlar elle özel yraçlarla oluşturulmalıdır:

```
metin ... $\left|\frac{a'}{b'}\right|$
formülünde \left ve \right komutları
yerine $\bigl|\frac{a'}{b'}\bigr|$ olarak
\bigl ve \bigr komutları yazılmalıdır.
```

metin ... $\left|\frac{a'}{b'}\right|$ formülünde `\left` ve `\right` komutları yerine $\bigl|\frac{a'}{b'}\bigr|$ olarak `\bigl` ve `\bigr` komutları yazılmalıdır.

3.5.12 Oklar

Tablo 3.13 'de değişen ayraç olarak bazı düşey okları verilmiştir. Tablo 3.17 'de standart okları tanımlayan komutlar, tablo 3.18 'de amssymb paketin özel okları ve tablo 3.19 'de ise bazı okların $\mathcal{AMS}$'deki karşıtı verilmektedir.

Tablo 3.17: Oklar

`\leftarrow, \gets`	$\leftarrow$	`\longleftarrow`	$\longleftarrow$	`\uparrow`	$\uparrow$
`\rightarrow, \to`	$\rightarrow$	`\longrightarrow`	$\longrightarrow$	`\downarrow`	$\downarrow$
`\leftrightarrow`	$\leftrightarrow$	`\longleftrightarrow`	$\longleftrightarrow$	`\updownarrow`	$\updownarrow$
`\Leftarrow`	$\Leftarrow$	`\Longleftarrow`	$\Longleftarrow$	`\Uparrow`	$\Uparrow$
`\Rightarrow`	$\Rightarrow$	`\Longrightarrow`	$\Longrightarrow$	`\Downarrow`	$\Downarrow$
`\Leftrightarrow`	$\Leftrightarrow$	`\Longleftrightarrow`	$\Longleftrightarrow$	`\Updownarrow`	$\Updownarrow$
`\mapsto`	$\mapsto$	`\longmapsto`	$\longmapsto$	`\nearrow`	$\nearrow$
`\hookleftarrow`	$\hookleftarrow$	`\hookrightarrow`	$\hookrightarrow$	`\searrow`	$\searrow$
`\leftharpoonup`	$\leftharpoonup$	`\rightharpoonup`	$\rightharpoonup$	`\swarrow`	$\swarrow$
`\leftharpoondown`	$\leftharpoondown$	`\rightharpoondown`	$\rightharpoondown$	`\nwarrow`	$\nwarrow$
`\rightleftharpoons`	$\rightleftharpoons$	`\iff`	$\iff$	`\leadsto`	$\leadsto$

`\leadsto` komutu latexsym paketiyle desteklenmelidir.

Tablo 3.18: $\mathcal{AMS}$'ın okları (amssymb paketi)

\dashleftarrow	⇠	\dashrightarrow	⇢	\multimap	⊸
\leftleftarrows	⇇	\rightrightarrows	⇉	\upuparrows	⇈
\leftrightarrows	⇆	\rightleftarrows	⇄	\downdownarrows	⇊
\Lleftarrow	⇚	\Rrightarrow	⇛	\upharpoonleft	↿
\twoheadleftarrow	↞	\twoheadrightarrow	↠	\upharpoonright	↾
\leftarrowtail	↢	\rightarrowtail	↣	\downharpoonleft	⇂
\leftrightharpoons	⇋	\rightleftharpoons	⇌	\downharpoonright	⇃
\Lsh	↰	\Rsh	↱	\rightsquigarrow	⇝
\looparrowleft	↫	\looparrowright	↬	\leftrightsquigarrow	↭
\curvearrowleft	↶	\curvearrowright	↷		
\circlearrowleft	↺	\circlearrowright	↻		

Tablo 3.19: $\mathcal{AMS}$'ın karşıt okları (amssymb paketi)

\nleftarrow	↚	\nrightarrow	↛	\nleftrightarrow	↮
\nLeftarrow	⇍	\nRightarrow	⇏	\nLeftrightarrow	⇎

3.5.13 Karışık simgeler

Bu paragrafta ele alınacak simgeler önceki ele alınan simge tiplerine benzer olması rağmen onların hiç birine ait değildir.

Tablo 3.20 'de verilen simgeler gerek matematik blokta gerekse normal metinde kullanılabilir.

Tablo 3.20: Metinde ve matematik blokta kullanılan simgeler

\copyright	©	\dag	†	\ddag	‡	\P	¶	\S	§
\pounds	£	\checkmark [a]	✓	\circledR [a]	®	\maltese [a]	✠		

[a] amsfonts veya amssymb paketi yüklenmelidir.

Tablo 3.21 'de LaTeX 'in sınıflandırılmayan simgeleri, tablo 3.22 'de ise $\mathcal{AMS}$-LaTeX 'in sınıflandırılmayan simgeleri verilmektedir.

Tablo 3.21: Ek simgeler

\aleph	$\aleph$	\emptyset	$\emptyset$	\infty	∞	\Re	$\Re$
\angle	$\angle$	\exists	$\exists$	\jmath	$\jmath$	\sharp	$\sharp$
\bot	$\bot$	\flat	$\flat$	\mho [a]	$\mho$	\spadesuit	$\spadesuit$
\Box [a]	$\Box$	\forall	$\forall$	\nabla	∇	\surd	$\surd$
\clubsuit	$\clubsuit$	\hbar	$\hbar$	\natural	$\natural$	\top	$\top$
\Diamond [a]	$\Diamond$	\heartsuit	$\heartsuit$	\neg, \lnot	$\neg$	\triangle	$\triangle$
\diamondsuit	$\diamondsuit$	\Im	$\Im$	\partial	∂	\wp	$\wp$
\ell	ℓ	\imath	$\imath$	\prime	$\prime$		

[a] latexsym paketi yüklenmelidir.

Tablo 3.22: $\mathcal{AMS}$'ın ek simgeleri (amssymb paketi)

\angle	$\angle$	\complement	$\complement$	\measuredangle	$\measuredangle$
\backprime	$\backprime$	\diagdown	$\diagdown$	\mho	$\mho$
\Bbbk	$\Bbbk$	\diagup	$\diagup$	\nexists	$\nexists$
\bigstar	$\bigstar$	\eth	$\eth$	\sphericalangle	$\sphericalangle$
\blacklozenge	$\blacklozenge$	\Finv	$\Finv$	\square	$\square$
\blacksquare	$\blacksquare$	\Game	$\Game$	\triangledown	$\triangledown$
\blacktriangle	$\blacktriangle$	\hbar	$\hbar$	\varnothing	$\varnothing$
\blacktriangledown	$\blacktriangledown$	\hslash	$\hslash$	\vartriangle	$\vartriangle$
\circledS	$\circledS$	\lozenge	$\lozenge$	\beth	$\beth$
\daleth	$\daleth$	\gimel	$\gimel$		

3.6 Harf üstü ve harf altı simgeleri

3.6.1 Şapka ve dalgalar

Tablo 3.11 'de şapka ve dalgaları tanımlayan sırasıyla \hat ve \tilde komutları ele alınmıştır. Fakat bu komutlar sadece tek bir harfe (veya simgeye) şapka veya dalga yazdırır. \widetilde ve \widehat komutları üç harf veya simgeye kadar örtülebilir sırasıyla dalga ve şapkayı yazdırır:

```
$\widetilde{XYZ} \ \widetilde{abc}\quad
 \widehat{f*g}=\hat f\cdot\hat g$
```

$$\widetilde{XYZ} \quad \widetilde{abc} \qquad \widehat{f*g} = \hat{f} \cdot \hat{g}$$

3.6.2 Üst ve alt çizgiler

Aşağıdaki komutlar kendi argümanına sırasıyla üst ve alt yatay çizgi yazdırır

```
\overline{ }          \underline{ }
```

Bir formülde birkaç `\overline` ve/veya `\underline` komutu kullanılmışsa, bu komutların argümanı farklı yükseklikte olduğundan üst ve/veya alt çizgiler farklı düzeylerde olmaktadır: $\overline{x}+\overline{t}$, $\underline{x}+\underline{f}$. Bu çizgilerin aynı düzeyde olması için her komutun argümanında ek olarak `\strut` komutu yazılmalıdır. `\strut` komutu eni sıfır, yüksekliği ise: $\underline{\overline{\updownarrow}}$ olan bir simgedir. Dolayısıyla, görüntüde bu simgenin kendisi gözükmeyecektir, o argümanda sadece bir standart yükseklik oluşturmaktadır:

```
$\overline{\strut x}+\overline{\strut t} \
\underline{\strut x}-\underline{\strut g}
\qquad \underline{x-g}$
```

$$\overline{x}+\overline{t} \qquad \underline{x}-\underline{g} \qquad \underline{x-g}$$

3.6.3 Küme parantezleri

Satır üstü ve satır altı yatay küme parantezleri sırasıyla

```
\overbrace{ }^{ }      ve      \underbrace{ }_{ }
```

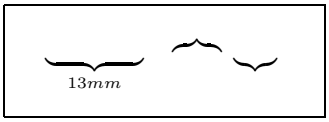
komutlarıyla yazdırılır. Komutların ikinci argümanı zorunlu değildir. Bu argüman `\overbrace` komutunda küme parantezin üstüne, `\underbrace` komutunda ise küme parantezin altına bir indis olarak yazdırılmaktadır:

```
$\underbrace{\overbrace{a+b+\cdots +
z}^{26} + 1 + \cdots + 10}_{36}$
```

$$\underbrace{\overbrace{a+b+\cdots+z}^{26}+1+\cdots+10}_{36}$$

`\overbrace` ve `\underbrace` komutların birinci argümanı zorunludur. Bu argümanda metin (veya formül) dışında `\hspace`, `\hspace*`, `\quad`, ... gibi komutlar da kullanılabilir veya bu argüman boş da olabilir

```
$\underbrace{\hspace*{1.3cm}}_{13mm} \quad
 \overbrace{\qquad} \ \underbrace{}$
```



3.6.4 Oklar

Bir formüle bir üst ok `\overleftarrow` veya `\overrightarrow` komutuyla yazdırılır. `\overleftarrow` komutu ok'a sol yön, `\overrightarrow` komutu ise sağ yön yazdırır:

```
$\overrightarrow{abc} +
 \overleftarrow{abc} = \vec{0}$,
$\overleftarrow{abc+\overrightarrow{abc}}$
```

$\overrightarrow{abc} + \overleftarrow{abc} = \vec{0}, \quad \overleftarrow{abc + \overrightarrow{abc}}$

Bir formüle bir alt ok amsmath paketin `\underleftarrow` veya `\underrightarrow` komutuyla yazdırılır:

```
$\underrightarrow{\psi_{\gamma}A_tg},
\ \underleftarrow{\psi_{\gamma}A_tg},
\ \underleftarrow{\psi_{\gamma}A_tg +
\underrightarrow{\psi_{\gamma}A_tg}}$
```

$\underrightarrow{\psi_\gamma A_t g}, \quad \underleftarrow{\psi_\gamma A_t g}, \quad \underleftarrow{\psi_\gamma A_t g + \underrightarrow{\psi_\gamma A_t g}}$

Her iki tarafı yönlü olan üst ve alt okları amsmath paketin sırasıyla

`\overleftrightarrow` ve `\underleftrightarrow`

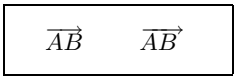
komutlarıyla yazdırılır:

```
$\overleftrightarrow{\psi _{\gamma}A_tg},$
$\underleftrightarrow{\psi _{\gamma}A_tg}$
```

$\overleftrightarrow{\psi_\gamma A_t g}, \quad \underleftrightarrow{\psi_\gamma A_t g}$

Bir vektörü yazmak için, genelde, `\overrightarrow` komutu kullanılır. Vektörün daha iyi gözükmesi için bu komutun argümanında `\mkern` komutu (*bkz.* 3.12.1, 174.sayfa) aşağıdaki gibi kullanılabilir:

```
$\overrightarrow{AB} \qquad
 \overrightarrow{\mkern -1mu AB\mkern 5mu}$
```



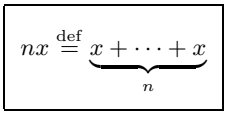
3.6.5 Keyfi simgeler

Bir satırda bir harf veya simgenin üstüne diğer bir simge

```
\stackrel{ }{ }
```

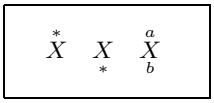
komutuyla yazdırılır. Komutun ikinci argümanı satırda kalacak simgedir, birinci argümanı ise bu simgenin üstüne yazdırılır:

```
$nx\stackrel{\mathrm{def}}{=}\underbrace{x +
\cdots + x}_{n}$
```



amsmath paketin iki argümanlı `\overset` ve `\underset` komutları `\stackrel` komutun bir genişletmesidir. Bu komutların birinci argümanı ikinci argümanın sırasıyla üst ve altına yazdırılır:

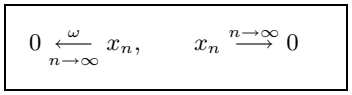
```
$\overset{*}{X} \quad \underset{*}{X}
\quad \overset{a}{\underset{b}{X}}$
```



3.7 İndisli oklar

İndisli oklar 3.6.5. paragraftaki komutlarla şöyle yapılabilir:

```
$0\overset{\omega}{\underset{n\to\infty}{
\longleftarrow}} x_n , \qquad
x_n \stackrel{n\to\infty}{\longrightarrow}0$
```



Sadece indisli okları yazdıran özel komutlar da vardır. amsmath paketinin iki argümanlı

```
\xleftarrow[ ]{ }          \xrightarrow[ ]{ }
```

komutları sırasıyla sol ve sağ yönlü indisli okları yazdırmaktadır. Komutların zorunlu olmayan birinci argümanı okun altına, zorunlu ikinci argümanı ise okun üstüne yazdırılmaktadır:

```
$$
1\xleftarrow[\infty\leftarrow n]{n\in N}
\Bigl(\frac{n-m}{n}\Bigr)\xrightarrow[
m\in Z_+]{m\to\infty}-\infty
$$
```

$$1 \xleftarrow[\infty\leftarrow n]{n\in\mathbb{N}} \frac{n-m}{n} \xrightarrow[m\in\mathbb{Z}_+]{m\to\infty} -\infty$$

3.8 $\mathcal{AMS}$ 'ın Binom katsayıları

Bir Binom ifadesi, genelde, LaTeX 'in `array` bloğunda yapılabilmektedir (*bkz.* 3.10.1.paragraf, 160.sayfa). Bu paragrafta, önceki paragraflarda olduğu gibi, $\mathcal{AMS}$-LaTeX 'in `amsmath` paketi ele alınmaktadır. Bu paketin iki argümanlı

```
\binom{ }{ }     \dbinom{ }{ }     \tbinom{ }{ }
```

komutların her biri bir Binom ifadesini yazdırır. `\dbinom` ve `\tbinom` komutları her zaman sırasıyla `displaystyle` ve `textstyle` yazı boyutuyla yazdırmaktadır:

```
$$\binom{n}{m}\quad \tbinom{n}{m}$$
$ \binom{n}{m} \quad \dbinom{n}{m}$
```

$$\binom{n}{m} \quad \tbinom{n}{m}$$
$$\tbinom{n}{m} \quad \dbinom{n}{m}$$

Komutların her iki argümanı da zorunlu değildir:

```
$$
\binom{n}{}\quad \binom{}{n}\quad \binom{}{}
$$
```

$$\binom{n}{}\quad \binom{}{n}\quad \binom{}{}$$

Fakat, argümanların küme parantezi, içi boş olması rağmen, varolmalıdır.

3.9 Matematiksel fonksiyonlar

3.9.1 Logaritmik ve trigonometrik fonksiyonlar

Analizdeki tüm temel fonksiyonlar: sin, cos, log, exp, ... TEX 'de kendi adından oluşan komutlarla tanımlanmıştır. TEX bu fonksiyonları doğru yazı fontuyla yazdırıp, fonksiyon adı ve argümanı arasına bir boşluk da verir. Tablo 3.23 'de bu fonksiyonların tam listesi verilmektedir.

Tablo 3.23: Logaritmik ve trigonometrik fonksiyonlar

\sin	\arcsin	\sinh	\sec	\log	\ker
\cos	\arccos	\cosh	\csc	\lg	\hom
\tan	\arctan	\tanh	\deg	\ln	
\cot	\arg	\coth	\dim	\exp	

Bu fonksiyonların herhangi birine bir alt ve/veya üst indis yazdırılabilir:

```
$
\cos^2(\pi/4)=0.5,\quad\log_{0.5}8=-3
$
```

$$\cos^2(\pi/4)=0.5,\quad\log_{0.5}8=-3$$

3.9.2 Limitli fonksiyonlar

lim tipli bir fonksiyon doğru yazı fontuyla yazdırılıp, fonksiyon adı ve argümanı arasına bir boşluk da verilir. Böyle bir fonksiyonun tablo 3.23 'de verilen fonksiyonlardan farkı şu ki, limitli bir fonksiyonun indisi 3.5.10. paragrafında ele alınan limitli operatörlerin indis yazdırma kuralına göre yazdırılmaktadır (*bkz.* 141.sayfa):

```
$$
\min_{x\in[0,1]}|\log_2|f(x)||\le
\max_{x\in[0,1]}|\log_2|f(x)||
$$
```

$$\min_{x\in[0,1]}|\log_2|f(x)|| \le \max_{x\in[0,1]}|\log_2|f(x)||$$

Tablo 3.24 'de limitli fonksiyonları tanımlayan komutlar verilmektedir.

Tablo 3.24: Limitli fonksiyonlar

`\limsup`	$\limsup$	`\lim`	$\lim$	`\max`	$\max$	`\det`	$\det$	`\sup`	$\sup$
`\liminf`	$\liminf$	`\inf`	$\inf$	`\min`	$\min$	`\gcd`	$\gcd$	`\Pr`	$\Pr$

Bir formülde üst ve alt limitler `\overline`, `\underline` ve `\lim` komutlarıyla şöyle yazdırılabilir:

```
$\overline{\lim} \quad \underline{\lim}$
```

$\overline{\lim} \quad \underline{\lim}$

Aslında bu limitleri de özel olarak yazdıran *özgün* komutlar vardır. Bu komutlar $\mathcal{AMS}$-LaTeX 'in amsopn paketine aittir. Bu paket ise amsmath paketin içindedir. Dolayısıyla, belgenin başlık kısmında amsmath paketi tanıtılmışsa, amsopn paketi tanıtılmayabilir. Tablo 3.25 'de amsopn paketin limitli fonksiyonları tanımlayan özgün komutları verilmektedir.

Tablo 3.25: $\mathcal{AMS}$'ın limitli fonksiyonları (amsopn paketi)

`\varlimsup`	$\varlimsup$	`\varliminf`	$\varliminf$
`\varinjlim`	$\varinjlim$	`\varprojlim`	$\varprojlim$

3.9.3 Yeni işlev adını tanımlamak

Yukarıda görüldüğü gibi, sin, log, lim, ... tipli bir fonksiyon doğru yazı tipiyle yazdırılmaktadır. Standart olmayan bir fonksiyon da standart fonksiyonlar sınıfına eklenebilir. Dolayısıyla bu fonksiyon da doğru yazı

tipiyle yazdırılır. amsopn paketinin aşağıdaki tanıtımı böyle bir fonksiyonu tanımlamaktadır

`\DeclareMathOperator{` *name*`}{`*function*`}`

Tanıtımın her iki argümanı zorunludur; ikinci argüman tanımlanacak yeni fonksiyondur, birinci argüman ise onun metinde komut olarak kullanılacak adıdır ve o (yani *name*) `\` simgesiyle başlanmalıdır. Örneğin, tanımlanacak yeni fonksiyonun hem kendisi hem de adı "SIN" olsun. Şu halde, bu fonksiyon

```
\DeclareMathOperator{\SIN}{SIN}
```

tanıtımıyla tanımlanmalıdır. Aşağıda "SIN" fonksiyonun uygulamasına bir örnek verilmektedir

```
\begin{align}
 x\sim 0 \Rightarrow \SIN(x) \sim 0
\end{align}
```

$$x \sim 0 \;\Rightarrow\; \mathrm{SIN}(x) \sim 0 \qquad (3.1)$$

Yeni fonksiyonun bir indisi `\lim` fonksiyonun indisi gibi olması için yukarıdaki tanıtımın yıldızlı şekli kullanılır:

`\DeclareMathOperator*{` *name*`}{`*function*`}`

Bunu bir örnekle gösterelim:

```
\DeclareMathOperator*{\trsup}{Trace\,sup}
```

⋮

```
\begin{align}
 \|f\|_{tr}\stackrel{\mathrm{def}}{=}
 \trsup_{t\in(0,1)} |f(t)|
\end{align}
```

$$\|f\|_{tr} \stackrel{\mathrm{def}}{=} \operatorname*{Trace\,sup}_{t\in(0,1)} |f(t)| \qquad (3.2)$$

NOT: `\DeclareMathOperator` ve `\DeclareMathOperator*` tanıtımları belgenin başlık kısmına amsopn veya amsmath paketinden sonra yazılmalıdır.

3.9.4 Modül fonksiyonu

Bir modül fonksiyonu $a \bmod b$ veya $x \equiv a \pmod b$ gibi tanımlanır. Her iki fonksiyon sırasıyla `\bmod` ve `\pmod` komutlarıyla yazdırılır:

```
$a\bmod b ,\qquad x\equiv a\pmod{b}$
```

$a \bmod b, \qquad x \equiv a \pmod b$

amsopn paketinin `\mod` ve `\pod` komutları bu fonksiyonları bir az değişik şekilde yazdırır:

```
$$
a\equiv b\mod c,\quad x\equiv y\pod z
$$
```

$$a \equiv b \mod c, \quad x \equiv y \pod z$$

3.10 Matris tipli şablonların yapısı

3.10.1 Matrisler

LaTeX 'de bir matris `array` bloğuyla oluşturulur

```
\begin{array}[align]{keys}
  strings
\end{array}
```

Bir matris birkaç satır ve sütundan oluşur. Bir satır sonu `\\` komutuyla gösterilmektedir. Sütunların sayısı komutun zorunlu *keys* argümanında aşağıdaki *anahtar* denilen seçenekleriyle gösterilir

- `l` - sütundaki formül sola yaslanır;
- `r` - sütundaki formül sağa yaslanır;
- `c` - sütundaki formül ortaya yaslanır.

Bir anahtar bir sütun üretir, dolayısıyla, anahtarlar sayısı sütunların sayısını oluşturur. Bir sütunun eni sütundaki en geniş formüle göre ayarlanır. İki sütun bir-birinden `&` simgesiyle ayrılır.

`array` bloğunun zorunlu olmayan *align* argümanı matrisin geçerli sayfadaki düşey yerini ayarlamaktadır. Bu argümanın aşağıdaki seçenekleri vardır:

`t` - matrisin ilk satırının taban çizgisi, geçerli satırın taban çizgisi düzeyindedir;

`b` - matrisin son satırının taban çizgisi, geçerli satırın taban çizgisi düzeyindedir;

`c` - matris geçerli satıra ortalanır.

`array` bloğunun *align* argümanı gösterilmediği takdirde LaTeX bu argümanın `c` seçeneğini alır.

Üstelik `array` bloğu `tabular` bloğuna çok benzerdir. Çünkü bu bloklarda satır ve sütunlar aynı şekilde oluşturulmaktadır (*bkz.* 2.10.1. paragraf, 74. sayfa).

`array` bloğunda yapılan matrise bir örnek verelim.

```
$$
\begin{array}{cccc}
a_{11} & a_{12} & \ldots & a_{1n} \\
a_{21} & a_{22} & \ldots & a_{2n} \\
\vdots & \vdots & \vdots & \vdots \\
a_{m1} & a_{m2} & \ldots & a_{mn}
\end{array}
$$
```

$$
\begin{array}{cccc}
a_{11} & a_{12} & \ldots & a_{1n} \\
a_{21} & a_{22} & \ldots & a_{2n} \\
\vdots & \vdots & \vdots & \vdots \\
a_{m1} & a_{m2} & \ldots & a_{mn}
\end{array}
$$

Görüldüğü gibi, `array` bloğunun kendisi de bir matematik bloğu içine alınmalıdır.

Bir matris soldan ve sağdan `\left` ve `\right` komutlarıyla bir parantez veya ayraç içine alınabilir (*bkz.* 3.5.11 ve 3.5.11. paragraf, 146 ve 147. sayfa). Örneğin, bir matris sol ve sağdan standart `( ... )` paranteze alınması için `\begin{array}` 'den önce `\left(` komutu, `\end{array}` 'den hemen sonra `\right)` komutu yazılmalıdır:

```
\begin{eqnarray*}
A = \left(
\begin{array}{cccc}
a_{11} & a_{12} &\ldots & a_{1n}\\
a_{21} & a_{22} &\ldots & a_{2n}\\
\vdots & \vdots &\vdots & \vdots\\
a_{m1} & a_{m2} &\ldots & a_{mn}
\end{array}
\right)
\end{eqnarray*}
```

$$
A = \left(
\begin{array}{cccc}
a_{11} & a_{12} & \ldots & a_{1n} \\
a_{21} & a_{22} & \ldots & a_{2n} \\
\vdots & \vdots & \vdots & \vdots \\
a_{m1} & a_{m2} & \ldots & a_{mn}
\end{array}
\right)
$$

Örnekte `array` bloğu `eqnarray*` bloğunun içine alınmıştır.

`tools` sınıfının `delarray` paketi bu işi daha da kolaylaştırır. Bu paket tanıtılmışsa, bir matris ayraçı doğrudan `array` bloğunda şöyle yazılabilir:

```
$$ J =
\begin{array}\|{cc}\|
   \begin{array}({cc})
   \lambda & 0 \\
         0 & \lambda
   \end{array}
   & 0 \\
0  &
   \begin{array}({cc})
   \mu & 0
         0 & \mu
   \end{array}
\end{array} $$
```

$$J = \begin{array}{|cc|} \begin{array}{cc} \lambda & 0 \\ 0 & \lambda \end{array} & 0 \\ 0 & \begin{array}{cc} \mu & 0 \\ 0 & \mu \end{array} \end{array}$$

Görüldüğü gibi, bu durumda, matris ayraçı *keys* argümanın soluna ve sağına yazılarak gösterilmektedir.

Bir satırda birkaç `array` bloğu kullanılabilir:

```
\begin{eqnarray*} \left(
    \begin{array}{cc}
       a_{11} & a_{12}\\
       a_{21} & a_{22}
    \end{array} \right)
 \cdot \left[
    \begin{array}{c}
       x_1 \\
       x_2
    \end{array} \right] = \left[
    \begin{array}{c}
    b_1\\
    b_2
    \end{array} \right]
\end{eqnarray*}
```

$$\left(\begin{array}{cc} a_{11} & a_{12} \\ a_{21} & a_{22} \end{array}\right) \cdot \left[\begin{array}{c} x_1 \\ x_2 \end{array}\right] = \left[\begin{array}{c} b_1 \\ b_2 \end{array}\right]$$

`array` bloğunda sadece bir matris değil, birkaç satır ve sütundan oluşan, istenilen tipteki, bir tablo da oluşturulabilir:

```
$$ \begin{array}{ccccc}
         &       & {\cal O} &    &         \\
         &       &  \vline  &    &         \\
{\cal M} & \gets &   XY     &\to &{\cal N}\\
         &       &  \vline  &    &         \\
         &       &  \infty  &    &
\end{array} $$
```

$$\begin{array}{ccccc}
 & & \mathcal{O} & & \\
 & & \vline & & \\
\mathcal{M} & \gets & XY & \to & \mathcal{N} \\
 & & \vline & & \\
 & & \infty & &
\end{array}$$

Görüldüğü gibi, tablo tipindeki herhangi bir şekil `\\` ve `&` komutlarıyla birkaç satır ve sütuna bölünerek `array` bloğunda yazdırılabilir.

Yukarıda söylendiği gibi, `array` bloğunun zorunlu olmayan *align* argümanın `t`, `c` ve `b` seçenekleriyle matrisin geçerli sayfadaki düşey yeri değiştirilebilir. Bunu bir örnekle gösterelim:

```
$$ A =
-\begin{array}[t]{c} a\\b \end{array},
-\begin{array}{c} a\\b \end{array},
-\begin{array}[b]{c} a\\b \end{array}
$$
```

$$A = -\begin{array}[t]{c} a \\ b \end{array}, -\begin{array}{c} a \\ b \end{array}, -\begin{array}[b]{c} a \\ b \end{array}$$

Görüldüğü gibi, seçeneğin `t` değerinde geçerli satırın taban çizgisi (bu çizgi tire `"-"` ile işaretlenmiştir) matris ilk satırın taban çizgisi düzeyinde; seçeneğin `b` değerinde bu çizgi son satırın taban çizgisi düzeyinde olmaktadır ve seçenek gösterilmediği taktirde ise, matris geçerli satıra ortalanmaktadır.

`tabular` bloğunda olduğu gibi, `array` bloğunda da:

- *keys* argümanın `l`, `c` ve `r` anahtarı dışında, önemli olan, bir `|` anahtarı daha vardır. Bu anahtar iki sütun arasına bir düşey doğru yazdırır.

- `\hline`, `\cline` ve `\multicolumn` komutları kullanılabilir (*bkz.* 2.10.1. paragraf, 74.sayfa).

Bu komutların `array` bloğuna uygulamasını iki örnekle gösterelim

```
\[ M = \left(
\begin{array}{clc}
1 &       0                  & 0 \\
\cline{2-3}
0 & \multicolumn{1}{|r}{0}  & i \\
0 & \multicolumn{1}{|r}{-i} & 0
\end{array}
\right) \]
```

$$M = \left(\begin{array}{clc} 1 & 0 & 0 \\ \cline{2-3} 0 & \multicolumn{1}{|r}{0} & i \\ 0 & \multicolumn{1}{|r}{-i} & 0 \end{array} \right)$$

```
$$ N = \left(
\begin{array}{clc}
\hline
\multicolumn{3}{|c|}{A} \\ \cline{1-3}
\multicolumn{2}{l}{0}  & 1 \\ \cline{1-2}
\multicolumn{1}{|r}{0} &
\multicolumn{2}{|r}{-i}\\
\cline{1-2}
\end{array}
\right) $$
```

$$N = \left(\begin{array}{clc} \hline \multicolumn{3}{|c|}{A} \\ \cline{1-3} \multicolumn{2}{l}{0} & 1 \\ \cline{1-2} \multicolumn{1}{|r}{0} & -i \\ \cline{1-2} \end{array} \right)$$

`\arrayrulewidth` komutunun değeri değiştirilerek `\hline`, `\cline` ve `|` komutlarıyla yazdırılan çizgi kalınlığı değiştirilebilir.
Yukarıda delarray paketinin bazı özellikleri ele alınmıştır. Bu özelliklerden yararlanarak aşağıdaki özel bir matris yazdırılabilir.

```
$$ \begin{array}{rc}
\hspace*{1.5cm} p & q \ { } \\
\multicolumn{2}{l}{A =
\begin{array}{c} p\\ q \end{array}
\begin{array}({cc})
I_p & O \\  O & J_q \end{array} }
\end{array} $$
```

$$A = \begin{array}{rc} & \begin{array}{cc} p & q \end{array} \\ \begin{array}{c} p \\ q \end{array} & \begin{array}({cc}) I_p & O \\ O & J_q \end{array} \end{array}$$

Aslında, böyle tipli bir matris TEX 'in

`\bordermatrix{`*matrix*`}`

komutuyla yazdırılabilir. Bu komutla $n \times n$ boyutlu bir matris $(1,1)$ öğesi olmayan $(n+1) \times (n+1)$ boyutlu bir matris olarak ayarlanmaktadır; matrisin her satırı `\cr` komutuyla bitmektedir.

Şimdi, yukarıdaki son örneği `\bordermatrix` komutuyla yazdıralım

```
$$ A = \bordermatrix{
  & p   & q   \cr
p & I_p & 0   \cr
q & 0   & J_q \cr } $$
```

$$A = \bordermatrix{ & p & q \cr p & I_p & O \cr q & O & J_q \cr }$$

`array` bloğunda bir matris öğeleri arasındaki uzaklık değiştirilebilir. `\arraycolsep` komutunun değeri iki komşu sütun arasındaki uzaklığın yarısına eşittir. Yeni bir değer verilmezse LaTeX bu uzaklığı `5pt` olarak alır. İki komşu satır arasındaki uzaklık ise metindeki normal bir "␣" boşluğunun `\arraystretch` komutunun değerine çarpımına eşittir. `\arraystretch` komutuna yeni bir değer verilmezse LaTeX bu değeri `1` olarak alır. Her iki komutun değeri de `\renewcommand` komutuyla değiştirilebilir.

3.10.2 $\mathcal{AMS}$ 'ın matrisleri

$\mathcal{AMS}$'ın bir matrisi `amsmath` paketinin

```
\begin{matrix}  ... \end{matrix}
\begin{pmatrix} ... \end{pmatrix}
\begin{bmatrix} ... \end{bmatrix}
\begin{vmatrix} ... \end{vmatrix}
\begin{Vmatrix} ... \end{Vmatrix}
```

bloklarının herhangi biriyle yazdırılabilir. Bu komutlu parantezlerin argümanı yoktur, dolayısıyla, matrisin sütunlarının sayısı da gösterilmemektedir. Matrisin her öğesi sütunda otomatik ortalanır. $\mathcal{AMS}$'in bir matrisi bu özellikleriyle `array` bloğun bir matrisinden farklılık göstermektedir.

Aşağıda `matrix`, `pmatrix`, `bmatrix`, `vmatrix` ve `Vmatrix` bloklarında oluşturulan matrislere birer tane örnek verilmektedir.

```
$$ A_1 = \begin{matrix}
            a & b \\
            c & d
         \end{matrix}  $$
$$ A_2 = \begin{pmatrix}
            a & -b \\
           -c & d
         \end{pmatrix} $$
$$ A_3 = \begin{bmatrix}
           -a & b \\
            c & -d
         \end{bmatrix} $$
$$ A_4 = \begin{vmatrix}
            1 & i \\
           -i & -1
         \end{vmatrix} $$
$$ A_5 = \begin{Vmatrix}
            1 & -i \\
           -i & 1
         \end{Vmatrix} $$
```

$$A_1 = \begin{matrix} a & b \\ c & d \end{matrix}$$

$$A_2 = \begin{pmatrix} a & -b \\ -c & d \end{pmatrix}$$

$$A_3 = \begin{bmatrix} -a & b \\ c & -d \end{bmatrix}$$

$$A_4 = \begin{vmatrix} 1 & i \\ -i & -1 \end{vmatrix}$$

$$A_5 = \begin{Vmatrix} 1 & -i \\ -i & 1 \end{Vmatrix}$$

$\mathcal{AMS}$-LaTeX bir matriste 10 sütuna kadar izin vermektedir. Eğer daha fazla sütun gerekse, `MaxMatrixCols` değişkenin değeri değiştirilmelidir:

```
\setcounter{MaxMatrixCols}{15}
```

Bu tanıtım matriste 15 sütuna kadar izin vermektedir.

Büyük matrislerde bazen bir veya birkaç öğe yerine *üç* veya *daha fazla nokta* yazdırılması gerekmektedir. Bu noktalar `\dots`, `\cdots`, `\ldots` komutlarıyla veya

```
\hdotsfor[n]{m}
```

özel komutuyla yazdırılabilir. Komutun zorunlu olmayan n argümanında standart birime göre noktalar arasındaki uzaklık ve zorunlu olan m argümanında ise noktalarla kapsayan sütunlar sayısı verilmektedir. Bunu bir örnekle gösterelim:

```
$$ \Delta :=
\begin{Vmatrix}
 1 &  0 & \hdotsfor{2} & 0 \\
-1 &  1 & 0 & \dots    & 0 \\
      \hdotsfor[2]{5}      \\
-1 & -1 & \dots & -1   & 1
\end{Vmatrix} $$
```

$$\Delta := \begin{Vmatrix} 1 & 0 & \cdots & 0 \\ -1 & 1 & 0 & \dots & 0 \\ & & \cdots & & \\ -1 & -1 & \dots & -1 & 1 \end{Vmatrix}$$

Yukarıda tanımlanan komutlarla bir metin içinde küçük bir matris yazdı-

rılamaz. Böyle bir matris amsmath paketinin komutlu

```
\begin{smallmatrix}  ... \end{smallmatrix}
```

paranteziyle yazdırılabilir. Bunu da bir örnekle gösterelim:

```
Metin içinde küçük bir matris yazdırılması
için {\tt smallmatrix} bloğu şöyle kulla-
nılmalıdır : $\left( \begin{smallmatrix}
a & b \\ c & d \end{smallmatrix} \right)$.
Burada metin yine devam etmektedir.
```

Metin içinde bir küçük matris yazdırılması için smallmatrix bloğu şöyle kullanılmalıdır: $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix} \right)$. Burada metin yine devam etmektedir.

3.10.3 Kesir tipli bir formül

Bir kesrin \frac komutuyla yazdırılması 3.5.9. paragrafında ele alınmıştır (*bkz.* 139. sayfa). Kesir çizgisi olmayan kesir tipli bir formül LaTeX 'de sadece array bloğuyla yazdırılabilir:

```
$
\begin{array}{c} x+y \\ xy \end{array}
\left\langle
\begin{array}{c}x-y \\ xy \end{array}
\right\rangle \quad \left(
\begin{array}{c} xy \\ x-1 \end{array}
\right)
$
```

$$\begin{array}{c} x+y \\ xy \end{array} \left\langle \begin{array}{c} x-y \\ xy \end{array} \right\rangle \quad \left(\begin{array}{c} xy \\ x-1 \end{array} \right)$$

$\mathcal{AMS}$-LaTeX 'in de bu tipli kesirleri yazdıran özel bir komutu vardır. Kesir tipli herhangi bir formül amsmath paketinin

```
\genfrac{l}{r}{widthline}{n}{numerator}{denominator}
```

komutuyla yazdırılabilir. Komutun tüm altı argümanı da zorunludur. *l* ve *r* argümanı "kesrin" sırasıyla sol ve sağ ayracı; *widthline* argümanı kesir çizgisin kalınlığı; *n* argümanında 0, 1, 2 ve 3 sayıların biriyle "kesir pay ve paydasının" yazı tipi gösterilir (displaystyle, ...) ve sonuç olarak , *numerator* ve *denominator* argümanı kesrin sırasıyla pay ve pay-

dasıdır. Komutun istenilen bir argümanı gösterilmeden küme parantez içi boş bırakılabilir.

`\genfrac` komutuna bir örnek verelim

```
$$
\genfrac{}{}{}{}{n+m}{n}
\genfrac{}{}{0pt}{}{n+m}{n}
\genfrac{\langle}{\rangle}{0pt}{}{n}{m}
\genfrac{\langle}{\rangle}{0pt}{2}{n}{m}
\genfrac{[}{]}{}{1}{n-m}{m}
$$
```

$$\genfrac{}{}{}{}{n+m}{n} \quad \genfrac{}{}{0pt}{}{n+m}{n} \quad \genfrac{\langle}{\rangle}{0pt}{}{n}{m} \quad \genfrac{\langle}{\rangle}{0pt}{2}{n}{m} \quad \genfrac{[}{]}{}{1}{n-m}{m}$$

3.10.4 Parçalı fonksiyonlar

Parantezli koşullar sistemi `array` bloğunda `\left`, `\left.`, `\right` ve `\right.` komutlarıyla şöyle yazdırılabilir:

```
$$
f(x) = \left\{
\begin{array}{cl}
\sin x &x\in[-\pi,\pi]\textrm{ise,}\\
0      &\textrm{diğer durumlarda.
\end{array}
\right.
$$
```

$$f(x) = \left\{ \begin{array}{cl} \sin x & x\in[-\pi,\pi]\textrm{ise,}\\ 0 & \textrm{diğer durumlarda.} \end{array} \right.$$

$\mathcal{AMS}$-LaTeX 'de amsmath paketinin parantezli

```
\begin{cases}  ...  \end{cases}
```

komutu yukarıdaki sistemi daha basit olarak oluşturmaktadır:

```
$$
f(x) =
\begin{cases}
\sin x & x\in[-\pi,\pi]\text{ise,}\\
0      & \text{diğer durumlarda.}
\end{cases}
$$
```

$$f(x) = \begin{cases} \sin x & x\in[-\pi,\pi]\text{ise,}\\ 0 & \text{diğer durumlarda.} \end{cases}$$

Görüldüğü gibi, `cases` bloğunun argümanı yoktur ve her sütun otomatik sola yaslanmaktadır.

3.11 Yazı stilleri

3.11.1 Formül içinde metin

Bir formül içinde bir metin yazısı 11. paragrafında ele alınan `\textrm`, `\textsf`, `\texttt`, `\textmd`, `\textbf`, `\textit`, `\textsl`, `\textsc`, `\textup` ve `\textnormal` komutlarıyla ayarlanabilir (*bkz.* 92. sayfa). Bunu bir örnekle gösterelim:

```
$$
\textsf{her} x\in\mathbf{R} \textsf{için}
x^2 \geq 0 \textsf{'dir.}
$$
```

her $x \in \mathbf{R}$ için $x^2 \geq 0$ 'dir.

Bundan başka, `\mbox` komutu da kullanılabilir (*bkz.* 2.8.1. paragraf, 67. sayfa). Bu durumda, formül içindeki bir metin formülden önceki geçerli metin yazısıyla yazdırılır:

```
{\it                              ...
her $x\in\mathbf{R} \mbox{için}
 x^2 \geq 0 \ \mbox{'dir;}$ ... }
```

... her $x \in \mathbf{R}$ için $x^2 \geq 0$ 'dir; ...

Üstelik `\mbox` komutunun argümanı da bir formülü içerebilir:

```
{\sc ... Her $x\in\mathbf{R} \
\mbox{için $x^2\geq 0$ 'dir;}$ ... }
```

... Her $x \in \mathbf{R}$ için $x^2 \geq 0$ 'dir; ...

Aslında formül içindeki bir metnin normal metin yazısıyla yazdırılması için `\textrm` ve `\mbox` komutları yerine `\mathrm{ }` komutunun kullanılması daha iyidir. Bunu aşağıdaki bir örnekle gösterelim:

```
$$
2^{\mbox{nd}} \qquad  2^{\textrm{nd}}
\qquad  2^{\mathrm{nd}}
$$
```

2^{nd} 2^{nd} 2^{nd}

`\mathrm` komutu bir matematik blokta kullanılabilen tek komut değildir. Böyle komutların tam listesi tablo 3.26 'de verilmektedir (*bkz.* 3.11.2. paragraf, 171. sayfa).

Bir formül içinde büyük olmayan bir metin `amstext` paketinin `\text{ }`

komutuyla da yazdırılabilir. amstext paketi amsmath paketinin içindedir, dolayısıyla, belgenin başlık kısmında amsmath paketi tanıtılmışsa, amstext paketi tanıtılmayabilir. Ama, eğer amsmath paketi tanıtılmamışsa, amstext paketin sadece kendisi tanıtılabilir. Bir matematik blokta \text komutun kendisi metini gerekli metin yazısıyla yazdırır:

```
$$
A^{\text{üst indis}}_{\text{alt indis}}
$$
```

$$A^{\text{üst indis}}_{\text{alt indis}}$$

Denklemler arasına metin koyulması

Bir matematik blok kesilmeden bloğun bir veya iki satırına amsmath paketinin \intertext komutuyla normal bir metin yazdırılabilir. Bu komut sadece \\ veya * komutundan sonra yazılmalıdır. Bunu bir örnekle gösterelim:

```
\begin{align*}
a_1 &= 1 \\
a_2 &= 3 \\
a_3 &= 5 \\
\intertext{şu şekilde devam edi-
lirse, her $n\in\mathbf{N}$ için}
a_n &= 2n-1 \\
\intertext{olduğu elde edilir.}
\end{align*}
```

$$\begin{aligned} a_1 &= 1 \\ a_2 &= 3 \\ a_3 &= 5 \end{aligned}$$
şu şekilde devam edilirse, her $n \in \mathbf{N}$ için
$$a_n = 2n - 1$$
olduğu elde edilir.

3.11.2 Matematik alfa sayısalı

Bir matematik blokta her şey özel bir matematik yazısıyla yazdırılır. Bu yazı tipi *matematik alfa ayısal* adı verilen özel komutlarla başka bir matematik yazısına dönüştürülebilir. Fakat bu komutlar sadece harf, rakam, vurgu ve yunanca harflere etkilidir; matematiksel simgeler ($\sum$, $\int$, $\oplus$, ...) ise, değişmeyecektir. Tablo 3.26 'de bütün matematik alfa sayısalları örneklerle verilmektedir.

\mathtt alfa sayısalı \dot vurgusuna etkisizdir. \mathnormal alfa sayısalı ise amsmath paketi tanıtıldıktan sonra vurguları daha düzgün yazdırır;

Tablo 3.26: Matematik alfa sayısalı

`\mathcal` [a]	$\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$		
`\mathnormal`	$ABC, abc, 123, \hat{a}, \tilde{b}, \Psi\Omega$	`\mathrm`	$\mathrm{ABC, abc, 123, \hat{a}, \tilde{b}, \Psi\Omega}$
`\mathbf`	$\mathbf{ABC, abc, 123, \hat{a}, \tilde{b}, \Psi\Omega}$	`\mathsf`	$\mathsf{ABC, abc, 123, \hat{a}, \tilde{b}, \Psi\Omega}$
`\mathtt`	$\mathtt{ABC, abc, 123, \hat{a}, \tilde{b}, \Psi\Omega}$	`\mathit`	$\mathit{ABC, abc, 123, \hat{a}, \tilde{b}, \Psi\Omega}$

[a] Alfa sayısal sadece Latince baş harflere etkilidir .

fakat, amsmath paketi bazı alfa sayısalların çalışmasını etkiyebilir, örneğin, `$\mathbf{\vec a}$` ifadesi $\mathbf{\vec a}$ yerine $\mathbf{\tilde a}$ 'yı yazdırabilir.

$\mathcal{AMS}$-LaTeX 'in eucal, eufrak, ve amsfonts paketleri üç alfa sayısalı daha tanımlamaktadır. Bu durumda, eucal paketi `mathscr` seçeneğiyle tanıtılmalıdır. $\mathcal{AMS}$-LaTeX 'in bu alfa sayısalları tablo 3.27 'de verilmektedir.

Tablo 3.27: $\mathcal{AMS}$ 'ın matematik alfa sayısalı

`\mathscr` [a]	$\mathscr{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
`\mathfrak` [b,c]	$\mathfrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
	$\mathfrak{abcdefghijklmnopqrstuvwxyz}$
`\mathbb` [c]	$\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$

[a] eucal paketi `mathscr` seçeneğiyle tanıtılmalıdır;
[b] eufrak paketi tanıtılmalıdır;
[c] amsfonts veya amssymb paketi tanıtılmalıdır.

RSFS 'in mathrsfs paketi `\mathscr` komutuyla bir alfa sayısalı daha tanımlamaktadır. Bu alfa sayısal tablo 3.28 'de verilmektedir.

Tablo 3.28: RSFS 'in matematik alfa sayısalı (mathrsfs paketi)

`\mathscr`	$\mathscr{A\,B\,C\,D\,E\,F\,G\,H\,I\,J\,K\,L\,M\,N\,O\,P\,Q\,R\,S\,T\,U\,V\,W\,X\,Y\,Z}$

Görüldüğü gibi, mathrsfs ve eucal paketlerinin farklı alfa sayısalı aynı `\mathscr` adlıdır. Dolayısıyla, belgenin başlık kısmında bu paketlerin hangisi son olarak tanıtılmışsa, onun alfa sayısalı çalışır. Metinde her iki

alfa sayısalın çalışması için bu alfa sayısallardan birinin adının değiştirilmesi gerekmektedir. Örneğin, eucal paketinin bu alfa sayısalı değiştirilmeden, mathrsfs paketinin \mathscr adlı alfa sayısalının değiştirilmesi için belgenin başlık kısmında

```
\DeclareSymbolFont{rsfs}{U}{rsfs}{m}{n}
\DeclareSymbolFontAlphabet{\rsfsmathscr}{rsfs}
\usepackage[mathscr]{eucal}
```

tanıtımları yazılmalıdır. Burada \rsfsmathscr komutu \mathscr adlı alfa sayısalının yeni adıdır. Böylece mathrsfs paketinin \mathscr adlı alfa sayısalı metinde yeni \rsfsmathscr adıyla yazılmalıdır:

```
$\mathscr{ABCDEFGH}\dots\mathscr{Z}$
$\rsfsmathscr{ABCDEFGH}\dots\rsfsmathscr{Z}$
```

$\mathscr{ABCDEFGH}\dots\mathscr{Z}$
$\rsfsmathscr{ABCDEFGH}\dots\rsfsmathscr{Z}$

3.11.3 Simgelerin koyu doygunluğu

Matematik simgelerin doygunluğu LaTeX 'in

\mathversion{*version*}

komutuyla değiştirilebilir. Zorunlu *version* argümanın sadece iki: normal ve bold seçeneği vardır. Bu komut bir formülden önce bir tanıtım olarak kullanılmalıdır:

```
$M'_n \otimes \sqrt{\sinx-\lnx}$
{
\mathversion{bold} $M'_n\otimes\sqrt{\sinx-\lnx}$
}
```

$M'_n \otimes \sqrt{\sinx-\lnx}$
{
\mathversion{bold} $M'_n\otimes\sqrt{\sinx-\lnx}$
}

Örnekten anlaşılacağı gibi, \mathversion tanıtımının etki dairesindeki tüm harf ve simgelerin doygunluğu bold doygunluğuna (yani **koyu** doygunluğuna) dönüşmektedir. Üstelik bu tanıtım tüm formüle aittir. Dolayısıyla, bir formülün sadece bazı simgelerini **koyu** doygunluğuyla yazdırmak için bu formül birkaç defa kesilerek onun her parçası \mathversion{bold} tanıtımıyla başlatılmalıdır:

```
$M'_n \otimes \sqrt{\sinx-\lnx}$
{\mathversion{bold} $M'_n$} $\otimes$
{\mathversion{bold} $\sqrt{\sinx-\lnx}$}
```

$M'_n \otimes \sqrt{\sin x - \ln x}$
$\boldsymbol{M'_n \otimes \sqrt{\sin x \quad \ln x}}$

Fakat bir formül her yerinden kesilemez, örneğin, örnekteki formülün $\sqrt{\sin x - \ln x}$ parçası hiçbir yerinden kesilemez. Dolayısıyla, bu parça tümden ya normal doygunluğuyla ya da koyu doygunluğuyla yazdırılır.

Böyle durumda, her sefer olduğu gibi, bize yine $\mathcal{AMS}$-LaTeX yardımcı olabilir: amsbsy paketinin `\boldsymbol{ }` komutu argümanını koyu doygunluğuyla yazdırır. Bu komut bir matematik bloğunda kullanılabilir:

```
\(M'_{\boldsymbol{n}} \otimes \boldsymbol{\pm}
  \sqrt{\sinx - \boldsymbol{\lnx}}\)
```

$M'_{\boldsymbol{n}} \otimes \sqrt{\sin x - \boldsymbol{\ln x}}$

amsmath paketi amsbsy 'i içeriyor, dolayısıyla, belgenin başlık kısmında amsmath paketi tanıtılmışsa, amsbsy paketi tanıtılmayabilir.

Bazı matematiksel simgelerin koyu doygunluğu $\mathcal{AMS}$-LaTeX 'de tanımlanmamış olabilir. Bu halde, `\mathversion{bold}` ve `\boldsymbol` komutları o simgeyi normal doygunluğuyla yazdırır. Böyle durumda, amsbsy paketinin `\pmb` [3] komutu kullanılabilir. Bu komut kendi argümanını her zaman koyu doygunluğuyla yazdırır:

```
$$ \sum _{k=0}^{\infty} \xi _k ,$$
{  \mathversion{bold}
$$ \sum _{k=0}^\infty \xi_k ,
   \quad\mathop{\pmb{\sum}}_{k=0}^\infty \xi_k $$
}
```

$$\sum_{k=0}^{\infty} \xi_k,$$
$$\boldsymbol{\sum_{k=0}^{\infty} k, \quad \mathop{\pmb{\sum}}_{k=0}^{\infty} k}$$

3.12 Formüllerin ayarlanması

3.12.1 Matematik bloklarında boşluklar

Bazen bir formülde TeX 'in simgeler arasına koyduğu bir boşluk başarısız olabilir. Böyle durumda, simgeler arasına gerekli boşluklar tablo 3.29 'de verilen komutlar yardımıyla elle koyulmalıdır.

[3] İngilizce: "poor man's bold" kelimelerin baş harfleri alınmıştır

Tablo 3.29: Matematik bloklarında boşluklar

TEX	AMS-LATEX (amsmath paketi)	aralık	değeri em, mu	değeri mm
\qquad		⌋ ⌊	2em	≈7mm
\quad		⌋ ⌊	1em	≈3,5mm
\;	\thickspace	⌋⌊	5mu	≈1mm
\:	\medspace	⌋⌊	4mu	≈0,77mm
\,	\thinspace	⌋⌊	3mu	≈0,58mm
\!	\negthinspace	‖	-3mu	≈-0,58mm
	\negmedspace	‖	-4mu	≈-0,77mm
	\negthickspace	‖	-5mu	≈-1mm

Aşağıdaki örnekte TEX 'in koyduğu boşlukları tablo 3.29 'deki komutlarla elle koyulan boşluklarıyla kıyaslayalım

```
$\int\int f dxdy\quad\sqrt{3}x\quad n/\ln x$
$\int\!\!\int f\,dxdy \quad
 \sqrt{3}\,x \quad n/\!\ln x$
```

$$\int\int f dxdy \quad \sqrt{3}x \quad n/\ln x$$
$$\int\!\!\int f\,dxdy \quad \sqrt{3}\,x \quad n/\!\ln x$$

Görüldüğü gibi, tablo 3.29 'deki boşluklar, genelde, simgeler arasına küçük eksi veya artı boşluklar koyulması için kullanılmaktadır; büyük boşluklar ise `\hspace` veya `\hspace*` komutlarıyla koyulabilir (2.2.1.paragraf, 45. sayfa). Bir matematik blokta bu komutlar dışında `\mkern` komutu da kullanılabilir. Fakat bu komutun argümanında uzaklık birimi `mu` (`1mu`≈`0,2mm`) ile gösterilmelidir:

```
$a \mkern 30mu b \mkern 40mu c \mkern -20mu d$
```

$$a \mkern 30mu b \mkern 40mu c \mkern -20mu d$$

amsmath paketin `\mkern` komutuna benzer olan `\mspace{ }` komutu vardır. Bu komutun argümanında da uzaklık birimi `mu` ile gösterilmelidir:

```
$a \mkern 15mu b \mspace{30mu} c \mspace{-20mu} d$
```

$a \mkern 15mu b \mspace{30mu} c \mspace{-20mu} d$

3.12.2 Hecelemede işleçlerin çiftlenmesi

Bir formül hecelemesinde TEX formülü sadece bir ikili ilişki veya işlem yerinden kesişi 3.1.1.paragrafında ele alınmıştır (*bkz.* 111.sayfa). Bu durumda, TEX heceleme yerindeki ikili işlemin (veya ilişkinin) ikinci kopyasını yeni satıra yazdırmamaktadır:

```
Pisagor teoremi söylüyor ki:
\(a^2 + b^2 = c^2\).
```

Pisagor teoremi söylüyor ki: $a^2 + b^2 = c^2$.

Hecelemede işlem veya ilişkinin iki kopyasının oluşturulması için TEX 'in

`\discretionary{`*end line*`}{`*begin line*`}{`*text*`}`

komutu kullanılmalıdır. Komutun tüm üç argümanı da zorunludur. Satır içinde komutun sadece üçüncü argümanı, satır sonunda komutun birinci argümanı ve yeni satır başında ise komutun ikinci argümanı yazdırılır. Bir matematik blokta komutun üçüncü argümanı boş olmalıdır. Bunu bir örnekle gösterelim:

```
Pisagor teoremi söylüyor ki: \(a^2 +
\discretionary{}{+}{} b^2 = c^2\).
Formül satır içine düşse, bu komut
işlemin ikinci kopyasını yazdırma-
maktadır: \(a^2 +
\discretionary{}{+}{} b^2 = c^2\).
```

Pisagor teoremi söylüyor ki: $a^2 + + b^2 = c^2$.
Formül satır içine düşse, bu komut işlemin ikinci kopyasını yazdırmamaktadır: $a^2 + b^2 = c^2$.

Görüldüğü gibi, `\discretionary` komutu bir formülün satır sonuna düşüşü mümkün olan tüm ikili işlem veya ilişkilerine ek olarak yazılmalıdır. Bunu kolaylaştırmak için aşağıdaki bir tanıtımla bir yardımcı komut tanımlayalım:

```
\newcommand{\hm}[1]{#1\nobreak\discretionary{}{\hbox{\ensuremath{#1}}}{}}
```

Bu tanıtımda yukarıda kullanılan `\discretionary{}{+}{}` tipli tüm komutlar kısaca `\hm` (`\hm`*operation*) komutuyla yeniden adlandırılmaktadır. `\hm` komutuyla yukarıdaki örnek kısaca şöyle yazılabilir:

```
Pisagor teoremi söylüyor ki:
\(a^2 \hm+ b^2 \hm= c^2\).
```

Pisagor teoremi söylüyor ki: $a^2 + + b^2 = c^2$.

3.12.3 Bölünmez tire

Bir metinde *n-boyutlu*, *μ-ölçülü*, ... gibi ifadelerde heceleme tireye düşebilir:

```
$e_1,e_2,\dots , e_n$ vektörler sistemi $n$-boyutlu uzayın tabanıdır.
```

$e_1, e_2, \dots, e_n$ vektörler sistemi n-boyutlu uzayın tabanıdır.

Böyle ifadelerde hecelemenin tireye düşmesine izin verilmemesi için amsmath paketin `\nobreakdash` komutu kullanılmalıdır. Bu komut kendisinden sonradaki ilk kelimenin hecelemesine izin vermemektedir.

```
$e_1, e_2, \dots , e_n$ vektörler sistemi $n$\nobreakdash-boyutlu uzayın tabanıdır.
```

$e_1, e_2, \dots, e_n$ vektörler sistemi n-boyutlu uzayın tabanıdır.

`\nobreakdash` komutun kelimenin tümüne değil, sadece bir kısmına etkili olması için kelimenin herhangi bir yerine eni sıfır olan bir boşluk şöyle yazılmalıdır

```
$e_1, \dots , e_n$ vektörler sistemi $n$\nobreakdash-bo\hspace{0pt}yutlu uzayın tabanıdır.
```

$e_1, \dots, e_n$ vektörler sistemi n-boyutlu uzayın tabanıdır.

Aslında `\nobreakdash` komutu "15–32" tipli kısa ifadelerde sık-sık kullanılmaktadır. Kullanımı daha da kolaylaştırmak için bu komut

```
\newcommand{\ndash}{\nobreakdash--}
```

olarak yeniden tanıtılmalıdır. Şu halde, "15–32" ifadesi metinde kısaca `15\ndash 32` olarak yazılabilir.

3.12.4 Görünmez simgeler

Bazen bir formülde kendisi yazdırılmayan, fakat formülde yer tutan bir simge gerekmektedir. Böyle simgelere *görünmez simgeler* veya *pantom*lar (adı var kendi yok) denir. Görünmez simgeyi içeren formüller 3.5.1. paragrafta ele alınmıştır (*bkz.* 130.sayfa). Herhangi bir simge

`\phantom{`*formula*`}`

komutuyla bir görünmez simgeye dönüştürülebilir. Bunun için bu simge (veya formül) `\phantom` komutunun *formula* argümanına yazılmalıdır. Bunu bir örnekle gösterelim:

```
$\int$ simgesi aşağıdaki formülde \phantom
komutu altında görünmez simgeye dönüşe-
rek kayıp olmaktadır:
$\sqrt{\phantom{\int}a(x)dx}$
```

$\int$ simgesi aşağıdaki formülde \phantom komutu altında görünmez simgeye dönüşerek kayıp olmaktadır: $\sqrt{a(x)dx}$

`\phantom` 'ın *formula* argümanı zorunludur. Bu argüman gösterilmediği taktirde, TEX komuttan sonradaki ilk simgeyi görünmez simgeye dönüştürmektedir:

```
$\sum\limits _{k=1}^n \phantom A_k = 1$
```

$$\sum_{k=1}^{n} {}_k = 1$$

Örneklerden anlaşılacağı gibi, `\phantom` komutu *formula* argümanına yazılan bir simgeyi görünmez simgesine öyle dönüştürmektedir ki, sanki bu simge önce yazılıp, sonra silinmiş gibidir. Değişik bir ifade ile, `\phantom` komutu altındaki simge yazdırılmaması rağmen onun formüldeki yeri iptal edilmemektedir.

Bir *düşey pantom*, yani görünmez bir ifadesine dönüşen ve formüldeki yeri de iptal edilen bir simge, TEX 'in

`\vphantom{`*formula*`}`

komutuyla oluşturulur. Komutun özel `\vphantom{(}` hali `\mathstrut` komutuyla gösterilir. 3.5.1. paragrafta ele alınan (*bkz.* 130. sayfa) $Fe_2^{+2}Cr_2O_4$ formülün alt indisleri türlü düzeyde olduğundan bu formül iyi gözükmüyor. Bu problem `\vphantom` komutu yardımıyla şöyle çözülmektedir:

```
$
\mathrm{Fe_2^{+2}Cr_2^{\vphantom{+2}}O_4^{\vphantom{+2}}}
$
```

$\mathrm{Fe_2^{+2}Cr_2^{\vphantom{+2}}O_4^{\vphantom{+2}}}$

Bir *yatay pantom* TEX 'in `\hphantom{`*formula*`}` komutuyla oluşturulur:

```
Bir formül şu $\hphantom{\cos\alpha}$
boş yere elle yazılabilir.
```

Bir formül şu boş yere elle yazılabilir.

3.12.5 Yer tutmayan görünen simgeler

TEX bir simge veya harfe onun boyutuna göre yer ayırır. Bu boyutun bir parametresi, gerekirse, iptal edilebilir. TEX 'in

`\lefteqn{`*formula*`}`

komutu *formula* argümanın yatay boyutunu iptal etmektedir. Değişik bir ifade ile, komutun *formula* argümanı kendisi için bir yatay aralığı ayırtılmadan yazdırılır:

```
$\lefteqn{=}\backslash \quad \lefteqn{=}/
 \quad a\lefteqn{=}| \quad a\lefteqn{=}||||
 \quad a\lefteqn{\in}/ $
```

$\lefteqn{=}\backslash \quad \lefteqn{=}/ \quad a\lefteqn{=}| \quad a\lefteqn{=}|||| \quad a\lefteqn{\in}/$

TEX 'in `\smash` komutu `\lefteqn` komutu gibi olup, bu komut *formula* argümanın düşey boyutunu iptal etmektedir. Bu durumda, komutun argü-

manı kendisi için bir düşey aralığı ayırtılmadan yazdırılır:

```
$$\sqrt{\smash{\int}f(x)dx} \qquad
  \sqrt{\int f(x)dx} $$
```

$$\sqrt{\smash{\int}f(x)dx} \qquad \sqrt{\int f(x)dx}$$

amsmath paketi \smash komutuna zorunlu olmayan bir argümanı daha eklemektedir. Bu argümanın t seçeneği zorunlu *formula* argümanın yüksekliğini, b seçeneği ise *formula* 'in derinliğini iptal etmektedir:

```
$$
\sqrt{\smash[b]{\int}f(x)dx}
\sqrt{\smash[t]{\int}f(x)dx}
\sqrt{\int f(x)dx}
$$
```

$$\sqrt{\smash[b]{\int}f(x)dx} \quad \sqrt{\smash[t]{\int}f(x)dx} \quad \sqrt{\int f(x)dx}$$

3.13 Kuram, önerme ve tanımlar

Bir matematik metinde teorem, lemma, önerme, tanım, aksiyom, ... gibi blokların varolması doğaldır. LaTeX bu blokların her birini özel olarak oluşturarak, onları otomatik olarak ayrı-ayrı numaralandırır ve onlara göndermeyi de otomatik oluşturur. "*Teorem*" tipli böyle bloklar LaTeX 'in aşağıdaki iki tanıtımından biriyle oluşturulabilir:

```
\newtheorem{env}[theorem]{type}
\newtheorem{env}{type}[section]
```

Bu tanıtımlar belgenin başlık kısmına yazılmalıdır. Tanıtımların *env* ve *type* argümanı zorunludur. Yeni bloğun adı *env* argümanına, bu bloğun hangi tipe ait olduğu ise *type* argümanına yazılmalıdır. Yeni blok adı LaTeX 'in varolan bir blok veya değişkeniyle (*equation*, *eqnarray*, *center*,

...) aynı olmamalıdır. Zorunlu olmayan *theorem* argümanı gösterilmemişse, LaTeX bir *env* için kendi adıyla tanımlanan bir değişken oluşturur. Eğer bu argüman gösterilmişse, onun değişkeni *env* için de kullanılır, yani *theorem* ve *env* blokları tek değişkeniyle ortak numaralandırılır. Buradan anlaşılacağı gibi, birinci komutun zorunlu olmayan *theorem* argümanında LaTeX 'in varolan bir blok veya değişkeni yazılmalıdır. Örneğin,

```
\newtheorem{onerme}[equation]{Önerme}
```

tanıtımıyla değişkeni *equation* değişkeniyle aynı olan **Önerme** adlı *onerme* bloğu tanımlanmaktadır.

İkinci `\newtheorem` tanıtımın *section* argümanında LaTeX 'de varolan bir değişken yazılmalıdır. Bu argümanda, genelde, bölümü (veya paragrafı) tanımlayan bir komut yazılmalıdır:

```
\newtheorem{uyari}{Uyarı}[subsection]
```

Bu durumda, *env* bloğun numarasına *section* değişkenin de numarası eklenir. *env* bloğun kendisi komutlu

```
\begin{env}[name]  ... \end{env}
```

paranteziyle verilmelidir. Bloğun zorunlu olmayan *name* argümanında, genelde, *theorem* 'in kime ait olduğu veya herhangi bir ek bilgi yazılabilir.

Aşağıda böyle bloklara birkaç örnek verelim.

```
\newtheorem{laws}{Kanun}[subsection]
⋮
\begin{laws} \label{Murphy}
Bir şeyin yapılması için en az iki
yol var olup, bu yolların biri
felakete götürürse, bu taktirde
elbette kimse bunu yapacaktır.
\end{laws}
Kanun \ref{Murphy} 'yi Murphy
keşfetmiştir.
```

Kanun 3.13.0.1 *Bir şeyin yapılması için en az iki yol var olup, bu yolların biri felakete götürürse, bu taktirde elbette kimse bunu yapacaktır.*

Kanun 3.13.0.1 'yi Murphy keşfetmiştir.

```
\newtheorem{theorems}{Teorem}
⋮
\begin{theorems}[Fermat]\label{Fer}
$n\in\mathbb{N}$, $n>2$ ve $x,y,z\in
\mathbb{Z}\setminus\{0\}$ için $x^n+
y^n=z^n$ denklemin çözümü yoktur
\end{theorems}
$3^2+4^2=5^2$ olduğundan teorem
\ref{Fer} $n=2$ için doğru değildir.
```

Teorem 1 (Fermat) *$n \in \mathbb{N}$, $n > 2$ ve $x, y, z \in \mathbb{Z} \setminus \{0\}$ için $x^n + y^n = z^n$ denklemin çözümü yoktur*

$3^2+4^2 = 5^2$ olduğundan teorem 1 $n = 2$ için doğru değildir.

```
\newtheorem{definitions}{Tanım}
⋮
\begin{definitions}\label{ttekfon}
$f(-x)=-f(x)$ koşulunu sağlayan $f$
'ye\textnormal{tek fonksiyon} denir.
\end{definitions}
Tanım \ref{ttekfon} 'ye göre $y=x$
fonksiyonu tektir.
```

Tanım 1 *$f(-x) = -f(x)$ koşulunu sağlayan f 'ye* tek fonksiyon *denir.*

Tanım 1 'ye göre $y = x$ fonksiyonu tektir.

Görüldüğü gibi, *env* bloğunda önce *type* argümanı koyu yazısıyla, onun numarası, eğer varsa parantez içinde bloğun *name* argümanı ve sonra bloğun metni italik yazıyla yazdırılmaktadır.

Bir matematik metinde teorem, lemma, önerme, sonuç, tanım, aksiyom, ... gibi tüm bloklar ayrı-ayrı numaralandırılmadan düz olarak tek numaralandırılabilir. Bunun için belgenin başlık kısmına aşağıdaki tanıtımlar grubu yazılmalıdır:

```
\newtheorem{theorem}{Teorem}
\newtheorem{definition}[theorem]{Tan\i m}
\newtheorem{proposition}[theorem]{\"{O}nerme}
\newtheorem{remark}[theorem]{Uyar\i }
\newtheorem{example}[theorem]{\"{O}rnek}
\newtheorem{lemma}[theorem]{Lemma}
\newtheorem{corollary}[theorem]{Sonu\c{c}}
⋮
```

LaTeX 'in `tools` bloğunun theorem paketi *env* bloğunda bazı değişikler yapılmasına izin verir. Örneğin, ayrı bir satıra ilk önce *type* argümanın numarası, sonra ise *type* 'in kendisi yazdırılabilir.
$\mathcal{AMS}$-LaTeX 'in amsthm paketi ise, yukarıdaki tanıtımın yıldızlı

```
\newtheorem*{env}{type}
```

şeklini eklemektedir. Bu durumda, *env* bloğu numaralandırılmamaktadır. LaTeX 2.09 'de "*Teorem*" tipli basit bir *env* bloğu

```
\proclaim{type}.
```

komutuyla oluşturulabilir. Bu komutun *type* argümanı koyu yazıyla yazdırılır, fakat onun numarası, gerekirse, elle yazılmalıdır. `{type}` 'den sonradaki "." (*nokta*) da önemlidir. Çünkü, LaTeX 2.09 bu noktaya kadar her şeyi koyu yazıyla yazdırır. Değişik bir ifade ile, `\proclaim` bloğunun içeriği bu noktadan sonra başlamaktadır. İçerik bittikten sonra `\par` (*bkz.* 2.3.paragraf, 46.sayfa) komutu yazılmalı veya boş satır kaldırılmalıdır. Bloğun içeriği ise *slanted* yazısıyla yazdırılmaktadır. Bunu bir örnekle gösterelim.

```
\proclaim{Teorem 1.2} (Pisagor).
Bir dik üçgenin $a$ ve $b$ dik kenar
kareleri toplamı $c$ hipotenüs kare-
sine eşittir: \(a^2+b^2=c^2\).
```

Teorem 1.2 (Pisagor). *Bir dik üçgenin a ve b dik kenar kareleri toplamı c hipotenüs karesine eşittir:* $a^2 + b^2 = c^2$.

NOT: `\proclaim` komutu sadece LaTeX 2.09 'una aittir, dolayısıyla, bu komut LaTeX 2_ε 'de çalışmayacaktır.

3.14 Denklemlerin ek numaralandırılması

amsmath paketinin `subequations` bloğu bir birine bağlı olan denklemlerin ek numaralandırılmasına izin verir. Bu ise, gerek tüm formüle bir tek gönderme ve gerekse bu denklemlerin herhangi birine ayrı olarak gönderme yapılabilmesine izin verir. Denklemler grubu $\mathcal{AMS}$-LaTeX 'in numaralandıran herhangi bir bloğuna (`eqnarray`, `gather`, ...) alınmalıdır:

```
\begin{subequations}
\begin{eqnarray}
  denklemler
\end{eqnarray}
\end{subequations}
```

`subequations` 'in içinde, fakat `equation` bloğun içinde olmayan `\label` komutu tüm denklemleri işaretlemektedir. `equation` bloğunda yazılan `\label` ise sadece bir denklemi işaretlemektedir. Bunu bir örnekle izah edelim.

```
\begin{subequations} \label{denkabc}
\begin{align}
x=r\sin\theta\cos\phi \label{denka}\\
y=r\sin\theta\sin\phi \label{denkb}\\
z=r\cos\theta         \label{denkc}
\end{align}
\end{subequations}
(\ref{denka}) denklemine göre ...
(\ref{denkb}) denklemine göre ...
(\ref{denkc}) denklemine göre ...
(\ref{denkabc}) denklemler sistemi
...
```

$$x = r\sin\theta\cos\phi \tag{3.1a}$$
$$y = r\sin\theta\sin\phi \tag{3.1b}$$
$$z = r\cos\theta \tag{3.1c}$$

(3.1a) denklemine göre ...
(3.1b) denklemine göre ...
(3.1c) denklemine göre ...
(3.1) denklemler sistemi ...

Ek numaralandırma formatı, gerekirse, değiştirilebilir. Denklemler grubunun numarası `parentequation` değişkeninde saklanır. Bir denklem numarası, her zamanki gibi, `\theequation` komutuyla yazdırılır. Fakat, o `subequations` bloğuna girerken, aşağıdaki gibi, otomatik olarak yeniden tanımlanır:

```
\renewcommand{\theequation}{\theparentequation\alph{equation}}
```

Bundan dolayı, ek numaralar Latince harfleriyle yazdırılmaktadır. Bu biçim değiştirilmesi için yukarıdaki tanıtım gerekli değişiklerle `\begin{ subequations}` 'den hemen sonra yazılmalıdır. Bunu bir örnekle gösterelim.

```
\begin{subequations} \label{dabc}
\renewcommand{\theequation}{
\theparentequation\fnsymbol{equation}}
\begin{eqnarray}
x = r\sin\theta \cos\phi \label{da}\\
y = r\sin\theta \sin\phi \label{db}\\
z = r\cos\theta \label{dc}
\end{eqnarray}
\end{subequations}
(\ref{da}) ve (\ref{dc}) 'e göre ...
(\ref{dc}) denklemine göre ...
(\ref{dabc}) denklemler sistemi ...
```

$$x = r\sin\theta\cos\phi \tag{3.2$*$}$$
$$y = r\sin\theta\sin\phi \tag{3.2$\dagger$}$$
$$z = r\cos\theta \tag{3.2$\ddagger$}$$

(3.2$*$) ve (3.2$\ddagger$) 'e göre ...
(3.2$\ddagger$) denklemine göre ...
(3.2) denklemler sistemi ...

Görüldüğü gibi, bu örnekte, ek numaralandırma için `\fnsymbol` komu-

tunun simgeleri alınmaktadır. Genel olarak, `\renewcommand{ ...` tanıtımında `\alph`, `\fnsymbol` komutları dışında `\arabic` `\roman`, `\Roman` ve `\Alph` komutları da alınabilir (*bkz.* 1.13.paragraf, 34.sayfa).

3.14.1 Denklemlerin özel numaralandırılması

amsmath paketinin `\tag{`*number*`}` komutu numaralandırılan bir denklem numarasını değiştirilmesine izin verir. Bu komut denklemden önce veya sonra `\\` (çok satırlı denklemlerde) komutuna kadar yazılmalıdır. Bundan sonra, denklem numarası yerine parantez içinde `\tag` komutunun *number* argümanı yazdırılır. `\tag` komutu `equation` değişkenin değerini değiştirmeyecektir.

Aşağıda denklemlerin `\tag` komutuyla özel numaralandırılması bir örnekle açıklanmaktadır.

```
\begin{equation}\label{eq1}
  AX = B
\end{equation}
denklemi
\begin{equation}\label{eq2}
  AX - B = 0  \tag{\ref{eq1}$'$}
\end{equation}
şeklinde yazılabilir.\eqref{eq1} veya
\eqref{eq2} 'i kullanarak \dots
```

$$AX = B \tag{3.3}$$

denklemi

$$AX - B = 0 \tag{3.3$'$}$$

şeklinde yazılabilir. (3.3) veya (3.3$'$) 'i kullanarak ...

Komutunun yıldızlı `\tag*{`*number*`}` şekli denklem numarasını parantezsiz yazdırır:

```
\begin{equation}\label{eq3}
\tag*{i-1} \sqrt{ab}\leq\frac{a+b}{2}
\end{equation}
\ref{eq3} formülünde $a,b\geq 0$
olmalıdır.
```

$$\sqrt{ab} \leq \frac{a+b}{2} \tag{i-1}$$

i-1 formülünde $a, b \geq 0$ olmalıdır.

3.15 Değişik formüller

3.15.1 Diyagramlar

$\mathcal{AMS}$-LaTeX 'in amscd paketi bir CD bloğunu tanımlamaktadır. Bu blokta köşegen yönü olmayan basit bir komütatif diyagram şöyle oluşturulabilir

```
$$
\begin{CD}
\otimes_{\alpha}A_{\alpha} @>{\pi}>>
                        A_{\alpha_0}\\
@VVV          @VV{\textrm{End}(h)}V\\
A \oplus A_{\alpha'}/K @=
               A\oplus A_{\alpha_0}/K
\end{CD}
$$
```

$$\begin{CD}
\otimes_{\alpha}A_{\alpha} @>{\pi}>> A_{\alpha_0}\\
@VVV @VV{\textrm{End}(h)}V\\
A \oplus A_{\alpha'}/K @= A\oplus A_{\alpha_0}/K
\end{CD}$$

Görüldüğü gibi, CD bloğunda @>>>, @<<<, @VVV ve @AAA komutları sırasıyla sağ yön, sol yön, alt yön ve üst yönü yazdırmaktadır. Yatay yönlerde birinci ve ikinci > veya < simgesi arasındaki ifade yöne bir üst indis olarak, ikinci ve üçüncü simgesi arasındaki ifade ise yöne bir alt indis olarak yazdırılmaktadır. Benzer şeklinde, düşey yönlerde birinci ve ikinci A veya V simgesi arasındaki ifade yönün soluna, ikinci ve üçüncü simgesi arasındaki ifade ise yönün sağına yazdırılmaktadır. Nihayet, @= komutu yatay yön uzunluğunda bir eşitlik simgesini yazdırmaktadır. Bu komutların tümü aşağıdaki diyagramda ele alınmaktadır.

```
$$
\begin{CD}
\dots @>{a}>{b}> \dots\\
@V{c}V{d}V   @V{e}V{f}V\\
\dots   @=   \dots
\end{CD}
$$
```

$$\begin{CD}
\dots @>{a}>{b}> \dots\\
@V{c}V{d}V @V{e}V{f}V\\
\dots @= \dots
\end{CD}$$

3.15.2 Çerçeveli formüller

amsmath paketinin \boxed{*formula*} komutu *formula* 'yı bir çerçeveye almaktadır:

```
Her $\{X_{\lambda}\}$ altkümeler ailesi
için $\boxed{\cap _{\lambda}X_{\lambda}
\subseteq X_{\lambda} \subseteq \cup _{
\lambda}X_{\lambda}}$ formülü doğrudur.
```

Her $\{X_\lambda\}$ alt kümeler ailesi için $\boxed{\cap_\lambda X_\lambda \subseteq X_\lambda \subseteq \cup_\lambda X_\lambda}$ formülü doğrudur.

3.15.3 Formül boyutunun değiştirilmesi

Bir formülün boyutu, normal bir metin boyutu gibi, tablo 2.3 'de tanımlanan komutların herhangi biriyle değiştirilebilir (*bkz.* 95.sayfa). Bunu bir örnekle gösterelim:

```
{\Large
 \begin{equation}
  \sum_{k=0}^\infty \frac{1}{2^k}=2
 \end{equation}
}
```

$$\sum_{k=0}^{\infty} \frac{1}{2^k} = 2 \qquad (3.4)$$

Görüldüğü gibi, formül numarası da seçilen yazı boyutuyla yazdırılmaktadır. Yukarıda ele alınan `\tag*` komutuyla (*bkz.* 3.14.1.paragraf, 184. sayfa) formül ve onun numarası farklı boyutlarda şöyle yazdırılabilir:

```
{\Large
 \begin{equation}
  \sum_{k=1}^\infty \frac{1}{2^k}=1
        \tag*{\small(\theequation)}
 \end{equation}
}
```

$$\sum_{k=1}^{\infty} \frac{1}{2^k} = 1 \qquad (3.4)$$

Aynı düşünce ile, formül numarasının boyutu değiştirilmeden formül boyutu değiştirilebilir:

```
\addtocounter{equation}{1}
{\Large
 \begin{equation}
  \sum_{k=1}^\infty \frac{1}{k}=\infty
      \tag*{\normalsize(\theequation)}
 \end{equation}
}
```

$$\sum_{k=1}^{\infty} \frac{1}{k} = \infty \qquad (3.5)$$

`\tag*` komutu `equation` değişkenin değerini değiştirmediğinden dolayı

(*bkz.* 184. sayfa) son örnekte bu değişkenin değeri `\addtocounter` komutuyla (*bkz.* 2.6. sayfa) bir birim arttırılmaktadır.

3.15.4 Ayraç boyutunun değiştirilmesi

TEX bir formülün her simgesinin boyutu gibi, ayracının (*bkz.* 147.sayfa) boyutunu da kendisi ayarlamaktadır. Fakat bazı durumlarda, TEX 'in oluşturduğu ayraç boyutu bize yeterli gelemeyebilir. Örneğin,

```
$$
\sinx=x\lim_{p\to\infty}
\prod_{k=1}^{p-1}\left[1-
\frac{x^2}{4p^2\tan^2
\displaystyle\frac{k\pi}{2p}}
\right]
$$
```

$$\sin x = x \lim_{p\to\infty} \prod_{k=1}^{p-1} \left[1 - \frac{x^2}{4p^2 \tan^2 \displaystyle\frac{k\pi}{2p}} \right]$$

formülünde `\left[` ve `\right]` komutlarıyla yapılan sırasıyla sol ve sağ ayraç gereğinden daha büyük olduğu açıktır. Aslında bir ayraç boyutunun istenilen şekilde değiştirilmesi elle yapılmalıdır. Örneğin, son örnekte `\left` ve `\right` komutları yerine sırasıyla

```
\left[\rule{0pt}{22pt}\right.  ve  \left]\rule{0pt}{22pt}\right.
```

ifadeleri yazılırsa, formüldeki sol ve sağ ayraç önceki görüntüsüne göre daha iyi gözükmektedir:

$$\sin x = x \lim_{p\to\infty} \prod_{k=1}^{p-1} \left[1 - \frac{x^2}{4p^2 \tan^2 \displaystyle\frac{k\pi}{2p}} \right] \tag{3.6}$$

Eğer bir formülde böyle ayraçlar birkaç tane ise, bu özel ayraca `\newcommand` komutuyla, aşağıdaki gibi, yeni bir ad verilmesi daha iyidir:

```
\newcommand{\ang}[1]{\raisebox{-5pt}{$\left#1\rule{0pt}{22pt}\right.$}}
```

Bu tanıtımdan sonra, (3.6) formülü metinde şöyle yazılabilir:

```
\begin{equation} \label{eqbolucu}
\sin x = x \lim _{p\to\infty}\prod _{k=1}^{p-1}
\ang{[} 1-\frac{x^2}{4p^2\tan^2\displaystyle\frac{k\pi}{2p}} \ang{]}
\end{equation}
```

Görüldüğü gibi, `\ang` komutunda `\rule` ve `\raisebox` komutlarıyla formüldeki özel ayracın yükseklik ve derinlik boyutları sırasıyla `22pt` ve `5pt` olarak tanımlanmaktadır.

Kaynakça

[1] Donald E.Knuth. The TeXbook, Volume A of Compyters and Typesetting, Addison-Wesley, Reading, Massachusetts, USA, secound edition, 1984, ISBN 0-201-13448-9

[2] Leslie Lamport. LaTeX: A Document Preparation System. Addison-Wesley, Reading, Massachusetts, USA, secound edition, 1994, ISBN 0-201-52983-1

[3] Michel Goossens, Frank Mittelbach and Alexander Samarin. The LaTeX Companion. Addison-Wesley, Reading, Massachusetts, USA, 1994. ISBN 0-201-54199-8

[4] Helmut Kopka and Patrick Daly, A Guide to LaTeXe, Addison-Wesley, Reading, Massachusetts, USA, 1995, ISBN 0-201-42777-X

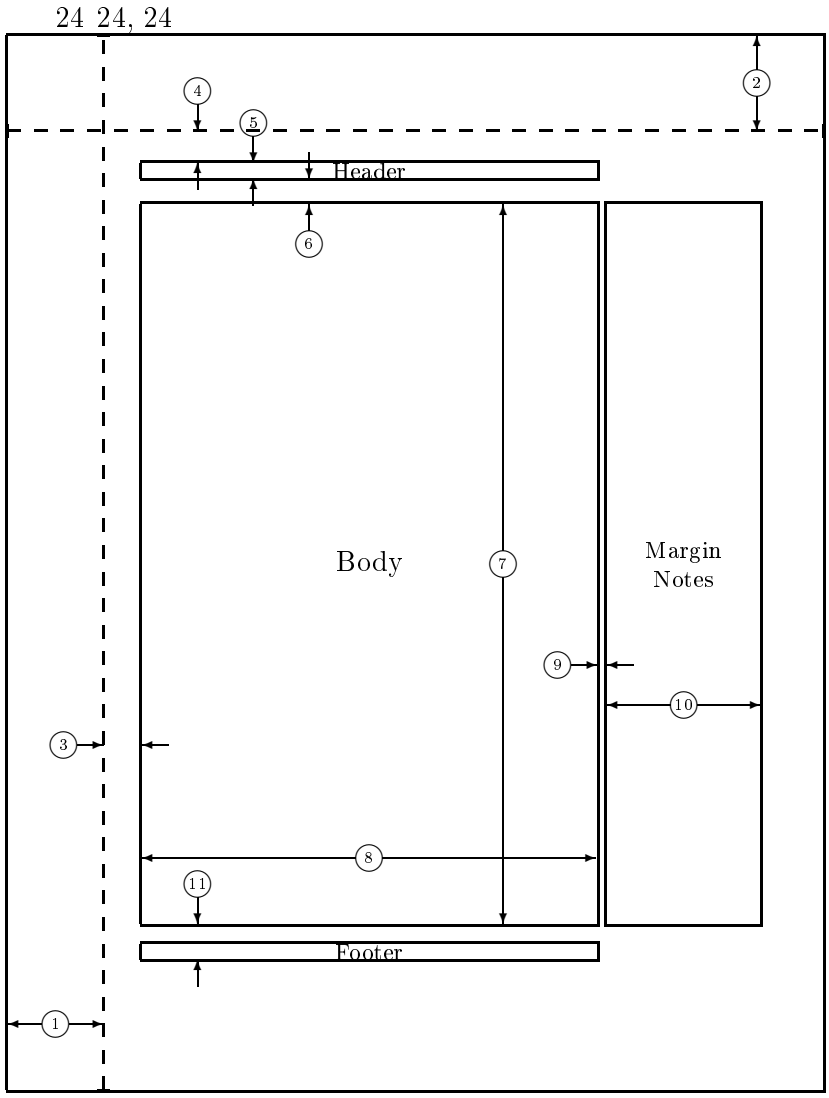
[5] Tobias Oetiker, The Not So Short Introduction to LaTeXe. Available as CTAN/info/lshort/lshort2e.pdf and CTAN/info/lshort/lshort2e.600.ps

[6] Graham Williams. *The TeX Catalogue* is a very complete listing of may TeX and LaTeX related packages. Available online from CTAN:/help/Catalogue/catalogue.html

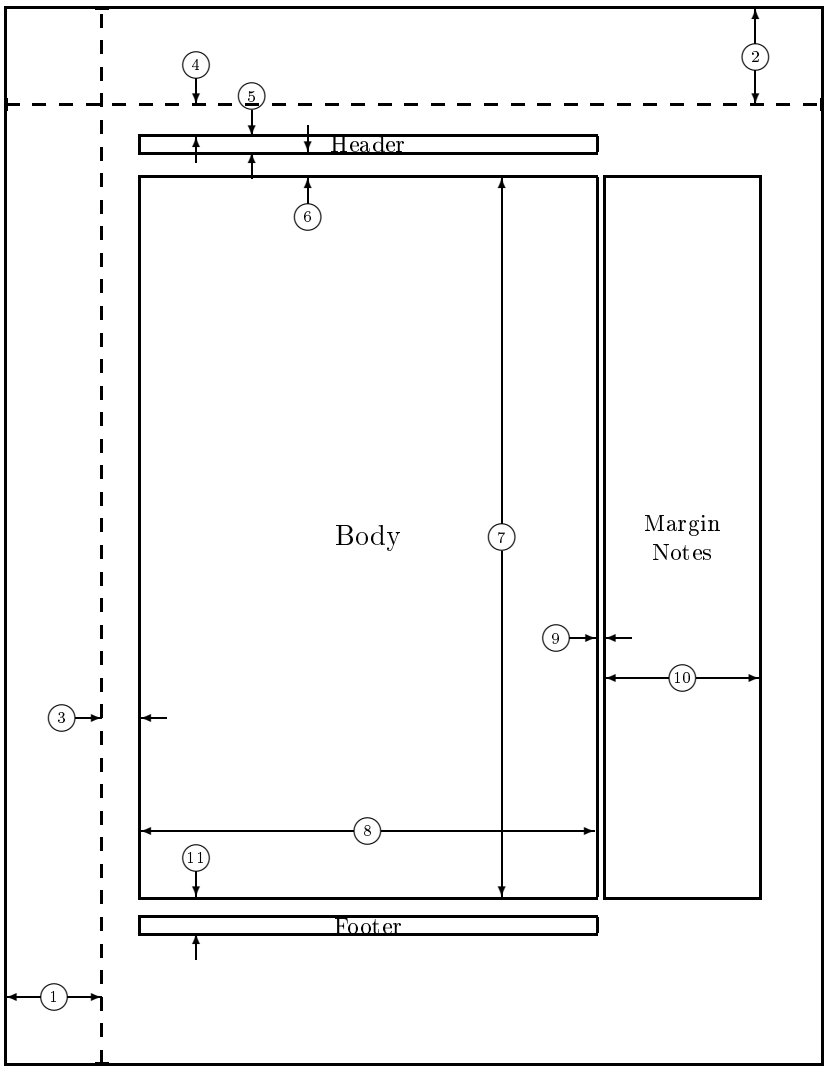
[7] Michel Goossens, Sebastian Rahtz, Eitan Gurari, Ross Moore and Robert Sutor. The LaTeX Web Companion. Addison-Wesley, Reading, Massachusetts, USA, 1999. ISBN 0-201-43311-7

[8] Sebastian Rahtz, hyperref package options, `texmf/doc/latex/hyperref/options.tex`

[9] Sebastian Rahtz, Hypertext marks in LaTeX: the hyperref package, `texmf/doc/latex/hyperref/manual.pdf`



1	one inch + \hoffset	2	one inch + \voffset
3	\oddsidemargin = 29pt	4	\topmargin = 24pt
5	\headheight = 12pt	6	\headsep = 19pt
7	\textheight = 542pt	8	\textwidth = 343pt
9	\marginparsep = 7pt	10	\marginparwidth = 116pt
11	\footskip = 27pt		\marginparpush = 5pt (not shown)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 614pt		\paperheight = 794pt



1	one inch + \hoffset	2	one inch + \voffset
3	\oddsidemargin = 29pt	4	\topmargin = 24pt
5	\headheight = 12pt	6	\headsep = 19pt
7	\textheight = 542pt	8	\textwidth = 343pt
9	\marginparsep = 7pt	10	\marginparwidth = 116pt
11	\footskip = 27pt		\marginparpush = 5pt (not shown)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 614pt		\paperheight = 794pt