

# Nie za krótkie wprowadzenie do systemu $\text{\LaTeX 2}_{\epsilon}$

---

*Albo  $\text{\LaTeX 2}_{\epsilon}$  w 129 minut*

**Tobias Oetiker**

Hubert Partl, Irene Hyna, Elisabeth Schlegl

**Tomasz Przechlewski i Ryszard Kubiak**

Janusz Gołdasz

Wydanie drugie, poprawione, uaktualnione i rozszerzone  
Oparte na wersji 4.20 *The Not So Short Introduction to  $\text{\LaTeX 2}_{\epsilon}$*   
z 31 maja 2006

**Styczeń 2007**

Copyright ©1995–2005 Tobias Oetiker and Contributors. All rights reserved.  
Copyright © 1999, 2007 for the Polish translation and extension JG, RK and TP  
All rights reserved.

This document is free; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This document is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this document; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Copyright © 1995–2005 Tobias Oetiker i wszyscy współautorzy *Wprowadzenia*.  
Copyright © 1998, 2007 polskiego tłumaczenia i opracowania JG, RK i TP.

Niniejszy dokument jest wolno dostępny; można go rozpowszechniać i/lub zmieniać zgodnie z postanowieniami Ogólnej Licencji Publicznej GNU – takiej, jaką opublikowała fundacja Free Software Foundation w wersji 2 tejże Licencji, albo (Wasz wybór) w dowolnej późniejszej.

Dokument jest rozpowszechniany w nadziei, że będzie użyteczny, jednakże BEZ ŻADNEJ GWARANCJI, nawet bez jakiegokolwiek domyślnej gwarancji WYNIKAJĄCEJ Z NABYCIA lub ODPOWIADANIA KONKRETNEMU CELOWI. Więcej szczegółów znajdziecie w Ogólnej Licencji Publicznej GNU.

Do dokumentu powinna być dołączona kopia Ogólnej Licencji Publicznej GNU; jeśli jej nie ma, to napiszcie do: the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Najnowszą polskojęzyczną wersję *Wprowadzenia* znajduje się pod adresem:  
<http://www.ctan.org/tex-archive/info/lshort/polish/>

Najnowszą anglojęzyczną wersję *Wprowadzenia* znajduje się w katalogu:  
<http://www.ctan.org/tex-archive/info/lshort>

# Podziękowania

Większość materiału w niniejszej książce pochodzi z napisanego w języku niemieckim austriackiego *Wprowadzenia do L<sup>A</sup>T<sub>E</sub>Xa 2.09*, którego autorami są:

Hubert Partl <partl@mail.boku.ac.at>

*Zentraler Informatikdienst der Universität für Bodenkultur Wien*

Irene Hyna <Irene.Hyna@bmwf.ac.at>

*Bundesministerium für Wissenschaft und Forschung Wien*

Elisabeth Schlegl <no\_email>

*in Graz*

Osoby zainteresowane wersją niemiecką, opracowaną przez Jörga Knappena do L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>, mogą ją znaleźć pod adresem [CTAN://info/lshort/german](http://CTAN://info/lshort/german).

Niżej wymienione osoby pomogły w tworzeniu *Wprowadzenia* swoimi poprawkami, sugestiami i propozycjami ulepszeń. Przyczyniły się one bardzo do nadania niniejszej książce jej obecnego kształtu. Chciałbym im za to serdecznie podziękować. Za wszelkie błędy, które nieuchronnie pozostały w tekście, ponoszę oczywiście odpowiedzialność wyłącznie ja sam. Wszystkie natomiast słowa zapisane bez błędu są w książce wyłączną zasługą osób z poniższej listy:

Rosemary Bailey, Marc Bevand, Friedemann Brauer, Jan Busa,  
Markus Brühwiler, Pietro Braione, David Carlisle, José Carlos Santos,  
Neil Carter, Mike Chapman, Pierre Chardaire, Christopher Chin, Carl Cerecke,  
Chris McCormack, Wim van Dam, Jan Dittberner, Michael John Downes,  
Matthias Dreier, David Dureisseix, Elliot, Hans Ehrbar, Daniel Flipo, David Frey,  
Hans Fugal, Robin Fairbairns, Jörg Fischer, Erik Frisk, Mic Milic Frederickx,  
Frank, Kasper B. Graversen, Arlo Griffiths, Alexandre Guimond, Andy Goth,  
Cyril Goutte, Greg Gamble, Frank Fischli, Morten Hgholm, Neil Hammond,  
Rasmus Borup Hansen, Joseph Hilferty, Björn Hvittfeldt, Martien Hulsen,  
Werner Icking, Jakob, Eric Jacoboni, Alan Jeffrey, Byron Jones, David Jones,  
Johannes-Maria Kaltenbach, Michael Koundouros, Andrzej Kawalec,  
Sander de Kievit, Alain Kessi, Christian Kern, Tobias Klauser, Jörg Knappen,  
Kjetil Kjernsmo, Maik Lehradt, Rémi Letot, Flori Lambrechts, Axel Liljencrantz,  
Johan Lundberg, Alexander Mai, Hendrik Maryns, Martin Maechler,  
Aleksandar S Milosevic, Henrik Mitsch, Claus Malten, Kevin Van Maren,

Richard Nagy, Philipp Nagele, Lenimar Nunes de Andrade, Manuel Oetiker,  
Urs Oswald, Martin Pfister, Demerson Andre Polli, Nikos Pothitos,  
Maksym Polyakov Hubert Partl, John Reffing, Mike Ressler, Brian Ripley,  
Young U. Ryu, Bernd Rosenlecher, Chris Rowley, Risto Saarelma,  
Hanspeter Schmid, Craig Schlenter, Gilles Schintgen, Baron Schwartz,  
Christopher Sawtell, Miles Spielberg, Geoffrey Swindale, Laszlo Szathmary,  
Boris Tobotras, Josef Tkadlec, Scott Veirs, Didier Verna, Fabian Wernli,  
Carl-Gustav Werner, David Woodhouse, Chris York, Fritz Zaucker, Rick Zacccone,  
oraz Mikhail Zotov.

### **Od autorów polskiej wersji**

Za pomoc redakcyjną dziękujemy Staszкови Wawrykiewiczowi i Włodkowi Macewiczowi.

# Przedmowa

L<sup>A</sup>T<sub>E</sub>X [12] jest systemem składu znakomicie nadającym się do tworzenia publikacji naukowych i technicznych o wysokiej jakości typograficznej. L<sup>A</sup>T<sub>E</sub>X nadaje się również do przygotowywania dowolnego rodzaju dokumentów, poczynając od prostych listów, a kończąc na grubych książkach. Do formatowania dokumentów L<sup>A</sup>T<sub>E</sub>X wykorzystuje program T<sub>E</sub>X [11, 5].

Niniejsze krótkie *Wprowadzenie* opisuje L<sup>A</sup>T<sub>E</sub>Xa w zakresie wystarczającym do większości zastosowań. Pełny opis L<sup>A</sup>T<sub>E</sub>Xa można znaleźć w [12, 6, 4].

*Wprowadzenie* zawiera sześć następujących rozdziałów:

**Rozdział 1** przedstawia ogólną strukturę dokumentów L<sup>A</sup>T<sub>E</sub>Xowych i krótko omawia historię systemu L<sup>A</sup>T<sub>E</sub>X. Po przeczytaniu tego rozdziału powinieneś mieć już zgrubne wyobrażenie o tym, czym jest L<sup>A</sup>T<sub>E</sub>X.

**Rozdział 2** podaje szczegóły dotyczące składania dokumentów. Omówiono w nim najważniejsze instrukcje i otoczenia. Po przeczytaniu tego rozdziału będziesz już umiał tworzyć proste dokumenty L<sup>A</sup>T<sub>E</sub>Xowe.

**Rozdział 3** poświęcono składaniu wzorów matematycznych. Wiele zamieszczonych w nim przykładów nauczy Cię wykorzystywać jedną z najważniejszych umiejętności T<sub>E</sub>Xa, jaką jest elegancki skład matematyki. Na końcu rozdziału zamieszczono zestawienie dostępnych w L<sup>A</sup>T<sub>E</sub>Xu symboli matematycznych.

**Rozdział 4** wyjaśnia, jak tworzyć skorowidze i spisy bibliograficzne oraz jak dołączać rysunki w formacie EPS. Znajdziesz tu również informacje o tworzeniu plików w formacie PDF za pomocą programu `pdflatex` oraz o kilku użytecznych pakietach L<sup>A</sup>T<sub>E</sub>Xowych.

**Rozdział 5** pokazuje, jak używać L<sup>A</sup>T<sub>E</sub>Xa do tworzenia grafiki. Zamiast przygotowywać rysunek w jakimś programie graficznym, zachować go w pliku, po czym włączyć do dokumentu L<sup>A</sup>T<sub>E</sub>Xowego, opisujesz rysunek w dokumencie, a do jego narysowania używasz samego L<sup>A</sup>T<sub>E</sub>Xa.

**Rozdział 6** zawiera informacje, których wykorzystanie jest potencjalnie niebezpieczne, bo mówią o tym, jak można zmienić standardowy układ graficzny dokumentów L<sup>A</sup>T<sub>E</sub>Xowych. Niewłaściwe ich użycie może spowodować pogorszenie ładnego na ogół składu L<sup>A</sup>T<sub>E</sub>X-owego.

Sądzymy, że powinieneś przeczytać wszystkie rozdziały, w powyższej kolejności. Ostatecznie, książka nie jest zbyt gruba. Szczególną uwagę zwróć na przykłady, gdyż właśnie w nich zawarto sporo wartościowych informacji.

L<sup>A</sup>T<sub>E</sub>X jest dostępny na większości platform sprzętowych, począwszy od PC i Macintosh, a skończywszy na dużych systemach wyposażonych w system UNIX czy VMS. W wielu sieciach uniwersyteckich można spotkać gotowe do użytku instalacje L<sup>A</sup>T<sub>E</sub>Xa. Informacje, jak rozpocząć pracę w lokalnej instalacji L<sup>A</sup>T<sub>E</sub>Xa, można znaleźć w [16]. Jeżeli nie wiesz, jak zacząć pracę z L<sup>A</sup>T<sub>E</sub>Xem, to zapytaj się osoby, od której otrzymałeś niniejsze *Wprowadzenie*. W tym dokumencie nie poruszamy spraw związanych z instalowaniem i konfigurowaniem systemu L<sup>A</sup>T<sub>E</sub>X. Mówimy wyłącznie o tym, jak pisać dokumenty, aby mogły być przetwarzane przez L<sup>A</sup>T<sub>E</sub>Xa.

Gdybyś potrzebował jakichkolwiek materiałów dotyczących L<sup>A</sup>T<sub>E</sub>Xa, to zajrzyj do jednego z archiwów sieci CTAN. Główny węzeł CTAN ma adres <http://www.ctan.org>. Można też korzystać z innych węzłów, takich jak <http://www.dante.de> lub <ftp://ftp.dante.de> albo dowolnego z wielu archiwów lustrzanych, na przykład polskiego <ftp://ftp.gust.org.pl>.

Wszystkie zasoby w każdym archiwum CTAN znajdują się w katalogu `tex-archive`. W dalszej części książki będziemy wielokrotnie używać odsyłaczy do zasobów CTANu, ale będziemy w nich pomijać protokół, adres konkretnego węzła oraz początkowy katalog `tex-archive`, to znaczy będziemy używać zapisu `CTAN://ścieżka-do-zasobu` zamiast <http://www.ctan.org/tex-archive/ścieżka-do-zasobu>.

W katalogu `CTAN://systems` znajdziesz oprogramowanie niezbędne do uruchomienia L<sup>A</sup>T<sub>E</sub>Xa na twoim komputerze.

## Od autorów polskiej wersji

Ponieważ uznaliśmy, że niektóre tematy autorzy *Wprowadzenia* przedstawili zbyt lakonicznie, zdecydowaliśmy się na ich poszerzenie. Zmiany w stosunku do oryginału dotyczą w szczególności: zagadnień przygotowywania dokumentów w języku polskim (punkt 2.5 i wiele uzupełnień w innych punktach), opisu klasy letter (punkt 2.14) oraz informacji dotyczących tworzenia plików PDF (punkt 4.7). Ponieważ w naszej opinii w oryginale *Wprowadzenia* zdecydowanie za mało miejsca zajmuje ważne zagadnienie składania tabel, dodaliśmy poświęcony w całości temu tematowi obszerny punkt 6.8.

Jeżeli masz pomysł, co należałoby dodać, usunąć lub zmienić w tym dokumencie bądź jego tłumaczeniu – napisz. Jesteśmy szczególnie zainteresowani opiniami początkujących użytkowników L<sup>A</sup>T<sub>E</sub>Xa o tym, które fragmenty trudno zrozumieć i wymagają lepszego przedstawienia.

Ryszard Kubiak, [R.Kubiak@guests.ipipan.gda.pl](mailto:R.Kubiak@guests.ipipan.gda.pl)

Tomasz Przechlewski, [T.Przechlewski@GUST.org.pl](mailto:T.Przechlewski@GUST.org.pl)

Z autorem oryginalnej wersji angielskiej możesz się skontaktować, pisząc pod adresem: Tobias Oetiker ([oetiker@ee.ethz.ch](mailto:oetiker@ee.ethz.ch)).

Polecamy stronę <http://www.gust.org.pl> Polskiej Grupy Użytkowników Systemu T<sub>E</sub>X GUST jako dodatkowe, bogate źródło informacji o T<sub>E</sub>Xu.

# Spis treści

<b>Podziękowania</b>	<b>iii</b>
<b>Przedmowa</b>	<b>v</b>
<b>1. Podstawy, które warto znać</b>	<b>1</b>
1.1. Nazwa programu . . . . .	1
1.1.1. $\text{\TeX}$ . . . . .	1
1.1.2. $\text{\LaTeX}$ . . . . .	1
1.2. Podstawy . . . . .	2
1.2.1. Autor, redaktor i zecer . . . . .	2
1.2.2. Układ graficzny . . . . .	3
1.2.3. Zalety i wady . . . . .	3
1.3. Plik źródłowy . . . . .	4
1.3.1. Odstępy . . . . .	4
1.3.2. Znaki specjalne . . . . .	5
1.3.3. Polecenia $\text{\LaTeX}$ a . . . . .	5
1.3.4. Komentarze . . . . .	6
1.4. Struktura pliku źródłowego . . . . .	6
1.5. Typowa sesja pracy z $\text{\LaTeX}$ em . . . . .	7
1.6. Układ graficzny dokumentu . . . . .	9
1.6.1. Klasy dokumentów . . . . .	9
1.6.2. Pakiety . . . . .	10
1.6.3. Style strony . . . . .	11
1.7. Nazwy plików związanych z $\text{\LaTeX}$ em . . . . .	12
1.8. Duże dokumenty . . . . .	14
<b>2. Składanie tekstu</b>	<b>15</b>
2.1. Struktura tekstu i języka . . . . .	15
2.2. Składanie akapitów i łamanie stron . . . . .	17
2.2.1. Składanie akapitów . . . . .	17
2.2.2. Przenoszenie wyrazów . . . . .	19
2.3. Kilka gotowych oznaczeń napisów . . . . .	20
2.4. Znaki specjalne i symbole . . . . .	21

2.4.1.	Cudzysłowy . . . . .	21
2.4.2.	Pauzy i myślniki . . . . .	21
2.4.3.	Odstępy niełamliwe . . . . .	22
2.4.4.	Tylda ( $\sim$ ) . . . . .	23
2.4.5.	Oznaczenie stopni ( $\circ$ ) . . . . .	23
2.4.6.	Symbol waluty euro ( $\text{€}$ ) . . . . .	23
2.4.7.	Wielokropek ( $\dots$ ) . . . . .	24
2.4.8.	Ligatury . . . . .	24
2.4.9.	Akcenty i znaki specjalne . . . . .	25
2.5.	L <sup>A</sup> T <sub>E</sub> X wielojęzyczny . . . . .	25
2.5.1.	Język polski w dokumentach . . . . .	28
2.6.	Odstępy między wyrazami . . . . .	30
2.7.	Tytuły, śródtytuły i punkty . . . . .	31
2.8.	Odsyłacze . . . . .	34
2.9.	Przypisy . . . . .	34
2.10.	Wyróżnienia . . . . .	35
2.11.	Otoczenia . . . . .	35
2.11.1.	Otoczenia <code>itemize</code> , <code>enumerate</code> i <code>description</code> . . . . .	36
2.11.2.	Otoczenia <code>flushleft</code> , <code>flushright</code> i <code>center</code> . . . . .	36
2.11.3.	Otoczenia <code>quote</code> , <code>quotation</code> i <code>verse</code> . . . . .	37
2.11.4.	Streszczenie . . . . .	37
2.11.5.	Symulacja maszynopisu . . . . .	38
2.11.6.	Otoczenie <code>tabular</code> . . . . .	38
2.12.	Wstawki . . . . .	40
2.13.	Ochrona poleceń kruchych . . . . .	43
2.14.	Listy . . . . .	43
<b>3.</b>	<b>Wyrażenia matematyczne</b> . . . . .	<b>45</b>
3.1.	Wstęp . . . . .	45
3.2.	Grupowanie . . . . .	47
3.3.	Części składowe wyrażeń matematycznych . . . . .	47
3.4.	Odstępy w trybie matematycznym . . . . .	51
3.5.	Wyrównywanie w pionie . . . . .	52
3.6.	Fantomy . . . . .	54
3.7.	Stopień pisma . . . . .	55
3.8.	Twierdzenia, definicje, itp. . . . .	56
3.9.	Symbole półgrube . . . . .	57
3.10.	Zestawienie symboli matematycznych . . . . .	58
<b>4.</b>	<b>Rysunki, skorowidze, generowanie plików PDF...</b> . . . . .	<b>65</b>
4.1.	Włączanie grafiki w formacie EPS . . . . .	65
4.2.	Spis literatury . . . . .	68
4.3.	Skorowidze . . . . .	69
4.4.	Paginy górne i dolne . . . . .	71



4.5. Pakiet <code>verbatim</code> . . . . .	72
4.6. Instalowanie dodatkowych pakietów . . . . .	72
4.7. Tworzenie plików PDF: <code>pdfL<sup>A</sup>T<sub>E</sub>X</code> i <code>hyperref</code> . . . . .	74
4.7.1. Tworzenie dokumentów PDF w <code>L<sup>A</sup>T<sub>E</sub>Xu</code> . . . . .	74
4.7.2. Fonty bitmapowe i obwiedniowe . . . . .	75
4.7.3. Dołączanie grafiki . . . . .	76
4.7.4. Łączy hipertekstowe . . . . .	77
4.7.5. Problemy z łączami . . . . .	79
4.7.6. Problemy z zakładkami . . . . .	79
4.7.7. Interpretacja pliku źródłowego przez <code>L<sup>A</sup>T<sub>E</sub>Xa</code> i <code>pdfL<sup>A</sup>T<sub>E</sub>Xa</code> . . . . .	80
4.7.8. Wymiary kartki papieru . . . . .	81
4.8. Tworzenie prezentacji . . . . .	81
4.9. Pakiet <code>pdfscreen</code> . . . . .	83
<b>5. Tworzenie grafiki matematycznej</b> . . . . .	<b>84</b>
5.1. Przegląd . . . . .	84
5.2. Otoczenie <code>picture</code> . . . . .	85
5.2.1. Podstawowe polecenia . . . . .	85
5.2.2. Odcinki . . . . .	86
5.2.3. Strzałki . . . . .	87
5.2.4. Okręgi . . . . .	88
5.2.5. Tekst i wzory . . . . .	89
5.2.6. Polecenia <code>\multiput</code> i <code>\linethickness</code> . . . . .	89
5.2.7. Owale . . . . .	90
5.2.8. Wielokrotne użycie pudełek z rysunkami . . . . .	91
5.2.9. Krzywe Béziera drugiego stopnia . . . . .	92
5.2.10. Krzywe łańcuchowe . . . . .	93
5.2.11. Prędkość w Szczególnej Teorii Względności . . . . .	94
5.3. <code>X<sub>y</sub>-pic</code> . . . . .	94
<b>6. Adaptowanie <code>L<sup>A</sup>T<sub>E</sub>Xa</code></b> . . . . .	<b>98</b>
6.1. Definiowane instrukcje i otoczeń . . . . .	98
6.1.1. Instrukcje definiowane przez użytkownika . . . . .	99
6.1.2. Otoczenia definiowane przez użytkownika . . . . .	100
6.1.3. Nadmiarowe odstępy . . . . .	101
6.1.4. Własne pakiety . . . . .	101
6.2. Fonty . . . . .	102
6.2.1. Instrukcje przełączające stopień pisma . . . . .	102
6.2.2. Uwaga, niebezpieczeństwo! . . . . .	105
6.2.3. Użycie alternatywnych krojów pisma . . . . .	105
6.3. Odstępy . . . . .	107
6.3.1. Zmiana wielkości interlinii . . . . .	107
6.3.2. Odstępy wokół akapitów . . . . .	108
6.3.3. Odstępy poziome . . . . .	108

6.3.4. Odstępy pionowe . . . . .	110
6.4. Układ graficzny strony . . . . .	111
6.5. Więcej o odległościach . . . . .	113
6.6. Pudelka . . . . .	113
6.7. Kreski i podpory . . . . .	115
6.8. Więcej o składaniu tabel . . . . .	116
6.8.1. Tabele o zadanej szerokości . . . . .	116
6.8.2. Pakiet <code>longtable</code> . . . . .	119
6.8.3. Pakiet <code>array</code> . . . . .	120
6.8.4. Pakiet <code>tap</code> . . . . .	122
<b>Bibliografia</b>	<b>123</b>
<b>Skorowidz</b>	<b>125</b>

# Spis rysunków

1.1. Zawartość minimalnego pliku źródłowego . . . . .	7
1.2. Przykład artykułu do czasopisma . . . . .	8
4.1. Przykład wykorzystania pakietu <code>fancyhdr</code> . . . . .	71
4.2. Prosty kod dla klasy <code>beamer</code> . . . . .	82
6.1. Przykładowy pakiet . . . . .	102
6.2. Parametry układu graficznego strony . . . . .	112

# Spis tabel

1.1. Wybrane pakiety z podstawowej dystrybucji $\text{\LaTeX}$	12
2.1. Torba pełna symboli euro	24
2.2. Akcenty i znaki specjalne	25
2.3. Opcjonalny argument otoczeń <code>table</code> i <code>figure</code>	41
3.1. Akcenty matematyczne	58
3.2. Litery alfabetu greckiego	58
3.3. Symbole relacji	59
3.4. Symbole operacji dwuargumentowych	60
3.5. Symbole zmiennej wielkości	60
3.6. Strzałki	60
3.7. Ograniczniki	61
3.8. Duże ograniczniki	61
3.9. Różne symbole	61
3.10. Symbole niematematyczne	61
3.11. Ograniczniki (pakiet <code>AMS</code> )	61
3.12. Symbole Greckie i Hebrajskie (pakiet <code>AMS</code> )	61
3.13. Symbole relacji (pakiet <code>AMS</code> )	62
3.14. Strzałki (pakiet <code>AMS</code> )	62
3.15. Negacje symbolów relacji i strzałek (pakiet <code>AMS</code> )	63
3.16. Relacje dwuargumentowe (pakiet <code>AMS</code> )	63
3.17. Różne symbole (pakiet <code>AMS</code> )	64
3.18. Kroje pisma dostępne w trybie matematycznym	64
4.1. Znaczenie ważniejszych kluczy polecenia <code>\includegraphics</code>	66
4.2. Przykłady składni polecenia <code>\index</code>	70
6.1. Polecenia wyboru krojów i odmian	103
6.2. Polecenia jednoczesnego wyboru stopnia pisma i interlinii	103
6.3. Wielkość stopnia pisma w klasach standardowych	104
6.4. Polecenia wyboru fontów w trybie matematycznym	105
6.5. $\text{\LaTeX}$ owe jednostki miary	109
6.6. Tytuł tabeli	119

# Rozdział 1

## Podstawy, które warto znać

*W pierwszej części tego rozdziału przedstawimy krótko filozofię oraz historię systemu  $\text{\LaTeX}$  2 $\epsilon$ . W części drugiej skoncentrujemy się na podstawowych elementach dokumentu  $\text{\LaTeX}$ owego. Po przeczytaniu tego rozdziału czytelnik powinien z grubsza wiedzieć, jak działa  $\text{\LaTeX}$ , co jest niezbędne do rozumienia materiału prezentowanego w następnych rozdziałach.*

### 1.1. Nazwa programu

#### 1.1.1. $\text{\TeX}$

$\text{\TeX}$  jest programem komputerowym stworzonym przez Donalda E. Knutha [11]. Jest przeznaczony do składu tekstów oraz wzorów matematycznych. Knuth rozpoczął pracę nad  $\text{\TeX}$ em w 1977 roku, aby wykorzystać potencjał składu cyfrowego, stosowanego wówczas na coraz szerszą skalę w poligrafii. Miał też nadzieję, że uda się odwrócić tendencję do pogarszania się jakości typograficznej, co uwiadamiało się w jego własnych książkach i artykułach. W używanej obecnie postaci  $\text{\TeX}$  został udostępniony w roku 1982, a niewielkie rozszerzenie, dotyczące ośmiobitowego kodowania znaków, pojawiło się w roku 1989.  $\text{\TeX}$  ma renomę programu nadzwyczaj stabilnego, pracującego na różnego rodzaju sprzęcie oraz praktycznie wolnego od błędów. Numery wersji  $\text{\TeX}$ a zbiegają do liczby  $\pi$ , a obecny wynosi 3,14159.

Słowo  $\text{\TeX}$  należy wymawiać „tech”. Zgłoska „ch” ma związek z tym, że znak X przypomina grecką literę „chi”.  $\text{\TeX}$  jest też pierwszą sylabą greckiego słowa *tecnologia* (technologia). W sytuacjach, w których nie można zapisywać nazwy  $\text{\TeX}$  z charakterystycznym obniżeniem litery E, należy zamiennie używać wersji *TeX*.

#### 1.1.2. $\text{\LaTeX}$

$\text{\LaTeX}$  jest zestawem instrukcji (poleceń, makrodefinicji, makr) umożliwiającym autorom złożenie i wydrukowanie ich prac na najwyższym poziomie

typograficznym. Pierwszą wersję  $\text{\LaTeX}$ a opracował Leslie Lamport [12]. Do formatowania dokumentu  $\text{\LaTeX}$  używa programu  $\text{\TeX}$ . Pielęgnacją dzisiejszych wersji  $\text{\LaTeX}$ a zajmuje się Frank Mittelbach.

Kilka lat temu pakiet  $\text{\LaTeX}$  został rozszerzony przez zespół o nazwie  $\text{\LaTeX3}$  ( *$\text{\LaTeX3}$  team*), kierowany przez Franka Mittelbacha. Celem tego rozszerzenia było wprowadzenie kilku od dawna postulowanych ulepszeń oraz unifikacja rozmaitych odmian  $\text{\LaTeX}$ a, które rozpowszechniły się od chwili powstania kilkanaście lat wcześniej  $\text{\LaTeX}$ a w wersji 2.09. Nową wersję pakietu nazwano  $\text{\LaTeX} 2_{\epsilon}$ , w celu odróżnienia jej od wersji poprzednich. Niniejszy dokument opisuje właśnie  $\text{\LaTeX} 2_{\epsilon}$ .

Słowo  $\text{\LaTeX}$  należy wymawiać „lej-tech” albo „la-tech.” Jeżeli nie można zapisać symbolu  $\text{\LaTeX}$ , to zamiennie należy użyć zapisu  $\text{LaTeX}$ .  $\text{\LaTeX} 2_{\epsilon}$  wymawiamy „la-tech dwa i”, a zamienną wersją zapisu jest  $\text{LaTeX2e}$ .

## 1.2. Podstawy

### 1.2.1. Autor, redaktor i zecer

Aby wydać książkę, autor dostarcza maszynopis do wydawnictwa. W wydawnictwie redaktor decyduje o układzie graficznym dokumentu (szerokość szpalty, krój pisma, odstępy przed i po tytułach rozdziałów itd.). Redaktor zapisuje swoje decyzje w maszynopisie, w formie odpowiednich instrukcji, i przekazuje go zecerowi. Na podstawie maszynopisu oraz instrukcji zecer wykonuje skład.

Redaktor-człowiek odgaduje, co miał na myśli autor, gdy zapisywał maszynopis. Wykorzystując swoje doświadczenie zawodowe, ustala, które miejsca w maszynopisie oznaczają tytuły rozdziałów, podrozdziałów, cytaty, przykłady, wzory matematyczne itd.

$\text{\LaTeX}$  gra rolę redaktora, a  $\text{\TeX}$  – zecera. Z tym że  $\text{\LaTeX}$  jest „zaledwie” programem komputerowym i dlatego potrzebuje dodatkowej pomocy autora, który powinien dostarczyć niezbędnych do składu informacji o strukturze logicznej dokumentu. Informacje te autor zapisuje w pliku źródłowym dokumentu jako „polecenia dla  $\text{\LaTeX}$ a”.

Praca z  $\text{\LaTeX}$ em zdecydowanie różni się od podejścia stosowanego w procesorach tekstu typu WYSIWYG<sup>1</sup>, takich jak MS Word albo OpenOffice. Pierwszy sposób można określić „formatowaniem logicznym” drugi – „formatowaniem wizualnym”. Używając programów typu WYSIWYG, autor decyduje interakcyjnie o wyglądzie graficznym dokumentu, w miarę dopisywania tekstu. Przez cały czas widzi na ekranie, jak tekst będzie wyglądał po wydrukowaniu.

Używając  $\text{\LaTeX}$ a, nie można na ogół oglądać dokumentu w jego ostatecznej postaci i jednocześnie dopisywać tekstu. Można natomiast obejrzeć

---

<sup>1</sup>ang. *What you see is what you get* (dostaniesz to, co widzisz).

dokument na ekranie po przetworzeniu go  $\text{\LaTeX}$ em<sup>2</sup>. Gdy jest już gotowy, można dokument wysłać na drukarkę.

### 1.2.2. Układ graficzny

Projektowanie książek jest sztuką. Amatorzy często popełniają poważny błąd, zakładając, że zaprojektowanie układu graficznego książki jest jedynie kwestią estetyki (jeżeli dokument ładnie wygląda, to jest dobrze złożony). Ponieważ dokumenty są przeznaczone do czytania, a nie do powieszenia, jak obraz na ścianie w galerii, to o wiele większe znaczenie niż piękny wygląd ma łatwość czytania i przyswajania tekstu. Przykłady:

- stopień pisma oraz numerację rozdziałów, podrozdziałów i punktów należy ustalić tak, by czytelnik mógł się szybko zorientować w strukturze dokumentu;
- szerokość szpalty powinna być na tyle wąska, by czytelnik nie musiał wyęczać wzroku, wystarczająco jednak duża, aby tekst elegancko wypełniał stronę.

W systemach wizualnych często powstają dokumenty przyjemne dla oka, chociaż pozbawione struktury albo wykazujące brak konsekwencji w strukturze.  $\text{\LaTeX}$  zapobiega powstawaniu takich błędów, nakłaniając autora do określenia *logicznej* struktury dokumentu. Do  $\text{\LaTeX}$ a należy dobór najodpowiedniejszego dla niej układu graficznego.

### 1.2.3. Zalety i wady

Tematem często dyskutowanym, gdy użytkownicy programów typu WYSIWYG spotykają użytkowników  $\text{\LaTeX}$ a, są „zalety  $\text{\LaTeX}$ a w porównaniu ze zwykłym procesorem tekstu” albo na odwrót. Najlepiej podczas takich dyskusji siedzieć cicho. Czasami jednak nie ma ucieczki. . .

Na wszelki wypadek trochę amunicji. Oto najważniejsze zalety  $\text{\LaTeX}$ a w porównaniu ze zwykłymi procesorami tekstu:

- Dostępne są gotowe, przygotowane przez zawodowców szablony, dzięki zastosowaniu których dokumenty wyglądają „jak z drukarni”.
- Wygodnie składa się wzory matematyczne.
- Do rozpoczęcia pracy wystarczy poznać zaledwie kilkanaście łatwych do zrozumienia instrukcji, określających strukturę logiczną dokumentu. Nie trzeba zaprzętać sobie głowy formatowaniem dokumentu.
- Nawet takie elementy jak: przypisy, odnośniki, spisy treści, spisy tabel, skorowidze oraz spisy bibliograficzne przygotowuje się bardzo łatwo.

---

<sup>2</sup>Na szybkim komputerze przetworzenie trwa często zaledwie kilka sekund. Dysponując dużym ekranem, można jednocześnie wyświetlić okno z plikiem źródłowym oraz okno podglądu, otrzymując w ten sposób system prawie WYSIWYG.

- Istnieje wiele bezpłatnych pakietów poszerzających typograficzne możliwości  $\text{\LaTeX}$ a. Dostępne są na przykład pakiety umożliwiające wstawianie do dokumentów grafiki w formacie Postscript, tworzenie hipertekstowej wersji dokumentów w formacie PDF czy też przygotowanie spisów bibliograficznych według ściśle określonych reguł, obowiązujących w różnych wydawnictwach. Opis wielu z tych pakietów można znaleźć w podręczniku [6].
- $\text{\LaTeX}$  zachęca autorów do tworzenia dokumentów o dobrze określonej strukturze.
- $\text{\TeX}$  – program formatujący używany przez  $\text{\LaTeX}$  2 $\epsilon$  – jest bezpłatny i w najwyższym stopniu przenośny. Dzięki temu można działać na praktycznie dowolnej platformie systemowo-sprzętowej.

$\text{\LaTeX}$  ma także pewne wady, chociaż ciężko mi znaleźć jakąkolwiek istotną. Jestem jednak pewien, że inne osoby wskażą ci ich setki; –)

- $\text{\LaTeX}$  nie działa u tych, którzy zaprzędali swoje dusze...
- Chociaż przez zmianę niektórych parametrów można dostosowywać predefiniowane układy graficzne do własnych potrzeb, to jednak zaprojektowanie całkowicie nowego układu jest pracochłonne<sup>3</sup>.
- Trudno jest tworzyć dokumenty o nieokreślonej, bałaganiarskiej strukturze.
- Twój chomik może nie być w stanie, mimo kilku obiecujących kroków wstępnych, w pełni pojąć koncepcję znakowania logicznego.

### 1.3. Plik źródłowy

Plik źródłowy  $\text{\LaTeX}$ a to zwykły plik tekstowy (plik ASCII). Taki plik można utworzyć w dowolnym edytorze tekstowym. Powinien on zawierać tekst dokumentu oraz instrukcje dla  $\text{\LaTeX}$ a określające, jak tekst ma zostać złożony.

#### 1.3.1. Odstępy

Znaki *niewidoczne*, takie jak odstępy (spacje) lub znaki tabulacji, są przez  $\text{\LaTeX}$ a traktowane jednakowo – jako odstęp. *Kolejno* po sobie występujące znaki odstępu  $\text{\LaTeX}$  traktuje jak *pojedynczy* odstęp. Znaki odstępu znajdujące się na początku wiersza są prawie zawsze ignorowane. Pojedynczy koniec linii jest traktowany jak odstęp.

Pusty wiersz pomiędzy dwoma wierszami tekstu oznacza koniec akapitu. Kolejno występujące puste wiersze  $\text{\LaTeX}$  traktuje jak jeden. Przykładem może być poniższy tekst. Po prawej stronie (w ramce) przedstawiono wynik składu, a po lewej – zawartość pliku źródłowego.

<sup>3</sup>Plotki mówią, że jest to jeden z ważniejszych problemów, nad jakim pracują twórcy systemu  $\text{\LaTeX}$ 3.



Nie ma znaczenia, czy między słowami  
jest jedna czy więcej spacji.

Pusty wiersz zakończył poprzedni  
akapit.

Nie ma znaczenia, czy między słowami jest  
jedna czy więcej spacji.  
Pusty wiersz zakończył poprzedni akapit.

### 1.3.2. Znaki specjalne

Niektóre znaki są zarezerwowane – w tym sensie, że albo mają dla  $\text{\LaTeX}$ a specjalne znaczenie, albo nie są dostępne we wszystkich standardowych krojach pisma. Użyte dosłownie w pliku źródłowym nie pojawią się na wydruku, lecz (najczęściej) spowodują błąd podczas przetwarzania tekstu. Oto ich lista:

`$ & % # _ { } ~ ^ \`

Znaki te można umieścić w dokumencie pod warunkiem, że w pliku źródłowym zostaną poprzedzone znakiem w-tył-ciacha (ang. *backslash*):

`\$ \& \% \# \_ \{ \}`

`$ & % # _ { }`

Samemu znaku w-tył-ciach *nie można* wstawić do tekstu metodą podwojenia, kombinacja `\\` jest bowiem poleceniem  $\text{\LaTeX}$ a, opisanym w punkcie 2.2.1. Znak w-tył-ciach można wstawić poleceniem `\backslash`. Uwaga: znaki dolara są tu niezbędne, a ich opuszczenie spowoduje błąd podczas przetwarzania.

### 1.3.3. Polecenia $\text{\LaTeX}$ a

Polecenia  $\text{\LaTeX}$ a mogą wystąpić w dwóch następujących odmianach:

1. Instrukcji rozpoczynających się znakiem w-tył-ciach „`\`”, po którym występuje ciąg liter. Końcem instrukcji jest wówczas odstęp lub inny znak niebędący literą. W nazwach instrukcji  $\text{\LaTeX}$  rozróżnia litery małe i duże, nie można też w nich używać polskich liter diakrytycznych.
2. Instrukcji składających się ze znaku w-tył-ciach oraz *jednego* znaku niebędącego literą.

$\text{\LaTeX}$  ignoruje znaki niewidoczne po instrukcji typu 1. Jeżeli po instrukcji ma występować w dokumencie odstęp, to należy bezpośrednio po niej umieścić kolejno: parę nawiasów klamrowych `{ }` i odstęp. Para znaków `{ }` zapobiega zignorowaniu przez  $\text{\LaTeX}$ a odstepu po nazwie instrukcji. Innym sposobem jest wstawienie specjalnej instrukcji `\` (tj. w-tył-ciach i spacja). Niektórzy zapobiegają „połykaniu” spacji w jeszcze inny sposób, a mianowicie otaczając nazwę instrukcji parą nawiasów `{ i }`.

Czytałem, że Knuth dzieli  
użytkowników systemu `{\TeX}` na  
`\TeX{}`ników oraz `\TeX` pertów.\\  
Dzisiaj jest `\today`.

Czytałem, że Knuth dzieli użytkowników sys-  
temu `\TeX` na `\TeX`ników oraz `\TeX`pertów.  
Dzisiaj jest 15 stycznia 2007.

Niektóre instrukcje `\LaTeX`owe mają argumenty. Podaje się je w nawiasach klamrowych `{ }`, każdy w osobnej parze nawiasów. Liczba oraz kolejność argumentów jest istotna i wynika z definicji instrukcji. Instrukcje mogą mieć także argumenty opcjonalne, podawane w nawiasach kwadratowych `[ ]`. W wypadku większej liczby argumentów opcjonalnych oddziela się je przecinkami. Kolejność argumentów opcjonalnych *nie odgrywa roli*.

Poniższe przykłady ilustrują postać instrukcji `\LaTeX`owych. Ich znaczenie jest tu nieistotne i zostanie opisane później.

Możesz na mnie `\textsl{polegać}`!

Tu wstaw zmianę wiersza.  
`\newline`  
Dziękuję.

Możesz na mnie *polegać*!  
Tu wstaw zmianę wiersza.  
Dziękuję.

#### 1.3.4. Komentarze

Po napotkaniu znaku `%` `\LaTeX` ignoruje resztę bieżącego wiersza (łącznie ze znakiem końca wiersza) oraz znaki odstępu na początku następnego. Znak `%` jest używany do umieszczania komentarzy i dodatkowych informacji w pliku źródłowym.

Mao zmarł `%` sprawdzić!  
`w~1975` roku.

Mao zmarł w 1975 roku.

Znaku `%` używa się niekiedy do dzielenia bardzo długich linii w pliku wejściowym, gdy niedozwolone jest użycie spacji lub złamanie wiersza.

### 1.4. Struktura pliku źródłowego

`\LaTeX` oczekuje, że plik źródłowy posiada określoną strukturę. W szczególności, każdy plik źródłowy składa się z dwóch części: preambuły oraz części głównej. Preambuła powinna się rozpoczynać od instrukcji `\documentclass`:

```
\documentclass{...}
```

Instrukcja ta określa rodzaj tworzonego dokumentu. Po niej można umieścić polecenia dotyczące stylu całego dokumentu oraz dołączyć pakiety poszerzające możliwości `\LaTeX`a. Pakiety dołącza się poleceniem `\usepackage`:

```
\usepackage{...}
```

Część główna dokumentu zaczyna się od instrukcji `\begin{document}`. Za nią znajduje się tekst dokumentu, wzbogacony o L<sup>A</sup>T<sub>E</sub>Xowe polecenia sterujące wyglądem. Na końcu dokumentu musi występować polecenie `\end{document}`. Tekst znajdujący się za tym poleceniem jest przez L<sup>A</sup>T<sub>E</sub>Xa ignorowany.

Rysunek 1.1 pokazuje zawartość minimalnego dokumentu L<sup>A</sup>T<sub>E</sub>Xowego. Użyte w nim instrukcje `\usepackage{...}`, niezbędne do składania w języku polskim, omawiamy w punkcie 2.5.

---

```
\documentclass{article}
\usepackage[MeX]{polski}
% kodowanie: latin2, utf8 lub cp1250
\usepackage[latin2]{inputenc}
\begin{document}
Małe jest piękne.
\end{document}
```

---

Rysunek 1.1: Zawartość minimalnego pliku źródłowego

Rysunek 1.2 przedstawia nieco bardziej rozbudowany plik źródłowy.

## 1.5. Typowa sesja pracy z L<sup>A</sup>T<sub>E</sub>Xem

Na pewno chciałbyś już sprawdzić, jak będzie wyglądał na papierze dokument z przykładu 1.1. Szczegółowy sposób uruchomienia L<sup>A</sup>T<sub>E</sub>Xa zależy od systemu operacyjnego, wersji i upodobań użytkownika<sup>4</sup>. L<sup>A</sup>T<sub>E</sub>X „jako taki” nie jest wyposażony w zintegrowane środowisko graficzne (IDE). Praca z takim systemem polega na wydawaniu odpowiednich poleceń z wiersza poleceń systemu operacyjnego. Oczywiście posługiwanie się L<sup>A</sup>T<sub>E</sub>Xem w ten sposób jest na dłuższą metę niewygodne, niemniej do stworzenia pierwszego dokumentu nic więcej nie potrzeba. Dzięki temu będziesz też wiedział, co kryje się za guzikami, gdy później zainstalujesz jedno z istniejących zintegrowanych środowisk graficznych do pracy z L<sup>A</sup>T<sub>E</sub>Xem.

1. Uruchom edytor i wpisz tekst z przykładu 1.1. Zapisz dokument jako tekstowy plik ASCII. W systemach typu Unix możesz do tego użyć praktycznie dowolnego edytora. W systemie MS Windows musisz zapisać plik jako *Zwykły tekst*, *Dokument tekstowy* lub wybrać *Wszystkie pliki* jako typ dokumentu. W różnych systemach obowiązują różne warianty kodowania polskich znaków. Najpopularniejsze z tych wariantów, to: kodowanie wielobajtowe *Unicode* (MS Windows,

---

<sup>4</sup>O tym, jak zainstalować L<sup>A</sup>T<sub>E</sub>Xa, można przeczytać na <http://www.gust.org.pl>.

---

```

\documentclass[a4paper,11pt]{article}
\usepackage{latexsym}
\usepackage[MeX]{polski}
\usepackage[latin2]{inputenc}% ew. utf8 lub cp1250
% Zdefiniowanie autora i~tytułu:
\author{H.~Szczególny}
\title{Minimalizm}
\frenchspacing
\begin{document}
% Wstawienie autora i~tytułu do składu:
\maketitle
% Wstawienie spisu treści:
\tableofcontents
\section{Kilka spostrzeżeń na wstępie}
Właśnie tu zaczyna się mój cudowny artykuł.
\section{Na pożegnanie}
\ldots{} A~tu się on kończy.
\end{document}

```

---

Rysunek 1.2: Przykład artykułu do czasopisma. Użyte w nim polecenia zostaną objaśnione w dalszej części.

nowe dystrybucje Linuksa), kodowanie CP 1250 (MS Windows) lub ISO 8859-2 (Linux) albo jeszcze coś innego (np. Mac). Dokładna postać wiersza `\usepackage[...]{inputenc}` zależy od wybranego kodowania; wewnątrz nawiasów kwadratowych należy wpisać albo `utf8` jeżeli używamy *Unicode*<sup>5</sup>, albo `latin2` jeżeli korzystamy z ISO 8859-2 lub też `cp1250` jeżeli używamy CP 1250<sup>6</sup>. Więcej informacji na temat kodowania znajdziesz w punkcie 2.5. Nie używaj odstępów w nazwie pliku, przynajmniej na początku swojej przygody z  $\text{\LaTeX}$ em, bo może ci to skomplikować życie. Wybierając nazwę dla pliku, podaj jako jej rozszerzenie `.tex`. Poprawną nazwą byłoby na przykład `foo.tex`.

2. Uruchom program `latex` na pliku utworzonym w punkcie 1, wpisując w wierszu poleceń<sup>7</sup>:

```
latex foo.tex
```

---

<sup>5</sup>Uwaga:  $\text{\LaTeX}$  potrafi przetwarzać dokumenty Unicodowe w ograniczonym zakresie, wystarczającym dla tekstów w językach europejskich, ale dalekim od kompletności.

<sup>6</sup>Wybierz kodowanie ANSI aby zapisać dokument w standardzie CP 1250.

<sup>7</sup>Ponieważ ten sposób pracy jest zwykle nieznanym użytkownikom systemu MS Windows, przypominamy, że wiersz poleceń możemy uruchomić na przykład tak: wywołujemy *Start/Uruchom*, wpisujemy `cmd` lub `command` i naciskamy klawisz Enter.

W wypadku sukcesu zakończy się to utworzeniem pliku o rozszerzeniu `.dvi`<sup>8</sup>. Polecenie trzeba powtórzyć kilka razy, by  $\text{\LaTeX}$  mógł utworzyć spis treści i/lub odsyłacze do pozycji bibliograficznych, rysunków, wzorów matematycznych i śródtytułów. Napotkawszy błąd w dokumencie,  $\text{\LaTeX}$  zakończy jego przetwarzanie i przejdzie do trybu dialogu z użytkownikiem. Wpisanie `Ctrl-D` (lub `Ctrl-Z` w MS Windows) w tym trybie powoduje powrót na poziom wiersza poleceń.

3. Teraz możesz obejrzeć plik DVI. Jest na to kilka sposobów. W systemie typu Unix wyposażonym w środowisko graficzne X Windows możesz wywołać:

```
xdvi foo.dvi &
```

W systemie MS Windows możesz skorzystać z programu `yap` bądź innej przeglądarki plików DVI. Możesz też przetworzyć plik DVI do formatu Postscript:

```
dvips -Pcmz foo.dvi -o foo.ps
```

Pliki w formacie Postscript można oglądać i drukować za pomocą programu `ghostscript`.

Jeśli twoja dystrybucja  $\text{\LaTeX}$ a zawiera program `dvipdf` do konwersji plików DVI na format PDF, to spróbuj go wywołać, wpisując:

```
dvipdf foo.dvi
```

Pliki PDF można oglądać i drukować, korzystając z programu Acrobat lub wspomnianego już `ghostscripta`.

## 1.6. Układ graficzny dokumentu

### 1.6.1. Klasy dokumentów

Na samym początku przetwarzania pliku źródłowego  $\text{\LaTeX}$  musi się dowiedzieć, jakiego typu dokument autor chce uzyskać. Określone jest to w instrukcji `\documentclass`:

`\documentclass[opcje]{klasa}`

gdzie *klasa* oznacza typ dokumentu. W niniejszym wprowadzeniu opisano następujące klasy ze standardowej dystrybucji  $\text{\LaTeX} 2_{\epsilon}$ <sup>9</sup>:

**article** artykuły, krótkie opracowania...

<sup>8</sup>W większości współczesnych dystrybucji systemu  $\text{\TeX}$  dostępne jest polecenie `pdflatex` do generowania dokumentu w formacie PDF. Więcej na ten temat w punkcie 4.7.

<sup>9</sup>W dystrybucji  $\text{\LaTeX} 2_{\epsilon}$  znajdują się także inne, rzadziej wykorzystywane klasy, np. `slides` do przygotowywania przeźroczy. Zamiast klasy standardowej można do tworzenia slajdów zastosować pakiet `foiltex`, dostępny pod adresem [CTAN://macros/latex/packages/supported/foiltex](http://CTAN://macros/latex/packages/supported/foiltex), bądź pakiet `beamer`, spod adresu [CTAN://macros/latex/contrib/beamer](http://CTAN://macros/latex/contrib/beamer) (krótkie wprowadzenie do pakietu `beamer` zawiera punkt 4.8).

`report` dłuższe opracowania, dysertacje magisterskie i doktorskie...  
`book` książki.  
`letter` listy.

Opcje pozwalają zmieniać sposób działania klas. Poszczególne opcje rozdziela się przecinkami. Najczęściej wykorzystywane opcje dla klas standardowych to:

`10pt`, `11pt`, `12pt`    Ustalenie stopnia pisma dla tekstu zasadniczego dokumentu. Domyślną wartością jest 10 punktów.  
`a4paper`, `letterpaper`, ...    Ustalenie wymiarów papieru. Wartością domyślną jest `letterpaper`. Inne dopuszczalne wartości to: `a5paper`, `b5paper`, `executivepaper` i `legalpaper`.  
`fleqn`    Składanie wyeksponowanych wzorów matematycznych od lewego marginesu zamiast domyślnego centrowania.  
`leqno`    Umieszczanie numerów wzorów matematycznych na lewym marginesie zamiast domyślnie na prawym.  
`titlepage`, `notitlepage`    Pierwsza z opcji powoduje, że  $\text{\LaTeX}$  składa tytuł (instrukcja `\maketitle`) oraz streszczenie (instrukcja `\abstract`) na oddzielnej stronie, druga – że skład tekstu zaczyna się na stronie tytułowej. W klasie `article` tytuł i streszczenie nie są domyślnie składane na oddzielnych stronach, podczas gdy w stylach `report` i `book` – są.  
`onecolumn`, `twocolumn`    Skład jedno- lub dwukolumnowy (dwukolumnowy).  
`oneside`, `twoside`    Druk na jednej lub na dwóch stronach kartki papieru. W klasach `article` i `report` domyślną opcją jest `oneside`, natomiast w klasie `book` – `twoside`. Włączenie opcji `oneside` powoduje przy okazji, że  $\text{\LaTeX}$  nie wyrównuje wysokości kolejnych stron, dopuszczając pewną ich zmienność.  
`openright`, `openany`    Wybranie pierwszej opcji powoduje, że tytuły rozdziałów będą umieszczane na stronach nieparzystych. W klasie `article` opcja nie ma znaczenia, gdyż w tej klasie nie jest zdefiniowane pojęcie rozdziału. W klasie `report` domyślną wartością jest `openany`, a w klasie `book` – `openright`.

Przykład. Plik źródłowy może się rozpoczynać od następującej instrukcji:

```
\documentclass[11pt,twoside,a4paper]{article}
```

W tym wypadku dokument zostanie złożony w klasie `article`, pismem w stopniu *11 punktów*, i zostanie przygotowany do wydruku po *dwóch* stronach kartki papieru formatu *A4*.

### 1.6.2. Pakiety

Pakiety rozszerzają możliwości  $\text{\LaTeX}$ a. Sam  $\text{\LaTeX}$  nie ma na przykład instrukcji do dołączania grafiki, kolorowania tekstu, łamania dużych tabel

itp. Do wykonywania tych zadań służą właśnie pakiety. Dołącza się je poleceniem:

```
\usepackage[opcje]{pakiet}
```

gdzie *pakiet* oznacza nazwę pakietu, a *opcje* – listę rozdzielonych przecinkami opcji. Część pakietów znajduje się w podstawowej dystrybucji L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> (zobacz tabela 1.1), kolejnych kilkanaście, m.in. tak użyteczne jak **babel** czy **graphicx**, jest do niej *obowiązkowo* dodawanych; pozostałe są rozpowszechniane oddzielnie. Gdy używamy L<sup>A</sup>T<sub>E</sub>Xa w systemie, którym zarządza (dobry) administrator, to informacja o dostępnych pakietach powinna się znajdować w [16]. Podstawowym źródłem informacji o L<sup>A</sup>T<sub>E</sub>Xu jest [6]. Zawiera on opis setek pakietów, a także informuje, jak można pisać własne rozszerzenia L<sup>A</sup>T<sub>E</sub>Xa.

Wartościowym źródłem informacji o istniejących pakietach L<sup>A</sup>T<sub>E</sub>Xa jest *T<sub>E</sub>X Catalogue Online* Grahama Williamsa [27], dostępny niestety tylko w języku angielskim. Użytkownikom polskojęzycznym polecamy *Wirtualną Akademię* Włodzimierza Macewicza [18].

### 1.6.3. Style strony

Typowa strona składa się z trzech podstawowych części. Powyżej kolumny tekstu głównego znajduje się *pagina górna* (główka), która może zawierać numer strony, tytuł rozdziału czy punktu. Poniżej kolumny tekstu znajduje się *pagina dolna* (stopka). W niniejszym wprowadzeniu pagina dolna jest pusta, górna natomiast zawiera numer strony oraz tytuł rozdziału na stronach parzystych, a tytuł punktu – na nieparzystych<sup>10</sup>.

L<sup>A</sup>T<sub>E</sub>X pozwala wybrać jeden z trzech sposobów składania pagin. Służy do tego instrukcja:

```
\pagestyle{styl}
```

Dopuszczalne wartości argumentu *styl* są następujące:

**plain** pagina górna jest pusta, a pagina dolna zawiera wycentrowany numer strony. Ten styl jest domyślny;

**headings** pagina górna zawiera numer strony oraz tytuł, pagina dolna jest pusta;

**empty** pagina górna i dolna są puste.

Możliwa jest także zmiana stylu *bieżącej* strony. Służy do tego polecenie:

```
\thispagestyle{styl}
```

---

<sup>10</sup>Paginę zawierającą oprócz kolejnego numeru kolumny (strony) także informację dotyczącą treści tej kolumny drukarze nazywają *żywą paginą*.

Tabela 1.1: Wybrane pakiety z podstawowej dystrybucji L<sup>A</sup>T<sub>E</sub>Xa

---

<code>doc</code>	Służy do drukowania dokumentacji pakietów oraz innych części składowych L <sup>A</sup> T <sub>E</sub> Xa. Opis znajduje się w pliku <code>doc.dtx</code> <sup>a</sup> .
<code>exscale</code>	Umożliwia skalowanie fontów matematycznych, tak by optycznie były zgodne z otaczającym tekstem, np. w tytułach rozdziałów. Opis w <code>ltxscale.dtx</code> .
<code>fontenc</code>	Definiuje układ znaków, którego ma używać L <sup>A</sup> T <sub>E</sub> X. Opis w punkcie 2.5 i w <code>ltoutenc.dtx</code> .
<code>ifthen</code>	Umożliwia korzystanie z poleceń typu <code>if... then do... otherwise do</code> . Opis w <code>ifthen.dtx</code> i [6].
<code>latexsym</code>	Udostępnia specjalny font symboliczny (fonty <i>lasy</i> ). Opis w <code>latexsym.dtx</code> i [6].
<code>makeidx</code>	Udostępnia polecenia do przygotowywania skorowidzów. Opis w punkcie 4.3 i [6].
<code>syntonly</code>	Powoduje, że dokument jest przetwarzany bez składania czegokolwiek. Przydatny do szybkiego sprawdzenia, czy dokument nie zawiera błędów. Opis w <code>syntonly.dtx</code> i [6].
<code>inputenc</code>	Definiuje układ znaków w pliku źródłowym, jak: ASCII, ISO Latin-1, ISO Latin-2, 437/850 IBM, Apple Macintosh, Next, ANSI-Windows, itd. Opis w <code>inputenc.dtx</code> .

---

<sup>a</sup>Plik ten powinien być zainstalowany w twoim systemie. Aby otrzymać z niego plik DVI, wystarczy w katalogu z prawem do zapisu napisać `latex doc.dtx`. To samo stosuje się do innych pakietów z tej tabeli.

Punkt 4.4 na stronie 71 niniejszego *Wprowadzenia* oraz podręcznik [6] w punkcie 4.4 na stronie 54 zawierają więcej informacji o paginach, w szczególności o sposobach samodzielnego definiowania ich wyglądu.

## 1.7. Nazwy plików związanych z L<sup>A</sup>T<sub>E</sub>Xem

Pracując z L<sup>A</sup>T<sub>E</sub>Xem, szybko zauważysz, że na dysku pojawia się mnóstwo plików o różnych rozszerzeniach nazwy, a ty nie wiesz, co to za jedno. W poniższym wykazie objaśniono rozmaite typy plików, z którymi możesz się zetknąć. Wykaz ten nie pretenduje do kompletnego, dlatego napisz do nas, gdy napotkasz jakieś nowe rozszerzenie, które uznasz za warte opisania.

- `.tex` Plik źródłowy z dokumentem w notacji L<sup>A</sup>T<sub>E</sub>Xa bądź zwykłego T<sub>E</sub>Xa. Można go kompilować programem `latex` bądź, odpowiednio, `tex`.
- `.sty` Pakiet makr L<sup>A</sup>T<sub>E</sub>Xowych. Plik tego typu można dołączać do dokumentu L<sup>A</sup>T<sub>E</sub>Xowego, używając do tego celu instrukcji `\usepackage`.



- .dtx** Udokumentowany T<sub>E</sub>X. Jest to podstawowy format, w jakim dystrybuowane są style L<sup>A</sup>T<sub>E</sub>Xa. Skutkiem kompilacji pliku tego typu jest broszurka z udokumentowanymi makrami.
- .ins** Instalator dla plików **.dtx**. Ściągając z sieci pakiet L<sup>A</sup>T<sub>E</sub>Xowy, otrzymasz na ogół pliki **.dtx** i **.ins**. Uruchomienie L<sup>A</sup>T<sub>E</sub>Xa na pliku **.ins** powoduje rozpakowanie pliku **.dtx**.
- .cls** Plik z klasą L<sup>A</sup>T<sub>E</sub>Xa definiującą wygląd składanych w L<sup>A</sup>T<sub>E</sub>Xu dokumentów. Właśnie do tych plików odnosi się występująca na początku dokumentu instrukcja `\documentclass`.
- .fd** Definicja niektórych właściwości fontów L<sup>A</sup>T<sub>E</sub>Xa.

W wyniku kompilacji dokumentu powstają następujące pliki:

- .dvi** *Device Independent File* (plik niezależny od urządzenia), będący wynikiem kompilacji pliku źródłowego przez „tradycyjnego” L<sup>A</sup>T<sub>E</sub>Xa<sup>11</sup>. Zawartość plików DVI możesz zobaczyć w przeglądarce plików DVI albo posłać na drukarkę, korzystając z programu dvips albo innego programu o podobnych funkcjach.
- .pdf** Portable Document Format (przenośny format dokumentów). Zagadnienie generowania plików w formacie PDF jest omawiane w punkcie 4.7, zaczynającym się na stronie 74.
- .log** Zawiera szczegółowy raport z tego, co się wydarzyło podczas kompilacji: które pliki były przetwarzane, co szczególnego i ewentualnie jakie błędy L<sup>A</sup>T<sub>E</sub>X w nich znalazł, a także – jakie pliki powstały w wyniku kompilacji.
- .toc** Zawiera nagłówki rozdziałów i punktów dokumentu. Jest czytany przez L<sup>A</sup>T<sub>E</sub>Xa w następnym przebiegu kompilacji, w celu wygenerowania spisu treści.
- .lof** Podobny do pliku **.toc**, z tym że zawiera wykaz ilustracji.
- .lot** Tak samo, lecz dotyczy wykazu tabel.
- .aux** Inny plik pomocniczy, przenoszący informację z jednego przebiegu kompilacji do następnego. Jest używany między innymi do magazynowania informacji związanej z odsyłaczami występującymi w dokumencie.
- .idx** Jeśli dokument zawiera skorowidz, to w tym pliku L<sup>A</sup>T<sub>E</sub>X zapisze wszystkie jego hasła. Do przetworzenia tego pliku służy program makeindex (lub plmindex, w przypadku języka polskiego). Więcej o tworzeniu skorowidzów przeczytasz w punkcie 4.3 na stronie 69.
- .ind** Przetworzony plik **.idx**, gotowy do włączenia do dokumentu w następnym cyklu kompilacji.
- .ilg** Sprawozdanie z tego, co zrobił program makeindex.

---

<sup>11</sup>W nowych wersjach systemu, L<sup>A</sup>T<sub>E</sub>X potrafi także generować dokumenty bezpośrednio w formacie PDF. Więcej informacji na ten temat zawiera punkt 4.7.

## 1.8. Duże dokumenty

Pracując nad dużym dokumentem, wygodnie jest podzielić plik źródłowy na mniejsze części. W  $\text{\LaTeX}$ u mamy dwie instrukcje ułatwiające pracę z tak podzielonymi dokumentami. Pierwszą z nich jest:

`\include{plik}`

Włącza ona do dokumentu zawartość innego pliku. Przed przetworzeniem, a także po przetworzeniu zawartości włączanego pliku  $\text{\LaTeX}$  rozpoczyna nową stronę.

Drugiej instrukcji używa się w preambule. Pozwala ona wstawiać do tekstu jedynie wybrane pliki.

`\includeonly{plik1,plik2,...}`

Spśród instrukcji `\include` zostaną wykonane tylko te, które dotyczą plików wymienionych w argumencie `\includeonly`. Uwaga: w wykazie plików nie wolno używać odstępów; poszczególne pliki należy oddzielać wyłącznie przecinkiem!

Polecenie `\include` rozpoczyna skład dołączanego tekstu od nowej strony. W połączeniu z poleceniem `\includeonly` w preambule instrukcja `\include` umożliwia przetwarzanie wybranych plików bez zmiany miejsc łamania poszczególnych stron i z zachowaniem prawidłowej numeracji stron, rozdziałów, tabel itp. Czasami jednak rozpoczynanie składu od nowej strony nie jest pożądane. W takiej sytuacji zamiast `\include` należy posłużyć się instrukcją:

`\input{plik}`

Wstawia ona zawartość podanego pliku już bez żadnych dodatkowych efektów.

Argument poleceń `\input` i `\include` może zawierać pełną ścieżkę do pliku, ale w imię wygody i przenośności nie należy używać ścieżek bezwzględnych. Na przykład:

```
\include{C:\Documents and Settings\elka\Moje dokumenty\r2.tex}
```

jest koszmarnym błędem w większości dystrybucji  $\text{\LaTeX}$ a. Nawet jeżeli nasz  $\text{\LaTeX}$  wie, co oznacza specyficzny dla MS Windows zapis `C:\`, oraz poradzi sobie z odstępami w nazwach katalogów i ze znakami `\` (które w tym wypadku nie są początkiem żadnego polecenia), to taki dokument przesłany komuś, kto będzie go kompilował w systemie Linux, sprawi mu mnóstwo problemów. Pamiętajmy: bez względu na to, jakiego systemu operacyjnego używamy, katalogi powinniśmy oddzielać znakiem `/`.

## Rozdział 2

# Składanie tekstu

*Po lekturze poprzedniego rozdziału znasz już podstawowe pojęcia związane z systemem  $\text{\LaTeX}$  2 $\epsilon$ . W tym rozdziale Twoja wiedza wzbogaci się o informacje niezbędne do tworzenia prawdziwych dokumentów.*

### 2.1. Struktura tekstu i języka

Hanspeter Schmid <hanspi@schmid-werren.ch>

Głównym zadaniem słowa pisanego jest przekaz myśli, informacji lub wiedzy. Nadanie zapisowi odpowiedniej struktury pomaga czytelnikowi lepiej zrozumieć przekazywane treści. Skład typograficzny może czytelnikowi tę logiczną i semantyczną strukturę tekstu przybliżyć.

$\text{\LaTeX}$  tym się różni od innych systemów składu, że do złożenia tekstu wystarcza mu znajomość logicznej i semantycznej struktury tekstu. Postać typograficzna jest wyprowadzana na podstawie „reguł” zawartych w klasie dokumentu i plikach z makroinstrukcjami.

Najważniejszą jednostką podziału tekstu w  $\text{\LaTeX}$ u (a także w typografii) jest akapit. Jest to „jednostka podziału” głównie dlatego, że według wszelkich kanonów sztuki typograficznej treść akapitu powinna być poświęcona jednej spójnej myśli lub pojęciu. Tak więc, gdy zaczyna się nowa myśl, powinien się zacząć nowy akapit. Kontynuacja dotychczasowej myśli w nowym akapicie jest błędem. Niezgodne z regułami sztuki jest też pojawienie się w tym samym akapicie całkowicie nowej myśli. W następnym punkcie omówimy instrukcje nakazujące  $\text{\LaTeX}$ owi złamanie linii bez rozpoczynania nowego akapitu, na przykład poleceniem `\`, a także sposób rozpoczęcia nowego akapitu, na przykład przez umieszczenie w kodzie źródłowym pustej linii.

Większość ludzi całkowicie lekceważy znaczenie właściwej organizacji akapitów. Co więcej, wiele osób nawet nie zdaje sobie sprawy, czym akapit naprawdę jest, i (szczególnie w  $\text{\LaTeX}$ u) kończy akapit, nawet o tym nie wiedząc. Błąd taki łatwo popełnić zwłaszcza w tekście z równaniami.

Zobaczmy, dlaczego w poniższych przykładach w jednej z takich sytuacji należy przejść do nowego akapitu, a w innej – nie. Czytelnik, który nie zna jeszcze wszystkich poleceń użytych w tych przykładach, powinien dokładnie przeczytać ten i następny rozdział, a następnie wrócić do tego punktu i przeczytać go jeszcze raz.

```
% Przykład 1
\ldots Słynne równanie Einsteina
\begin{equation}
e = m \cdot c^2 \; ; \; ,
\end{equation}
jest najbardziej znanym, ale też
najmniej rozumianym równaniem w fizyce.
```

```
% Przykład 2
\ldots którego wynikiem jest prądowe prawo Kirchhoffa:
\begin{equation}
\sum_{k=1}^n I_k = 0 \; ; \; .
\end{equation}
```

Napięciowe prawo Kirchhoffa ma zaś swój początek w\ldots

```
% Przykład 3
\ldots co ma określone zalety.
```

```
\begin{equation}
I_D = I_F - I_R
\end{equation}
jest rdzeniem innego modelu tranzystora. \ldots
```

Mniejszą od akapitu jednostką podziału tekstu jest zdanie. W tekstach angielskojęzycznych odstęp po kropce kończącej zdanie jest większy od odstępu po kropce oznaczającej skrót. Zależnie od kontekstu  $\LaTeX$  stara się użyć krótszego lub dłuższego odstępu. W razie pomyłek z jego strony powinniśmy mu wskazać nasze intencje. Jak to zrobić, wyjaśniamy w dalszej części tego rozdziału.

Właściwa organizacja tekstu dotyczy nawet fragmentów zdań. Wiele języków ma bardzo skomplikowaną interpunkcję, ale w większości wypadków (wliczając angielski i niemiecki) stawiając przecinek w określonym miejscu w zdaniu, nie popełnimy na ogół błędu, pamiętając o zasadzie, że przecinek oznacza krótką przerwę w wypowiedzi. Dlatego – jeśli nie jesteśmy pewni, gdzie w zdaniu należy go postawić – przeczytajmy zdanie na głos i postavmy przecinki wszędzie tam, gdzie zrobiliśmy krótką przerwę na wdech. Ale nie sugerujemy się wyłącznie tym! Jeśli w danym miejscu przecinek nie wygląda najlepiej, to go usuńmy; jeśli w innym miejscu odczuwamy potrzebę

wzięcia powietrza do płuc (lub zrobienia krótkiej przerwy), to postawmy tam przecinek. A najlepiej w celu rozwiania wątpliwości sięgnąć do słownika ortograficzno-interpunkcyjnego, który zresztą zawsze warto mieć pod ręką.

I na koniec: akapit nie jest największą logiczną jednostką podziału tekstu. Można jeszcze wspomnieć o rozdziałach, punktach, podpunktach itd. Jednakże od strony typograficznej już same nazwy poleceń w rodzaju `\section{Struktura tekstu i języka}` są na tyle oczywiste (dla znających język angielski), że sposób ich użycia jest łatwo zrozumiały.

## 2.2. Składanie akapitów i łamanie stron

### 2.2.1. Składanie akapitów

Książki najczęściej składa się tak, że wszystkie wiersze w akapitach są tej samej długości. Dążąc do optymalnej prezentacji akapitu,  $\text{\LaTeX}$  ustala miejsca złamań linii oraz wielkość odstępów między słowami. W razie potrzeby przenosi wyrazy, których nie jest w stanie zmieścić w wierszu. Sposób składania akapitów zależy od użytej klasy dokumentu. Najczęściej pierwszy wiersz akapitu jest wcięty, a między akapitami nie ma dodatkowych odstępów. Więcej na ten temat można przeczytać w punkcie 6.3.2.

Zgodnie z anglo-amerykańskimi zwyczajami typograficznymi  $\text{\LaTeX}$  nie wstawia wcięcia akapitowego bezpośrednio po tytułach rozdziałów, punktów itd. Polskie zwyczaje nakazują jednak rozpoczynanie także początkowych akapitów wcięciem. Efekt ten można osiągnąć przez dołączenie w preambule pakietu `indentfirst`. W niniejszym tłumaczeniu nie stosujemy wcięć w początkowych akapitach, gdyż akurat pod tym względem bardziej nam się podobają zwyczaje anglosaskie.

Czasami przydaje się instrukcja nakazująca  $\text{\LaTeX}$ owi złamanie linii. Polecenie:

`\l lub \newline`

rozpoczyna nową linię bez rozpoczynania nowego akapitu.

Natomiast instrukcja:

`\l*`

zakazuje dodatkowo złamania strony (w miejscu złamania linii). Z kolei instrukcja:

`\newpage`

rozpoczyna nową stronę.

Instrukcje:

`\linebreak[n], \nolinebreak[n], \pagebreak[n]` oraz `\nopagebreak[n]`

oznaczają, odpowiednio: zachętę do złamania wiersza, niezgodę na złamanie wiersza, zachętę do złamania strony i niezgodę na złamanie strony (w miejscu ich wystąpienia w dokumencie). Opcjonalny argument  $n$ , o dopuszczalnej wartości od 0 do 4, określa stopień tej zachęty (niezgody). Domyślna wartość 4 to bezwarunkowy zakaz lub nakaz złamania linii/strony. Wartość mniejsza od 4 pozostawia L<sup>A</sup>T<sub>E</sub>X-owi swobodę zignorowania instrukcji, jeżeli skład otrzymany w jej rezultacie byłby kiepskiej jakości.

Poleceń z grupy **break** nie należy mylić z tymi z grupy **new**. Mimo otrzymania polecenia typu **break** L<sup>A</sup>T<sub>E</sub>X stara się wyrównać wiersz do prawego marginesu czy też wypełnić stronę do całej jej wysokości. Nietrudno zgadnąć, jakiej instrukcji należy użyć, gdy naprawdę zależy nam na rozpoczęciu nowego wiersza<sup>1</sup>.

Jeżeli w wierszu zakończonym instrukcją `\newline` jest zbyt mało tekstu, to L<sup>A</sup>T<sub>E</sub>X nie wyrówna tego wiersza do prawego marginesu, lecz wstawi po tekście odpowiedni odstęp. Jeżeli zamiast `\newline` użyjemy `\linebreak`, to L<sup>A</sup>T<sub>E</sub>X postara się wyrównać zawartość kończącej linijki do prawego marginesu. Zbyt małe wypełnienie wiersza tekstem zmusi L<sup>A</sup>T<sub>E</sub>Xa do umieszczenia między wyrazami zbyt dużych odstępów. Efekt ten L<sup>A</sup>T<sub>E</sub>X sygnalizuje podczas przetwarzania dokumentu, wyświetlając komunikat:

```
Underfull \hbox (badness 10000) in paragraph at lines 4--5
```

Liczba po słowie **badness** w tym komunikacie wskazuje, jak bardzo L<sup>A</sup>T<sub>E</sub>Xowi „nie podoba się” złożony wiersz. Tutaj jest to maksymalna negatywna ocena 10000. Liczby na końcu komunikatu to numery pierwszej i ostatniej linii akapitu, w którym L<sup>A</sup>T<sub>E</sub>X musiał złożyć wiersz z nadmiernymi odstępami między wyrazami. Chociaż wielu użytkowników L<sup>A</sup>T<sub>E</sub>Xa nie zwraca uwagi na podobne ostrzeżenia, warto sobie zdawać sprawę, co one oznaczają.

Oprócz wyżej wymienionych istnieją jeszcze w L<sup>A</sup>T<sub>E</sub>Xu instrukcje:

`\clearpage, \cleardoublepage`

Obydwie rozpoczynają skład nowej strony. Instrukcja `\cleardoublepage` działa tak jak `\clearpage`, z tym że rozpoczynana strona ma numer nieparzysty; w razie potrzeby tworzona jest strona pusta (wakatowa)<sup>2</sup>. W trybie składu dwukolumnowego (opcja `twocolumn`) instrukcja `\newpage` kończy łam, natomiast `\clearpage` oraz `\cleardoublepage` kończą stronę, pozostawiając w razie potrzeby pusty prawy łam.

Jeżeli na stronie zakończonej instrukcją `\newpage` albo `\clearpage` jest zbyt mało tekstu, to L<sup>A</sup>T<sub>E</sub>X wstawia odpowiedni odstęp u dołu strony, wypełniający pozostałą część kolumny. W wypadku polecenia `\pagebreak` L<sup>A</sup>T<sub>E</sub>X wyrównuje zawartość kolumny do dolnego brzegu, wstawiając odstępy

<sup>1</sup>Zagadka ta jest łatwa dla znających język angielski – jest to polecenie `\newline`.

<sup>2</sup>Mówiąc precyzyjnie zależy to od stanu opcji `openright/openany`, w szczególności w klasie „article” `\cleardoublepage` domyślnie działa tak jak `\clearpage`.

po między akapitami lub innymi elementami na stronie. Jeżeli wstawione odstępy okazują się za duże, co zdarza się dość często, to podczas przetwarzania dokumentu generowane jest odpowiednie ostrzeżenie, na przykład:

```
Underfull \vbox (badness 10000) has occurred
while \output is active [7]
```

W powyższym komunikacie (z uwagi na wąskość szpalty przełamanym na dwie linijki) liczba po słowie **badness** wskazuje, jak bardzo  $\text{\LaTeX}$ owi nie podobą się złożona strona. Tutaj jest to 10000 – maksymalna w  $\text{\TeX}$ u ujemna ocena jakości składu. Liczba w nawiasach prostokątnych na końcu komunikatu oznacza numer strony, podczas składania której wystąpił problem.

$\text{\LaTeX}$  zawsze stara się znaleźć najlepszy podział akapitu na wiersze. Kiedy nie potrafi znaleźć podziału, który spełnia jego wysokie wymagania jakościowe, wtedy niektóre wyrazy wystają na prawy margines. Sytuacja taka jest sygnalizowana komunikatem podobnym do następującego:

```
Overfull \hbox (5.5452pt too wide) in paragraph at lines 79--83
```

W komunikacie tym liczba w nawiasie okrągłym oznacza, że pewien fragment tekstu wystaje o 5,5452 punktów drukarskich na prawy margines. Problem wystąpił w akapicie, który w pliku źródłowym jest w wierszach od 79 do 83. Podobne ostrzeżenia pojawiają się najczęściej wówczas, gdy  $\text{\LaTeX}$  nie potrafi przenieść wyrazów w akapicie zgodnie z zadanymi wzorcami przenoszenia tak, by nie popsuć jakości składu. Komunikat typu *overfull hbox* nie wystarcza na ogół do dokładnego ustalenia przyczyny jego wystąpienia. Można wtedy jako argumentu polecenia `\documentclass` użyć opcji **draft**, na skutek czego  $\text{\LaTeX}$  oznaczy wystające wiersze małą czarną sztabką na prawym marginesie szpalty.

Deklaracja `\sloppy` nakazuje  $\text{\LaTeX}$ owi nieco obniżyć jego domyślnie wysokie standardy. Zapobiega to – w większości wypadków – występowaniu zbyt długich linijek, kosztem jednak zwiększenia odstępów międzywyrazowych, czyli pogorszenia jakości składu. Mogą się pojawiać ostrzeżenia typu *underfull hbox*, co w większości wypadków (zwłaszcza gdy podana miara kiepskości jest powyżej 5000) oznacza, że skład nie jest zbyt udany. Instrukcja `\fussy` działa w odwrotnym kierunku, to znaczy przywraca domyślne, wysokie standardy  $\text{\LaTeX}$ a.

### 2.2.2. Przenoszenie wyrazów

W razie potrzeby  $\text{\LaTeX}$  przenosi (dzieli) wyrazy. Jeżeli algorytm podziału przeniósł jakiś wyraz błędnie, to właściwe miejsca przenoszenia można zadać instrukcją:

```
\hyphenation{słowo1 słowo2 słowo3...}
```

Słowa z listy argumentów można dzielić wyłącznie w miejscach oznaczonych znakiem `-`. Instrukcji tej wolno użyć jedynie w preambule dokumentu,

a wyrazy-argumenty mogą zawierać (oprócz znaku -) wyłącznie litery. Nie ma natomiast znaczenia, czy w tych wyrazach używa się liter wielkich czy małych. Instrukcja `\hyphenation` z przykładu poniżej pozwala podzielić słowo „ćwierć-li-trówka” jedynie w dwóch zaznaczonych miejscach i w ogóle zabrania dzielić słowa „szczypce”. Wyrazy z listy argumentów nie mogą zawierać żadnych znaków specjalnych ani symboli. Przykład:

```
\hyphenation{ćwierć-li-trówka szczypce}
```

W językach fleksyjnych, do jakich należy polski, instrukcja `\hyphenation` jest dużo mniej przydatna niż w angielskim. Aby dany wyraz zawsze był dobrze przenoszony, należałoby wypisać wszystkie jego formy. Opracowane przez Hannę Kołodziejską, Bogusława L. Jackowskiego i Marka Ryćko wzorce przenoszenia wyrazów dla języka polskiego sprawdzają się w tak znacznej większości wypadków, że praktycznie nie ma potrzeby stosowania tej konkretnej instrukcji. Co więcej, próba użycia polecenia `\hyphenation` w wypadku stosowania również pakietu `inputenc` (por. punkt 2.5) zakończy się błędem w czasie przetwarzania dokumentu.

Instrukcja `\-` wskazuje, w których miejscach *wolno*  $\text{\LaTeX}$ owi przenieść wyraz do nowego wiersza;  $\text{\LaTeX}$  nie podzieli tego wyrazu w żadnym oprócz wskazanych miejsc. Instrukcja odnosi się do konkretnego wystąpienia słowa w dokumencie i nie wpływa na miejsca podziału w innych jego wystąpieniach. Przydaje się ona szczególnie w wypadku wyrazów ze znakami specjalnymi, na przykład akcentowanymi, gdyż automatycznie  $\text{\LaTeX}$  dzieli jedynie wyrazy złożone z samych liter.

```
Nie\~bie\~sko\~bia\~ło\~zie\~ło\~%  
no\~nie\~bie\~ski
```

Niebieskobiałozielononiebieski

Tekst będący argumentem polecenia:

`\mbox{tekst}`

nigdy nie zostanie przeniesiony.

```
Numer mojego telefonu wkrótce  
się zmieni na \mbox{0116 291 2319}.
```

```
Parametr \mbox{\emph{nazwa}} to  
nazwa pliku.
```

Numer mojego telefonu wkrótce się zmieni na  
0116 291 2319.  
Parametr *nazwa* to nazwa pliku.

Polecenie `\fbox` jest podobne do `\mbox`, z tym że dodatkowo dookoła argumentu rysuje ramkę (por. punkt 6.6).

### 2.3. Kilka gotowych oznaczeń napisów

W przykładach na poprzednich stronach pojawiło się kilka prostych instrukcji  $\text{\LaTeX}$ a do składania krótkich napisów.



Instrukcja	Przykład	Opis
<code>\today</code>	15 stycznia 2007	Bieżąca data
<code>\TeX</code>	$\TeX$	Twój ulubiony system składu
<code>\LaTeX</code>	$\LaTeX$	Nazwa tej gry
<code>\LaTeXe</code>	$\LaTeX 2_{\epsilon}$	Obecne jej wcielenie

## 2.4. Znaki specjalne i symbole

### 2.4.1. Cudzysłowy

Znaku cudzysłowu " używa się inaczej niż na maszynie do pisania. W publikacjach drukowanych różnie oznacza się początek i koniec cudzysłowu. Występują także różnice w sposobach oznaczania cudzysłowów w różnych językach. Dwa apostrofy ‘ otwierają, a dwa apostrofy ’ zamykają  $\LaTeX$ owy cudzysłów według reguł języka angielskiego:

‘‘Please press the ‘x’ key.’’

“Please press the ‘x’ key.”

W języku polskim cudzysłowy otwierający oznaczają się dwoma przecinkami ,, , natomiast zamykający – dwoma apostrofami ’ ’<sup>3</sup>. Gdy zachodzi konieczność użycia cudzysłowu w tekście już objętym cudzysłowem, to stosuje się „cudzysłowy «francuskie»”, oznaczane w pliku źródłowym znakami, odpowiednio, mniejszości << i większości >>.

„Przechodź tylko po <<zebrach>>’’!

„Przechodź tylko po «zebrach»”!

### 2.4.2. Pauzy i myślniki

Zwyczajne maszyny do pisania posiadają tylko jeden znak w kształcie poziomej kreski „-”. W składzie drukarskim występują aż cztery rodzaje kresek poziomych. Są to: łącznik, myślnik, półpauza i minus, używany we wzorach matematycznych.

Łącznik (dywiz) jest najkrótszą z kresek. Stosuje się go do dzielenia i przenoszenia wyrazów oraz do łączenia wyrazów wieloczłonowych (np. „niebiesko-czarny”). Zgodnie z polskimi regułami wyraz wieloczłonowy można podzielić i przenieść albo w obrębie wyrazów składowych, albo na łączniku. W drugim z tych przypadków łącznik należy powtórzyć, to znaczy powinien się on znaleźć zarówno na końcu pierwszego wiersza, jak też na początku drugiego. Oto możliwe miejsca podziału wyrazu niebiesko-czarny:

<sup>3</sup>Porównaj uwagi o tym sposobie oznaczania cudzysłowów z punktu 2.5 w części dotyczącej fontów i ich kodowania (str. 27). Dotyczy to również cudzysłowów francuskich.

nie- biesko-czarny	niebie- sko-czarny	niebiesko- -czarny	niebiesko-czar- ny
-----------------------	-----------------------	-----------------------	-----------------------

Standardowy L<sup>A</sup>T<sub>E</sub>X nie zna polskich norm i dlatego dzieli wyrazy wieloczłonowe w miejscu połączenia, bez powielania łącznika. Jeżeli do składu w języku polskim korzystamy z zestawu platex, to w pliku źródłowym w miejsce łącznika w wyrazach wieloczłonowych powinniśmy zastosować instrukcję `\dywiz` (np. `niebiesko{\dywiz}czarny`).

W pewnych sytuacjach lepiej nie dzielić wyrazów połączonych łącznikiem. Jeżeli na przykład mówimy o wydziale K-2, kodzie pocztowym czy numerze telefonu, to w takich wypadkach łącznik zapisujemy w pliku źródłowym jako pojedynczą kreskę -.

Znaku łącznika używa się również do przenoszenia wyrazów. Jednak w L<sup>A</sup>T<sub>E</sub>Xu odbywa się to automatycznie i nie wymaga ręcznych ingerencji użytkownika.

Półpauza to kreska o połowę krótsza od myślnika. Stosuje się ją przede wszystkim w zapisie zakresów liczbowych, np. „str. 11–13”, czy „w latach 1960–1963”. Przed i po półpauzie nie dodaje się odstępów. Odstępy takie muszą się pojawić w sytuacjach takich jak: „11 października – 13 listopada”. Półpauzę zapisuje się za pomocą dwóch następujących po sobie minusów --.

Myślnik „—” to dłuższa kreska, używana jako znak przestankowy. Zapisujemy go za pomocą trzech następujących po sobie znaków -, czyli ---. W języku polskim należy przed i po myślniku umieścić odstęp, inaczej niż w krajach anglosaskich, gdzie nie otacza się myślnika odstępami. Wiele osób uważa konstrukcję złożoną z odstępów, myślnika i kolejnego odstępów za zbyt wybijającą się w składzie. Z tychże estetycznych powodów często w roli myślnika używa się „dwukreskowej” półpauzy, i takie właśnie podejście zastosowano w niniejszym tłumaczeniu.

We wzorach matematycznych, czyli wewnątrz trybu matematycznego, znak minusa uzyskujemy, pisząc zwyczajnie -. Przykładowo, zapis  $\$-2\$$  daje w składzie -2, podczas gdy  $-2$  daje -2.

### 2.4.3. Odstępy nieładliwe

Polskie zasady typograficzne nie pozwalają łamać akapitów z pozostawianiem na końcu wierszy jednoliterowych spójników bądź przyimków. Przykładowo, w zdaniu „Jan Kochanowski urodził się w Czernolesie” nieładnie na końcu wiersza wyglądałby przyimek „w”.

Odstępy, na których nie wolno złamać wiersza, zaznacza się w pliku źródłowym znakiem tyldy „~” zamiast zwykłym odstępem. Na przykład, aby w powyższej sytuacji zapobiec przeniesieniu składu do nowego wiersza, powinniśmy zapisać w pliku źródłowym: `w~Czarnolesie`.

Jest wiele sytuacji, w których związek fragmentów zdania jest tak silny, że wewnątrz nich nie należy łamać na wiersze. Nie zawsze decyzja jest tak prosta jak w wypadku wspomnianych spójników. Oto garść przykładów:

godz.~17.00; od~15 do 40~osób; na str.~2 napisano; rozdz.~2;  
2~rozdziały; p.~Jan Nowak; p.~J.~Nowak; I~część IX~Symfonii.

Ze względu na zależność od kontekstu obowiązek decydowania o użyciu tyldy spada na użytkownika L<sup>A</sup>T<sub>E</sub>Xa.

#### 2.4.4. Tylda (~)

W adresach internetowych często występuje znak tyldy. W L<sup>A</sup>T<sub>E</sub>Xu można by do jego uzyskania użyć instrukcji `\~`, ale wynik: `\~` nie jest chyba tym, czego oczekujemy. Lepiej zrobić tak:

```
http://www.rich.edu/~bush \\
http://www.clever.edu/$\sim$demo
```

```
http://www.rich.edu/~bush
http://www.clever.edu/~demo
```

A jeszcze lepiej dołączyć do preambuły dokumentu pakiet `url` i korzystać z polecenia `\url{...}`.

#### 2.4.5. Oznaczenie stopni (°)

Poniższy przykład ilustruje, jak w L<sup>A</sup>T<sub>E</sub>Xu uzyskuje się symbol stopni:

Jest  $-30^{\circ}$ ,  $\mathrm{C}$ .  
Niedługo zacznę nadprzewodzić.

Jest  $-30^{\circ}\mathrm{C}$ . Niedługo zacznę nadprzewodzić.

Pakiet `textcomp` udostępnia symbol stopni także jako `\textcelsius`.

#### 2.4.6. Symbol waluty euro (€)

Pisząc dziś o pieniądzach, nie można się obejść bez symbolu euro. Znak ten występuje w wielu współczesnych fontach. Po załadowaniu pakietu `textcomp` w preambule:

```
\usepackage{textcomp}
```

można do uzyskania symbolu euro użyć oznaczenia:

```
\texteuro
```

Jeśli używany font nie zawiera własnego symbolu euro albo nam się on nie podoba, to mamy dwie dodatkowe możliwości: Pierwszą jest pakiet `eurosym`. Udostępnia on oficjalny znak euro:

```
\usepackage[official]{eurosym}
```

Jeśli wolimy znak euro zgodny optycznie z fontem, to zastąpmy opcję `official` opcją `gen`.

Pakiet `marvosym` dostarcza wielu różnych symboli, w tym euro pod nazwą `\EURtm`. Jego wadą jest to, że nie udostępnia wersji pochylonej i wytłuszczonej euro.

Tabela 2.1: Torba pełna symboli euro

LM+textcomp	<code>\texteuro</code>	€	€	€
eurosym	<code>\euro</code>	€	€	€
[gen]eurosym	<code>\euro</code>	€	€	€
marvosym	<code>\EURtm</code>	€	€	€

#### 2.4.7. Wielokropek (...)

W typowym piśmie maszynowym przecinek oraz kropka zajmują tyle samo miejsca co każdy inny znak. W piśmie drukarskim szerokość tych znaków jest z reguły bardzo mała i dlatego, jeżeli umieścimy je obok siebie, to odstęp między nimi będą zbyt małe. Do uzyskiwania wielokropka (trzech kropek) używamy instrukcji `\ldots`. Przykład:

```
\ldots
```

Nie tak ..., lecz raczej tak:\\  
Nowy Jork, Tokio, Budapeszt, \ldots

Nie tak ..., lecz raczej tak:  
Nowy Jork, Tokio, Budapeszt, ...

#### 2.4.8. Ligatury

Ligatury (spójki) to znaki graficzne, w których połączono dwie lub trzy litery. W niektórych językach ligatury występują jako właściwe danej ortografii znaki pisma, np. *œ* w języku francuskim. Większość ligatur tworzy się ze względów estetycznych lub zwyczajowych.  $\text{\LaTeX}$  zna pięć następujących ligatur:

`ff fi fl ffi ffl` zamiast `ff fi fl ffi ffl`

$\text{\TeX}$  używa ligatur automatycznie. Można temu zapobiec, między znakami tworzącymi ligaturę wstawiając instrukcję `\mbox{}`:

Jak lepiej: geografii czy  
`geograf\mbox{ }ii?`

Jak lepiej: geografii czy geografii?

### 2.4.9. Akcenty i znaki specjalne

W  $\text{\LaTeX}$ u istnieją metody wstawiania znaków akcentowanych oraz spotykanych w różnych językach znaków specjalnych. W tabeli 2.2 zestawiono instrukcje akcentów. Użyto ich do litery „o”, ale można je również stosować do dowolnej innej litery.

W wypadku akcentów nad literami „i” oraz „j” należy znad tych liter usunąć kropkę. Służą do tego instrukcje  $\backslash i$  i  $\backslash j$ , wstawiające do składu specjalne wersje liter „i” oraz „j”.

```
H\^otel, na\"i ve, \'el\'eve,\\
sm\o rrebr\o d, !'Se\~norita!,\\
Sch\"onbrunner Schlo\ss{}
Stra\ss e
```

```
Hôtel, naïve, élève,
smørrebrød, ¡Señorita!,
Schönbrunner Schloß Straße
```

Tabela 2.2: Akcenty i znaki specjalne

ò	$\backslash 'o$	ó	$\backslash 'o$	ô	$\backslash ^o$	õ	$\backslash ~o$
-o	$\backslash =o$	ô	$\backslash .o$	ö	$\backslash "o$		
ö	$\backslash u o$	ö	$\backslash v o$	ő	$\backslash H o$	q	$\backslash c o$
q	$\backslash d o$	q	$\backslash b o$	ôo	$\backslash t oo$		
œ	$\backslash oe$	Œ	$\backslash OE$	æ	$\backslash ae$	Æ	$\backslash AE$
å	$\backslash aa$	Å	$\backslash AA$	ą	$\backslash k\{a\}$		
ø	$\backslash o$	Ø	$\backslash O$	ł	$\backslash l$	Ł	$\backslash L$
ı	$\backslash i$	Ĵ	$\backslash j$	ı	$\backslash 'ı$	ı	$\backslash ?ı$

## 2.5. $\text{\LaTeX}$ wielojęzyczny

Jeżeli  $\text{\LaTeX}$  ma składać tekst w językach innych niż angielski, to zasadniczo należy go dostosować w trzech następujących obszarach:

1.  $\text{\LaTeX}$  musi poznać reguły dzielenia wyrazów dla danego języka, co wiąże się z koniecznością stworzenia tak zwanego formatu  $\text{\LaTeX}$ a – z nowymi wbudowanymi weń regułami. Nie jest to zbyt trudne zadanie, ale szczegółowy sposób postępowania zależy od używanej dystrybucji. Więcej informacji na ten temat można znaleźć w tej części dokumentacji każdej dystrybucji  $\text{\LaTeX}$ a, która dotyczy instalowania systemu.
2. Wszystkie teksty generowane przez  $\text{\LaTeX}$ a automatycznie trzeba przystosować do danego języka. Dotyczy to: tytułów rozdziałów, spisu treści, spisu rysunków, tabel, dat, itp. Zmiany te umożliwia pakiet `babel` Johanna Braamsa.

3. Należy włączyć specyficzne dla danego języka reguły typograficzne. Na przykład w języku francuskim każdy dwukropek i wykrzyknik trzeba poprzedzić odstępem, a w języku polskim po numerach w tytułach rozdziałów i punktów stawia się kropkę.

Jeżeli dysponujemy dobrze skonfigurowanym  $\text{\LaTeX}$ em, czyli  $\text{\LaTeX}$ em z wygenerowanym formatem zawierającym odpowiednie reguły przenoszenia wyrazów, to resztę zadań polonizacyjnych załatwi pakiet `babel`. Wystarczy w tym celu do preambuły dokumentu wpisać instrukcję:

```
\usepackage{polish}
```

Jej ogólną postacią jest:

```
\usepackage[lista-języków]{babel}
```

Argument *lista-języków* to oddzielone przecinkami nazwy języków, które obsługuje zainstalowana wersja  $\text{\LaTeX}$ a. Ostatni na liście jest językiem domyślnym. Do przełączenia się w treści dokumentu na inny język służy polecenie:

```
\selectlanguage{język}
```

Jeśli używana wersja  $\text{\LaTeX}$ a nie obsługuje języka z listy, to `babel` zadziała z wyłączonym przenoszeniem wyrazów, co znacznie pogorszy jakość składu.

Pakiet `babel` dla każdego języka definiuje elementy wpisywane automatycznie przez program (np. dla języka polskiego „Spis treści” zamiast „Table of contents”) oraz udostępnia polecenia ułatwiające przygotowanie dokumentów w tym języku.

Dla niektórych języków `babel` udostępnia specjalne instrukcje, ułatwiające wprowadzanie znaków diakrytycznych i specjalnych. Teksty w języku niemieckim zawierają na przykład sporo liter z umlautami: (äöü). Wykorzystując pakiet `babel`, można wprowadzić literę ö, wpisując "o zamiast `\"o`.

W większości systemów komputerowych znaki akcentowane i specjalne (czyli znaki o kodach ASCII większych niż 127) można wprowadzać bezpośrednio z klawiatury. Przykładowo, polskie znaki diakrytyczne można wprowadzać, naciskając klawisz prawy-Alt i jednocześnie klawisz z odpowiednią literą.  $\text{\LaTeX}$  radzi sobie z takimi znakami. Począwszy od grudnia 1994 r. dystrybucje  $\text{\LaTeX}$ a zawierają pakiet `inputenc`, pozwalający kodować znaki diakrytyczne w różnych wariantach. Przykładowo, jeśli dokument jest kodowany w standardzie ISO 8859-2 (system operacyjny Unix/Linux), to pakiet `inputenc` należy dołączyć do dokumentu w następujący sposób:

```
\usepackage[latin2]{inputenc}
```

W wypadku dokumentu kodowanego w standardzie CP 1250 (system operacyjny MS Windows) powinniśmy zamiast opcji `latin2` wpisać `cp1250`. Dla dokumentów unikodowych należy użyć opcji `utf8`.

Chociaż pakiety `babel` oraz `inputenc` umożliwiają skład dokumentów w języku polskim, to nie są pozbawione wad. Kłopoty mogą sprawiać dokumenty

o rozbudowanej strukturze, na przykład zawierające skorowidze. Inne podejście do sprawy języka polskiego w  $\LaTeX$ u jest przedstawione w punkcie 2.5.1.

Kodowanie znaków ma znaczenie nie tylko w pliku źródłowym. Drugą stroną medalu jest układ znaków w foncie, czyli kodowanie fontu. Określa ono, w których miejscach *fontu* znajdują się poszczególne znaki. Standardowo  $\LaTeX$  używa kodowania o nazwie `OT1`, przyjętego dla oryginalnych  $\TeX$ owych fontów Computer Modern (CM). Są to fonty jedynie 128-znakowe, nie zawierają na przykład charakterystycznych polskich liter. Znak diakrytyczny można w nich skonstruować metodą nałożenia dwóch innych: litery i odpowiedniego akcentu. Ta metoda ma wady, bo  $\TeX$  nie może poprawnie przenosić wyrazów zawierających tak zapisane znaki diakrytyczne, kiepska jest też jakość typograficzna takich diakrytyków.

Na szczęście wszystkie współczesne dystrybucje  $\TeX$ a zawierają komplet fontów *European Computer Modern* (EC). Są to fonty zawierające do 256 znaków w kodowaniu `T1`. Pierwszych 128 znaków fontu EC jest (prawie) identycznych jak w odpowiadającym mu foncie CM. Pozostałe 128 znaków to znaki diakrytyczne występujące w różnych językach europejskich, w tym też komplet znaków niezbędnych do składania tekstów polskich. Fonty EC umożliwiają poprawne przenoszenie wyrazów, znacznie lepsza jest też jakość typograficzna znaków diakrytycznych.

Polscy użytkownicy mogą też korzystać z rodziny fontów PL (autorzy B. Jackowski, M. Ryćko, J. Nowacki) oraz nowszych fontów LM (autorzy B. Jackowski i J. Nowacki). W zakresie objętym przez rodzinę CM fonty PL/LM są całkowicie z nią zgodne, a dodatkowo zawierają wszystkie polskie znaki diakrytyczne. Polskie diakrytyki w fontach EC są kopią odpowiednich znaków z fontów PL, czyli „ogonki” wyglądają tak samo, niezależnie od tego, z której z rodzin korzystamy.

W fontach EC, PL i LM dostępne są ponadto znaki cudzysłówów francuskich i polskiego otwierającego, których nie ma w fontach CM. Opisany w punkcie 2.4.1 sposób wprowadzania tych znaków za pomocą par `, , , <<` i `>>` działa jedynie wówczas, gdy używamy fontów EC, PL bądź LM.

Aby przełączyć się na fonty EC lub PL, trzeba do preambuły dokumentu dołączyć pakiet `fontenc`:

```
\usepackage[T1]{fontenc} lub \usepackage[OT4]{fontenc}
```

Argument `T1`, określający kodowanie, jest „odpowiedzialny” za przełączenie się na fonty EC. Podobnie Argument `OT4` włącza fonty PL.

Uwaga: Powyższe dołączenie pakietu `fontenc` jest jedynie deklaracją. Jeżeli twoja dystrybucja  $\LaTeX$ a nie zawiera fontów, które chcesz włączyć, to powyższe polecenia nie zostaną wykonane, a  $\LaTeX$  przełączy się na font domyślny, zwykle nie zawierający polskich znaków.

Reasumując, oto przykładowa preambuła artykułu składanego w języku polskim przy wykorzystaniu pakietów `babel`, `inputenc` oraz `fontenc`:

```
\documentclass{article}
\usepackage[polish]{babel}
\usepackage[cp1250]{inputenc}
\usepackage[OT4]{fontenc} %% lub [T1]
```

Lepszy sposób polonizacji L<sup>A</sup>T<sub>E</sub>Xa opisano w następnym punkcie.

Rodzina fontów LH zawiera litery potrzebne do składania dokumentów w cyrylicy. Ze względu na dużą liczbę znaków w różnych pismach cyrylickich są one zgrupowane w czterech różnych kodowaniach: T2A, T2B, T2C, i X2<sup>4</sup>. Rodzina CB, w kodowaniu LGR, zawiera fonty do składu greki.

### 2.5.1. Język polski w dokumentach

Jak wspomniano w punkcie 2.5, poprawny skład w języku polskim wymaga trzech rzeczy: L<sup>A</sup>T<sub>E</sub>Xa z wbudowanymi w format polskimi regułami przenoszenia wyrazów, fontów zawierających polskie znaki diakrytyczne oraz dodatkowego pakietu obsługującego specyficzne dla języka polskiego reguły typograficzne. W tym punkcie zakładamy, że udało ci się skonfigurować L<sup>A</sup>T<sub>E</sub>Xa pod kątem pierwszych dwóch punktów, tj. wygenerowania formatu i zainstalowania fontów PL lub EC, i koncentrujemy się na szczegółowym opisie ostatniego aspektu polonizacji.

W każdej dystrybucji L<sup>A</sup>T<sub>E</sub>Xa znajduje się wspomniany w punkcie 2.5 pakiet `babel`. Mimo występujących w nim niedociągnięć można go polecić, szczególnie początkującym. Bardziej wymagający użytkownicy piszący po polsku powinni korzystać z opisanego dalej zestawu polonizacyjnego `platex`.

W wypadku języka polskiego wiele kłopotów sprawia kodowanie znaków diakrytycznych. Wynika to przede wszystkim z braku standardu: różne platformy systemowe promują w tym zakresie różne rozwiązania<sup>5</sup>. Można wyróżnić dwa sposoby zapisu diakrytyków w L<sup>A</sup>T<sub>E</sub>Xu: polecenia standardowe (opisane w punkcie 2.4.9 oraz poniżej) i notację „bezpośrednią”, posługującą się znakami o kodach większych od 127<sup>6</sup>.

Standardowe polecenia akcentowe L<sup>A</sup>T<sub>E</sub>Xa umożliwiają zapis wszystkich polskich znaków diakrytycznych w następującej postaci<sup>7</sup>:

<sup>4</sup>Listę języków obsługiwanych w tych kodowaniach można znaleźć w [26].

<sup>5</sup>Lekarstwem na tę bolączkę może być kodowanie wielobajtowe, tj. standard Unicode (UTF). Wprawdzie L<sup>A</sup>T<sub>E</sub>X potrafi przetwarzać dokumenty unikodowe jedynie w ograniczonym zakresie, ale jest on wystarczający dla tekstów w językach europejskich. Specjalna wersja T<sub>E</sub>Xa, XeT<sub>E</sub>X autorstwa Jonathana Kew, potrafi przetwarzać dokumenty zakodowane w unikodzie, a także generować skład z użyciem unikodowych fontów OpenType.

<sup>6</sup>Jest jeszcze trzeci sposób: tak zwana notacja prefiksowa (*/a* – *ą*, */n* – *ń*, */S* – *Ś* itp.). Metoda ta wyszła już w zasadzie z użytku, gdyż powszechnie dostępna jest metoda bezpośredniego wprowadzania znaków z polskimi ogonkami z klawiatury.

<sup>7</sup>Z notacją tą wiąże się jednak pewien problem: otoczenie `tabbing` zmienia lokalnie definicję kilku makr, w tym `\'`. Dlatego w jego obrębie do uzyskiwania znaków z akcentem *acute* trzeba używać notacji typu `\a'o`. Konsekwencją jest to, że zarówno w implementacji notacji prefiksowej, jak i „stron kodowych” pakietu `inputenc` trzeba się do akcentu *acute* dostawać nieco naokoło.



```
\k{a} \c{k}{e} \l{f} \n \o \s
\z \z \k{A} \c{C} \k{E} \l{f} \N
\O \S \Z \Z
```

ą ć ę ł ń ó ś ź ż Ą Ć Ę Ł Ń Ó Ś Ź Ż
-------------------------------------

Posługiwanie się powyższymi poleceniami do pisania tekstów po polsku jest uciążliwe, ale przydaje się na przykład do wstawienia niewielkich fragmentów do dokumentu, który będzie przetwarzany przez kogoś, kto języka polskiego nie zna i nie ma dobrze skonfigurowanej pod tym kątem instalacji (np. polskie wstawki w artykule konferencyjnym pisanym po angielsku).

Bezpośrednie wprowadzanie polskich znaków umożliwia mechanizm przekodowywania, uruchamiany przez umieszczenie komentarza strukturalnego w pierwszym wierszu pliku<sup>8</sup>:

```
%& --translate-file=cp1250pl
```

Wpis taki będzie poprawny w wypadku redagowania plików w systemie MS Windows i stosowania domyślnego w nim kodowania CP 1250. W wypadku systemu Unix/Linux należy zamiast `cp1250pl` wpisać `il2-pl` (przy założeniu, że posługujemy się kodowaniem ISO 8859-2).

Komentarz strukturalny jest alternatywą dla pakietu `inputenc`, opisanego w punkcie 2.5<sup>9</sup>. Mechanizm komentarza strukturalnego nie umożliwia poprawnego przetwarzania dokumentów unikodowych. Jeżeli upierasz się przy unikodzie to pozostaje ci tylko pakiet `inputenc`.

Zamiast pakietu `babel` można zastosować pakiet `polski` z zestawu `platex` (autorzy Mariusz Olko i Marcin Woliński). Jego niewątpliwą zaletą jest staranniejsza polonizacja, np. pakiet `polski` domyślnie przełącza się na fonty PL bez potrzeby dołączania pakietu `fontenc`. W wypadku przejścia z pakietu `babel` na `polski` dokumenty nie wymagają modyfikacji, oprócz oczywistej wymiany nazwy ładowanego pakietu w preambule.

Pakiet `polski` dołączamy w preambule dokumentu poleceniem:

<code>\usepackage[opcje]{polski}</code>
---

Lista ważniejszych opcji obejmuje:

- OT4 wybranie kodowania OT4 fontów, co w praktyce oznacza skład fontami PL;
- T1 wybranie kodowania T1 fontów, co oznacza skład fontami EC;
- OT1 wybranie kodowania OT1 fontów, co oznacza skład fontami CM (niezalecane);
- `plmath` włączenie polskich oznaczeń dla standardowych poleceń trygonometrycznych oraz symboli relacji mniejszy-lub-równy i większy-lub-równy (zalecane);

<sup>8</sup>Komentarz ten należy umieścić w pierwszym wierszu pliku, czyli jeszcze przed wierszem z `\documentclass`, a znak `%` musi być pierwszym znakiem tego wiersza.

<sup>9</sup>Oznacza to, że jednoczesne użycie komentarza strukturalnego `translate-file` i dołączanie pakietu `inputenc` jest błędem

`nomathsymbols` blokada zmiany znaczenia standardowych poleceń trygonometrycznych oraz symboli relacji mniejszy-lub-równy i większy-lub-równy (por. punkt 3.10, str. 59);

`MeX` tryb pełnej polonizacji (zalecane).

Jeżeli opcję układu kodowania w foncie pominięto, to pakiet `polski` używa fontów PL (w wypadku ich braku  $\text{\LaTeX}$  będzie sygnalizował błędy). Dotyczy to zarówno fontów tekstowych, jak i zawierających znaki matematyczne. W instalacji zawierającej fonty PL dołączenie pakietu `polski` bez opcji jest równoważne poleceniu:

```
\usepackage[OT4,plmath]{polski}
```

Polecenie `\selecthyphenation` pozwala przełączyć się na dany zestaw wzorców dzielenia wyrazów. Jest to odpowiednik polecenia `\selectlanguage` z pakietu `babel`. Argumentem jest nazwa języka.

Polonizacyjnym uzupełnieniem pakietu `polski` jest zestaw klas Marcina Wolińskiego `mwart`, `mwrep` i `mwbook`, dostępny pod adresem <http://www.mimuw.edu.pl/~wolinski/mwcls.html>. W klasach tych zostały uwzględnione m.in. takie zwyczaje jak: umieszczanie kropek po numerach śródtytułów, sposób formatowania przypisów oraz pagin, zakaz przenoszenia słów w śródtytułach, reguły umieszczania/pomijania paginacji. Uzyskano to za cenę znacznej ingerencji w sposób działania klas standardowych, czego skutkiem jest niekompatybilność z częścią pakietów  $\text{\LaTeX}$ a.

Oto przykładowa preambuła artykułu składanego w klasie `mwart`:

```
%& --translate-file=cp1250pl
\documentclass{mwart}
\usepackage[MeX]{polski}
\begin{document} ...
```

Jest to zalecany sposób rozpoczynania dokumentów w języku polskim. Warto z niego korzystać, zaopatrując się w niezbędne elementy: fonty PL, pakiet `polski` i klasy Marcina Wolińskiego. W standardowej dystrybucji  $\text{\LaTeX}$ a, w której na ogół znajdują się obecnie fonty PL, zadziała natomiast taka preambuła:

```
%& --translate-file=cp1250pl
\documentclass{article}
\usepackage{polski}
```

## 2.6. Odstępy między wyrazami

Aby wyrównać prawy margines,  $\text{\LaTeX}$  wstawia między słowami odstępy różnej wielkości. Odstęp wstawiany na końcu zdania jest trochę większy, ponieważ tak składa się książki w krajach anglosaskich.  $\text{\LaTeX}$  zakłada, że zdania mogą się kończyć kropką, znakiem zapytania lub wykrzyknikiem.

Jeżeli bezpośrednio przed kropką znajduje się duża litera, to  $\text{\LaTeX}$  nie traktuje takiego miejsca jako końca zdania, lecz jako kropkę po skrócie.

Wyjątki od powyższych zasad trzeba wyraźnie zaznaczyć w tekście. Znak  $\backslash$  poprzedzający spację oznacza odstęp normalnej wielkości. Tylda  $\sim$  również wstawia taki odstęp, z tym że  $\text{\LaTeX}$ owi nie wolno na nim złamać wiersza. Umieszczenie instrukcji  $\backslash@$  przed kropką jest dla  $\text{\LaTeX}$ a wskazówką, że ta kropka kończy zdanie, nawet jeśli następuje po dużej literze.

```
Pan~Kowalski ucieszył się\\
na jej widok (zob.~Rys.~5).\\
Podoba mi się JAVA\@. A~tobie?
```

```
Pan Kowalski ucieszył się
na jej widok (zob. Rys. 5).
Podoba mi się JAVA. A tobie?
```

Jak wspomniano, wstawianie większych odstępów na końcu zdań to zwyczaj anglosaski. W Europie kontynentalnej tradycyjnie się tego nie robi. Wstawianie większych odstępów na końcu zdań można wyłączyć poleceniem:

```
\frenchspacing
```

Pakiet `polski` domyślnie wykonuje instrukcję `\frenchspacing` za nas, czyli włącza odstęp „kontynentalne”.

## 2.7. Tytuły, śródtytuły i punkty

Podzielenie dokumentu na rozdziały, punkty, podpunkty itd. pomaga czytelnikom lepiej orientować się w tekście. Do dzielenia dokumentu na hierarchiczne części służą odpowiednie instrukcje  $\text{\LaTeX}$ owe. Do autora należy używanie tych poleceń w odpowiednim porządku.

W klasie `article` mamy do dyspozycji następujące instrukcje podziału hierarchicznego:

```
\section{...}           \paragraph{...}
\subsection{...}        \subparagraph{...}
\subsubsection{...}     \appendix
```

W klasach `report` (raport) i `book` (książka) mogą występować rozdziały:

```
\chapter{...}
```

Jeśli raport bądź książkę trzeba podzielić na części bez naruszania numeracji punktów i rozdziałów, to można użyć polecenia:

```
\part{...}
```

Ponieważ w klasie `article` najwyższą jednostką w hierarchii podziału jest `\section` (czyli *punkt*), łatwo tworzy się książki (klasa `book`), w których rozdziałami są poszczególne artykuły.  $\text{\LaTeX}$  dobierze za nas odpowiednie odstęp między rozdziałami oraz wielkość i krój pisma w śródtytułach.

Dwie z wymienionych instrukcji działają nieco inaczej niż pozostałe:

- instrukcja `\part` nie ma wpływu na numerację rozdziałów;
- instrukcja `\appendix` nie ma argumentów. Jest to deklaracja zmieniająca sposób numerowania z cyfr na litery. Dotyczy to rozdziałów w klasach `book` i `report`, a punktów w klasie `article`.

Argumentu instrukcji podziału dokumentu  $\text{\LaTeX}$  używa do przygotowania spisu treści. Instrukcja:

```
\tableofcontents
```

wstawia spis treści w miejscu jej użycia. Aby w spisie treści otrzymać poprawne numery stron, trzeba dokument przetworzyć („zlatechować”) dwukrotnie. Czasami niezbędna jest nawet trzecia kompilacja. Kolejny przebieg jest potrzebny, gdy pod koniec przetwarzania dokumentu  $\text{\LaTeX}$  pokazuje komunikat:

LaTeX Warning: Label(s) may have changed.

Rerun to get cross-references right.

$\text{\LaTeX}$  przetwarza dokument strona po stronie i w pojedynczym przebiegu nie może wstawić spisu treści na początku dokumentu, ponieważ nie jest jeszcze znana jego treść ani numeracja stron. Podobnie ma się sprawa ze spisami tabel czy rysunków. Rozwiązanie tego problemu jest tyleż proste co skuteczne. Podczas przetwarzania dokumentu  $\text{\LaTeX}$  zapisuje odpowiednie informacje do plików pomocniczych – w celu ich wykorzystania podczas kolejnych przebiegów.

Przeznaczenie danego pliku pomocniczego jest sygnalizowane przez rozszerzenie jego nazwy. I tak: plik o rozszerzeniu `.toc` zawiera spis treści, plik `.lot` – spis tabel, plik `.lof` – spis rysunków, `.aux` – informacje o odsyłaczach wewnątrz dokumentu (odsyłacze omawiamy w punkcie 2.8). Pełniejszy wykaz nazw plików  $\text{\LaTeX}$ owych podano w punkcie 1.7 na stronie 12.

Wymienione wyżej instrukcje podziału hierarchicznego posiadają także wersje „z gwiazdką”. Nazwa instrukcji w wersji „z gwiazdką” składa się z „normalnej” nazwy, po której występuje znak „\*”. W wyniku działania takiej instrukcji tytuł rozdziału lub punktu zostanie umieszczony w dokumencie, ale nie w spisie treści; tytuł nie zostanie też objęty numeracją. Przykładowo, wersją „z gwiazdką” instrukcji `\section{Pomoc}` jest `\section*{Pomoc}`.

Zwyczajem angielskim jest nieumieszczanie nienumerowanych śródtytułów w spisie treści. Polscy redaktorzy często się domagają, by na przykład „Wstęp” był śródtytułem nienumerowanym, a jednocześnie występował w spisie. Ten problem rozwiązujemy za pomocą polecenia:

```
\addcontentsline{spis}{poziom}{śródtytuł}
```

gdzie: *spis* to rozszerzenie nazwy pliku, w którym ma zostać zapisana informacja, *poziom* to `chapter`, `section` bądź inna nazwa polecenia hierarchicznego, a *śródtytuł* to sam śródtytuł. Przykład:

```
\chapter*{Wstęp}
\addcontentsline{toc}{chapter}{Wstęp}
```

Najczęściej hasła w spisie treści pokrywają się z tytułami rozdziałów czy punktów. Czasami jednak nie jest to pożądane, na przykład wówczas, gdy tekst hasła jest zbyt długi. W takich wypadkach hasło do spisu treści można podać jako *opcjonalny* argument instrukcji podziału hierarchicznego, na przykład tak:

```
\chapter[Krótki i~ekscytujący rozdział]{To jest
bardzo długi i~wyjątkowo nudny rozdział}
```

W wyniku wykonania tej instrukcji w spisie treści pojawi się „Krótki i ekscytujący rozdział”, natomiast w tytule rozdziału „To jest bardzo długi i wyjątkowo nudny rozdział”.

L<sup>A</sup>T<sub>E</sub>X składa część tytułową dokumentu, napotkawszy instrukcję:

```
\maketitle
```

Należy ją umieścić po `\begin{document}`, czyli nie w preambule.

Zawartość części tytułowej ustalają polecenia:

```
\title{...}, \author{...} oraz opcjonalnie \date{...}
```

Należy je umieścić w preambule. Jeżeli dokument ma kilku autorów, to ich nazwiska i imiona rozdzielamy instrukcją `\and`. Sposób użycia powyższych instrukcji demonstruje rysunek 1.2 ze strony 8.

W L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> istnieją trzy dodatkowe instrukcje dotyczące struktury dokumentu, dostępne jednak wyłącznie w klasie `book`. Oto ich nazwy, sposób użycia i przeznaczenie:

`\frontmatter` powinna być pierwszą instrukcją w treści dokumentu, czyli powinna wystąpić tuż po `\begin{document}`. Włącza ona rzymski zapis numerów stron, wyłączając jednocześnie numerowanie punktów podziału. To tak, jakby się używało instrukcji „gwiazdkowanych”, w rodzaju `\chapter*{Preface}`. Punkty podziału trafią jednak do spisu treści.

`\mainmatter` należy umieścić tuż przed pierwszym rozdziałem książki. Przełącza ona sposób oznaczania numerów stron na arabski, zerując zarazem licznik stron.

`\backmatter` powinna wystąpić przed ostatnimi fragmentami książki, takim jak spis literatury albo skorowidz.

Powyższe instrukcje przydają się do podzielenia książki na część wstępną (obejmującą tytułaturę, spisy treści, tabel, wstępy itd.), główną i zakończenie (załączniki, skorowidze, kolofon itd.). W części wstępnej tytuły rozdziałów są składane mniejszym stopniem pisma (czego należy oczekiwać), a numery stron są w notacji rzymskiej (co raczej odbiega od polskich zwyczajów typograficznych).

## 2.8. Odsyłacze

Książki, raporty i artykuły często zawierają odsyłacze do rysunków, tabel i innych fragmentów tekstu. Z odsyłaczami związane są w  $\text{\LaTeX}$ u następujące trzy instrukcje:

```
\label{etykieta}, \ref{etykieta} i \pageref{etykieta}
```

Argument *etykieta* jest ciągiem liter, cyfr lub znaków interpunkcyjnych. Nazwy etykiet ustala sam autor.  $\text{\LaTeX}$  zamienia  $\text{\ref{etykieta}}$  na numer tego rozdziału, punktu, rysunku, tabeli czy też równania matematycznego, *bezpośrednio* za którym umieszczona została instrukcja  $\text{\label}$  zawierająca identyczną *etykietę*. Instrukcja  $\text{\pageref{etykieta}}$  działa identycznie jak  $\text{\ref}$ , z tym że wstawia numer strony, na której znajduje się element oznaczony etykietą<sup>10</sup>. Oto przykład:

Odsyłacz do tego punktu  
 $\text{\label{sec:this}}$  wygląda tak:  
 „patrz punkt~ $\text{\ref{sec:this}}$  na  
 stronie~ $\text{\pageref{sec:this}}$ .”

Odsyłacz do tego punktu wygląda tak: „patrz  
 punkt 2.8 na stronie 34.”

Podobnie jak w wypadku spisów treści, tabel czy rysunków, do ustalenia właściwej numeracji odsyłaczy potrzebne są co najmniej dwie, a z reguły trzy kompilacje dokumentu. Podczas pierwszej  $\text{\LaTeX}$  wysyła do pliku pomocniczego z rozszerzeniem *.aux* (zob. punkt 1.7) informacje o odsyłaczach, które wykorzystuje podczas kolejnych kompilacji.

## 2.9. Przypisy

Do składania przypisów u dołu strony służy instrukcja:

```
\footnote{tekst przypisu}
```

Należy ją wstawić bezpośrednio po słowie lub zdaniu, do którego się odnosi. W krajach anglosaskich przypisy odnoszące się do całego zdania lub jego części umieszcza się natychmiast po kropce lub przecinku. W Polsce najczęściej umieszcza się je *przed* znakiem przestankowym (zasadę tę stosujemy w niniejszym tłumaczeniu).

$\text{\LaTeX}$  numeruje przypisy automatycznie. Sposób ich numerowania zależy od używanej klasy. W klasie **article** numeracja jest ciągła, w klasach **report** i **book** przypisy są numerowane w ramach rozdziałów.

<sup>10</sup>Warto pamiętać, że te instrukcje „nie wiedzą”, do czego tak naprawdę się odnoszą. Zadaniem instrukcji  $\text{\label}$  jest przechowanie związku wygenerowanej automatycznie liczby z miejscem w tekście.

Przypisy\footnote{To jest właśnie  
przypis.} są często  
stosowane przez  
użytkowników {\LaTeX}a

Przypisy<sup>a</sup> są często stosowane przez użytkowników L<sup>A</sup>T<sub>E</sub>Xa

<sup>a</sup>To jest właśnie przypis.

## 2.10. Wyróżnienia

W tekstach pisanych na maszynie fragmenty, które mają zostać wyróżnione, podkreśla się. W dokumentach drukowanych wyróżnienie fragmentu odbywa się przez złożenie go *kursywą*. Służy do tego L<sup>A</sup>T<sub>E</sub>Xowa instrukcja:

```
\emph{tekst}
```

Argumentem tej instrukcji jest tekst, który ma zostać wyróżniony.

```
\emph{\emph{Wyróżnienia} w tekście  
już wyróżnionym są składane  
\emph{zwykłym} krojem pisma.}
```

Wyróżnienia w tekście już wyróżnionym są składane zwykłym krojem pisma.

Zwróćmy uwagę, że istnieje różnica między wyróżnieniem części tekstu a złożeniem go inną czcionką:

```
\textit{Tekst można \emph{wyróżnić},  
składając go kursywą,}  
\textsf{czcionką \emph{szeryfową},}  
\texttt{a nawet \emph{maszynowo}.}
```

Tekst można wyróżnić, składając go kursywą, czcionką szeryfową, a nawet maszynowo.

## 2.11. Otoczenia

Do instrukcji formatujących zaliczają się *otoczenia* (zwane też *środowiskami*), czyli instrukcje postaci:

```
\begin{nazwa} tekst \end{nazwa}
```

gdzie *nazwa* jest nazwą otoczenia, a *tekst* jest fragmentem dokumentu, który ma zostać złożony inaczej niż poza otoczeniem.

Otoczenia można zagnieżdżać jedno w drugim:

```
\begin{aaa}...\begin{bbb}...\end{bbb}...\end{aaa}
```

Niedopuszczalne jest natomiast „przeplatanie” otoczeń:

```
\begin{aaa}...\begin{bbb}...\end{aaa}...\end{bbb}
```

Wiele standardowych poleceń L<sup>A</sup>T<sub>E</sub>Xa można zapisać w formie „otoczeniowej”. W takich wypadkach nazwa polecenia (bez w-tył-ciacha) jest nazwą otoczenia. Na przykład, zamiast polecenia `\em`, włączającego wyróżniający krój pisma, możemy zastosować otoczenie `\begin{em}...\end{em}`.

W kolejnych punktach przedstawiamy częściej używane otoczenia.

### 2.11.1. Otoczenia `itemize`, `enumerate` i `description`

Otoczenia `itemize` oraz `description` służą do tworzenia wyszczególnień, zaś `enumerate` do tworzenia wyliczeń. W każdym z nich element wyliczenia zaczyna się od instrukcji `\item`.

```
\begin{enumerate}
\item Taka lista:
\begin{itemize}
\item wygląda
\item[--] śmiesznie.
\end{itemize}
\item Pamiętaj:
\begin{description}
\item[Głupoty] nie staną się mądrościami,
  gdy się je wyliczy.
\item[Mądrości] można elegancko
  zestawiać w wyliczeniach.
\end{description}
\end{enumerate}
```

1. Taka lista:
  - wygląda
  - śmiesznie.
2. Pamiętaj:
 

**Głupoty** nie staną się mądrościami,  
gdy się je wyliczy.

**Mądrości** można elegancko zestawiać  
w wyliczeniach.

### 2.11.2. Otoczenia `flushleft`, `flushright` i `center`

W otoczeniach `flushleft` i `flushright` akapity są składane z wyrównaniem, odpowiednio, do lewego bądź prawego marginesu. Wewnątrz otoczenia `center` każdy wiersz akapitu jest wyśrodkowany w osi szpalty. Tak jak zawsze,  $\text{\LaTeX}$  dzieli akapity na wiersze automatycznie, można jednak w obrębie powyższych otoczeń wymusić zmianę wiersza poleceniem `\\`.

```
\begin{flushleft}
To jest tekst\\ wyrównany do lewej.
{\LaTeX} nie składa tu wierszy\\
z zachowaniem jednakowej długości.
\end{flushleft}
```

To jest tekst  
wyrównany do lewej.  $\text{\LaTeX}$  nie składa tu  
wierszy  
z zachowaniem jednakowej długości.

```
\begin{flushright}
To jest tekst\\ wyrównany do prawej.
{\LaTeX} nie składa tu wierszy\\
z zachowaniem jednakowej długości.
\end{flushright}
```

To jest tekst  
wyrównany do prawej.  $\text{\LaTeX}$  nie składa tu  
wierszy  
z zachowaniem jednakowej długości.

```
\begin{center}
To jest tekst\\wyśrodkowany.
\end{center}
```

To jest tekst  
wyśrodkowany.



### 2.11.3. Otoczenia `quote`, `quotation` i `verse`

Otoczenie `quote` przydaje się do składania cytatów oraz przykładów:

Jeżeli chodzi o długość wierszy,  
to regułą kciuka jest, że:

```
\begin{quote}
```

Przeciętnie wiersz nie powinien  
zawierać więcej niż 66 znaków.

Dlatego w `{\LaTeX}`u standardowe  
strony mają szerokie marginesy.  
`\end{quote}`

Dlatego też w gazetach stosuje  
się druk wielołamowy.

Jeżeli chodzi o długość wierszy, to regułą  
kciuka jest, że:

Przeciętnie wiersz nie powinien  
zawierać więcej niż 66 znaków.

Dlatego w `\LaTeX`u standardowe  
strony mają szerokie marginesy.

Dlatego też w gazetach stosuje się druk wie-  
łołamowy.

Istnieją ponadto dwa otoczenia o podobnym zastosowaniu: `quotation` oraz `verse`. Pierwsze z nich przydaje się do formatowania cytatów dłuższych niż jeden akapit. W przeciwieństwie do otoczenia `quote`, wewnątrz `quotation` `\LaTeX` rozpoczyna poszczególne akapity od wcięcia akapitowego. Otoczenie `verse` służy do składania wierszy. Poszczególne linijki zwrotek należy kończyć instrukcją `\\`, poszczególne zaś zwrotki – oddzielać pustą linią.

Na pamięć znam tylko jeden angielski  
wiersz. Ten o Humpty Dumptym.

```
\begin{flushleft}
```

```
\begin{verse}
```

Humpty Dumpty sat on a wall: \\

Humpty Dumpty had a great fall. \\

All the King's horses and all

the King's men \\

Couldn't put Humpty together again.

```
\end{verse}
```

```
\end{flushleft}
```

Na pamięć znam tylko jeden angielski wiersz.  
Ten o Humpty Dumptym.

Humpty Dumpty sat on a wall:

Humpty Dumpty had a great  
fall.

All the King's horses and all the  
King's men

Couldn't put Humpty together  
again.

### 2.11.4. Streszczenie

Publikacje naukowe zaczynają się zazwyczaj od streszczenia – przeglądu tego, co czytelnik napotka w dalszej części. W `\LaTeX`u do wyróżniania streszczeń służy otoczenie `abstract`. Używa się go na ogół w dokumentach klasy `article`.

```
\begin{abstract}
```

Streszczenie streszczenia.

```
\end{abstract}
```

Streszczenie streszczenia.

### 2.11.5. Symulacja maszynopisu

Tekst zawarty między `\begin{verbatim}` a `\end{verbatim}` jest przez  $\text{\LaTeX}$  składany dosłownie, czyli tak, by wyglądał jak napisany na maszynie, z zachowaniem zmian wiersza i odstępów z pliku źródłowego.

Aby uzyskać ten efekt, krój pisma zmienia się na imitujący pismo maszynowe (grotesk). Wszystkie znaki w tym kroju, włączając spację, mają jednakową szerokość. Zakończenie linii wewnątrz otoczenia `verbatim` prowadzi do rozpoczęcia nowego wiersza w wydruku, a *każda* spacja zamienia się na odstęp. Wewnątrz otoczenia `verbatim` *nie* są wykonywane instrukcje.

Wewnątrz akapitów imitację maszynopisu uzyskuje się za pomocą instrukcji:

```
\verb+tekst+
```

Znak `+` ogranicza tekst, który ma zostać wydrukowany dosłownie. Zamiast `+` można użyć innego znaku, byle to nie była litera, gwiazdka, spacja ani żaden znak, który występuje w *tekście*. Instrukcję `\verb` oraz otoczenie `verbatim` wykorzystujemy często w tej książce do składania przykładów  $\text{\LaTeX}$ owych.

Rozważmy przykład\ldots

```
\begin{verbatim}
{ for (i=1;i<=NF;i++) {l[$i]++; }
END {for (i in l) {print l[i]}
\end{verbatim}
```

Rozważmy przykład...

```
{ for (i=1;i<=NF;i++) {l[$i]++; }
END {for (i in l) {print l[i]}
```

Otoczenie `verbatim` oraz instrukcja `\verb` mają także wersje „z gwiazdką”, w których spacja z pliku źródłowego jest zamieniana na znak `_`. Jest to jedyna różnica działania w porównaniu do wersji bezgwiazdkowych:

```
\begin{verbatim*}
gwiazdkowa wersja
otoczenia verbatim
wyróżnia spacje
w tekście
\end{verbatim*}
```

```
gwiazdkowa_wersja
otoczenia_verbatim
wyróżnia_spacje
w_tekście
```

Otoczenia `verbatim` ani instrukcji `\verb` nie wolno używać wewnątrz argumentów innych instrukcji (więcej na ten temat w punkcie 2.13).

### 2.11.6. Otoczenie tabular

Do składania tabel służy otoczenie `tabular`.  $\text{\LaTeX}$  automatycznie ustala szerokość poszczególnych rubryk tabeli. Otoczenie ma jeden parametr obowiązkowy, *spec-kolumn*, który określa liczbę kolumn tabeli oraz sposób ich justowania:

```
\begin{tabular}{spec-kolumn}
```

Dla każdej kolumny należy w argumencie wstawić jedną z liter: `l`, `r` lub `c`, określając w ten sposób justowania zawartości kolumny. Dosunięcie zawartości kolumny do lewej oznaczamy literą `l`, do prawej – znakiem `r`, a wyśrodkowanie – znakiem `c`. Zapisu `p{szer-kolumn}` można użyć do zaznaczenia, że kolumna ma mieć szerokość *szer-kolumn*. Wewnątrz takiej kolumny tekst jest składany w prostokąt o zadanej szerokości, z wyrównywaniem obu marginesów. Znak `|` instruuje  $\text{\LaTeX}$ a, by kolumny tabeli rozdzielił pionową kreską.

Wewnątrz otoczenia `tabular` poszczególne wiersze oddzielamy instrukcją `\\`, a rubryki w wierszu – znakiem `&`. Instrukcja `\hline` wstawia poziomą kreskę na całą szerokość tabeli.

```
\begin{tabular}{|r|l|} \hline
7C0 & heksadecymalnie \\
3700 & oktalnie \\
11111000000 & binarnie \\
\hline \hline
1984 & dziesiętnie \\
\hline
\end{tabular}
```

7C0	heksadecymalnie
3700	oktalnie
11111000000	binarnie
1984	dziesiętnie

```
\begin{tabular}{|p{4.7cm}|} \hline
Ten akapit jest wewnątrz pudełka.
Mamy nadzieję, że uzyskany
efekt się podoba.\\ \hline
\end{tabular}
```

Ten akapit jest wewnątrz pudełka. Mamy nadzieję, że uzyskany efekt się podoba.

Instrukcję `@{...}` określamy odstęp między kolumnami. Zastępuje ona domyślny odstęp międzykolumnowy treścią umieszczoną między `{}` a `}`. Stosuje się ją często do wyrównywania zestawień liczbowych według cyfr znaczących. Można ją także wykorzystać do usunięcia odstępów w pierwszej i ostatniej kolumnie tabeli, co ilustruje poniższy przykład:

```
\begin{tabular}{@{} l @{}} \hline
bez odstępów na brzegach\\ \hline
\end{tabular}
```

bez odstępów na brzegach

```
\begin{tabular}{l} \hline
odstęp na brzegach tabeli\\ \hline
\end{tabular}
```

odstęp na brzegach tabeli

W  $\text{\LaTeX}$ u nie ma mechanizmu pozwalającego wyrównywać zestawienia liczbowe według cyfr znaczących<sup>11</sup>, ale efekt ten można uzyskać, składając liczbę w dwóch kolumnach: część całkowitą w kolumnie wyrównywanej do prawego brzegu i część dziesiętną w kolumnie wyrównywanej do lewego. Za pomocą instrukcji `@{,}` zastępujemy przecinkiem odstęp wstawiany normalnie między kolumnami. Trzeba jednak pamiętać o konieczności wpisywania znaku `&` zamiast przecinków w liczbach. Rubryki rozciągające się na kilka kolumn, jak nagłówek w poniższym przykładzie, tworzymy poleceniem `\multicolumn`:

<sup>11</sup>Do wyrównywania cyfr można skorzystać z pakietu `dcolumn` z zestawu „tools”.

```
\begin{tabular}{c r @{\,} l}
Wyrażenie &
\multicolumn{2}{c}{Wartość}\hline
 $\pi$  & 3,1416 & \\
 $\pi^\pi$  & 36,46 & \\
 $(\pi^\pi)^\pi$  & 80662,7 & \\
\end{tabular}
```

Wyrażenie	Wartość	
$\pi$	3,1416	
$\pi^\pi$	36,46	
$(\pi^\pi)^\pi$	80662,7	

Polecenie `\cline{m-n}` wstawia poziomą kreskę ciągnącą się od kolumny  $m$  do kolumny  $n$ :

```
\begin{tabular}{|c|c|c|c|l|}\hline
1 & \multicolumn{4}{c}{0}\hline
1 & 2 & 3 & 4 & 5 \\
1 & 2 & 3 & 4 & 5 \\
\hline
\end{tabular}
```

1	0			
1	2	3	4	5
1	2	3	4	5

W punkcie 6.8 opisano, jak sobie radzić ze składem bardziej skomplikowanych tabel.

## 2.12. Wstawki

Współczesne publikacje zawierają dużo rysunków i tabel. Elementów tych nie należy dzielić między strony i dlatego wymagają specjalnego potraktowania. W sytuacji gdy nie mieszczą się one na bieżącej stronie, na ogół przenosi się je i wstawia na początku strony następnej. W wypadku przeniesienia rysunku lub tabeli miejsce pozostałe na stronie wypełniane jest tekstem. Tego typu elementy będziemy nazywać *wstawkami*.

Aby w pełni skorzystać z  $\text{\LaTeX}$ owego mechanizmu wstawek, trzeba choćby powierzchownie rozumieć, jak  $\text{\LaTeX}$  manipuluje takimi obiektami. W przeciwnym razie wstawki mogą się stać źródłem irytacji, gdyż  $\text{\LaTeX}$  będzie je umieszczał wszędzie, tylko nie w miejscach, w którym byśmy sobie tego życzyli.

Do tworzenia wstawek mamy w  $\text{\LaTeX}$ u dwa otoczenia. Otoczenie **figure** służy do tworzenia rysunków, a otoczenie **table** – do tabel. Oba mają jeden parametr opcjonalny:

```
\begin{figure}[miejsce] albo \begin{table}[miejsce]
```

Argument *miejsce* określa, gdzie na stronie można umieścić wstawkę. Powinna to być sekwencja od jednego do pięciu znaków: **h**, **t**, **b**, **p** oraz **!**. Każdy znak określa dopuszczalny sposób umieszczenia wstawki; szczegółowe informacje na ten temat zestawiono w tabeli 2.3.

Przykładowa tabela może się zaczynać tak:

```
\begin{table}[!hbp]
```

Tabela 2.3: Opcjonalny argument otoczeń `table` i `figure`

Znak	Dopuszczalne miejsce umieszczenia wstawki
<code>h</code>	bez przemieszczenia, dokładnie w miejscu użycia
<code>t</code>	na górze strony
<code>b</code>	na dole strony
<code>p</code>	na stronie zawierającej wyłącznie wstawki
<code>!</code>	ignorując większość parametrów kontrolujących umieszczanie wstawek <sup>a</sup> , przekroczenie wartości, które mogą nie pozwolić na umieszczanie następnych wstawek na stronie.

<sup>a</sup>Są to parametry takie jak np. maksymalna dopuszczalna liczba wstawek na stronie

Argument `[!hbp]` oznacza, że tabelę można umieścić w miejscu, w którym pojawia się w pliku źródłowym (`h`), albo na dole strony (`b`), albo wreszcie na osobnej stronie zawierającej wyłącznie wstawki (`p`). Ponadto „`!`” oznacza, że  $\text{\LaTeX}$  ma pominąć większość parametrów sterujących umieszczaniem wstawek. Jeżeli otoczenia `table` użyto bez opcjonalnego argumentu, to jego domyślnymi wartościami są `[tbp]`.

$\text{\LaTeX}$  umieszcza każdą wstawkę zgodnie ze specyfikacją autora podaną w argumencie *miejsce*. Jeżeli nie może umieścić wstawki na bieżącej stronie, to dołącza ją albo do *kolejki rysunków*, albo do *kolejki tabel*<sup>12</sup>. Na początku składania nowej strony  $\text{\LaTeX}$  sprawdza, czy można ją zappełnić wstawkami czekającymi w kolejce. Jeśli nie jest to możliwe, to pierwsza wstawka każdej z kolejek traktowana jest tak, jak gdyby właśnie pojawiła się w tekście:  $\text{\LaTeX}$  stara się ją umieścić zgodnie z wartościami parametru *miejsce* (za wyjątkiem `h`, gdyż nie jest to już oczywiście możliwe). Nowe wstawki dołączane są na koniec odpowiednich kolejek.  $\text{\LaTeX}$  dba o właściwy porządek wstawek każdego typu. Może się zdarzyć, że pojedynczy rysunek, którego z jakichś względów nie można poprawnie wstawić, „ciągnie” za sobą wszystkie późniejsze rysunki, nawet aż na koniec dokumentu. Dlatego:

Jeżeli  $\text{\LaTeX}$  nie umieszcza wstawek zgodnie z oczekiwaniami, to z reguły któraś z nich blokuje całą kolejkę, a być może nawet wszystkie kolejki wstawek.

Wyjaśnwszy ów cokolwiek zawiły problem umieszczania wstawek, przejdźmy do omówienia kilku pozostałych spraw z nimi związanych. Poleceniem:

```
\caption{tekst}
```

wstawiamy tytuł rysunku lub tabeli. Kolejny numer rysunku bądź tabeli oraz słowo „Rysunek” bądź „Tabela” (lub „Tablica” – zależnie od używanego pakietu polonizacyjnego) zostaną wstawione automatycznie.

<sup>12</sup>Są to kolejki typu FIFO (pierwsze weszło – pierwsze wyjdzie).

Następujące instrukcje:

```
\listoffigures oraz \listoftables
```

działają analogicznie do instrukcji `\tableofcontents`, wstawiając do dokumentu, odpowiednio, spis rysunków oraz spis tabel. Poszczególnymi pozycjami tych spisów będą tytuły rysunków bądź tabel będące argumentami instrukcji `\caption`. Jeżeli tytuł jest długi, to do spisu można przesłać jego wersję skróconą, podaną jako opcjonalny argument instrukcji `\caption`:

```
\caption[Short]{LLLLLoooooonnnnnngggg}
```

Za pomocą instrukcji `\label` oraz `\ref` można tworzyć odsyłacze do tabel i rysunków.

Polecenie `\label` należy umieszczać *bezpośrednio* za instrukcją `\caption`. Dobrym pomysłem jest też umieszczenie jej wewnątrz argumentu instrukcji `\caption` (na przykład na końcu tytułu rysunku czy tabeli). Niektórzy użytkownicy błędnie sądzą, że wystarczy umieścić instrukcję `\label` wewnątrz otoczenia `figure` czy `table`, gdy tymczasem umieszczenie jej przed poleceniem `\caption` prowadzi do błędów w numerach odsyłaczy.

W poniższym przykładzie wstawka zawiera prostokąt o wymiarach 5 cm × 5 cm. Ten sposób postępowania można wykorzystać w celu zarezerwowania miejsca na rysunki, które zostaną wklejone później – do gotowego, wydrukowanego dokumentu.

Rysunek~\ref{white} jest przykładem Pop-Artu.

```
\begin{figure}[!htp]
\makebox[\textwidth]{\framebox[5cm]{\rule{0pt}{5cm}}}
\caption{Pięć na pięć centymetrów\label{white} }
\end{figure}
```

Zakładając w tym przykładzie, że kolejka rysunków jest pusta,  $\text{\LaTeX}$  najpierw spróbuje umieścić rysunek bez przesuwania go dokądkolwiek (**h**). Jeżeli okaże się to niemożliwe, to spróbuje go umieścić na górze strony (**t**). Jeżeli i to okaże się niewykonalne, to będzie się starał umieścić rysunek na stronie zawierającej wyłącznie wstawki (**p**). Jeżeli w kolejkach rysunków i tabel nie ma wstawek pozwalających wypełnić stronę, to  $\text{\LaTeX}$  rozpocznie nową stronę i spróbuje umieścić na niej rysunek, traktując go znowu tak, jakby właśnie pojawił się w tekście.

Czasami może wystąpić konieczność wykonania instrukcji:

```
\clearpage albo nawet \cleardoublepage
```

W wyniku jej zadziałania  $\text{\LaTeX}$  umieszcza w dokumencie wszystkie oczekujące w kolejkach wstawki, a następnie rozpoczyna skład od nowej strony. W wypadku użycia instrukcji `\cleardoublepage`  $\text{\LaTeX}$  rozpoczyna skład od strony nieparzystej (por. przypis 2 na str. 18).

W dalszej części książki przedstawimy, jak można do dokumentu dołączać rysunki w formacie Postscript (por. 4.1).

## 2.13. Ochrona poleceń kruchych

Niektórych poleceń nie można umieszczać wewnątrz argumentów innych poleceń, na przykład polecenie `\footnote` albo `\verb` nie może się pojawić w argumencie polecenia `\section` czy `\caption`. Kompilacja dokumentu zakończy się w takiej sytuacji błędem, a takie polecenia nazywamy *kruchymi* (ang. *fragile command*).

Polecenia kruche wymagają „ochrony”. Możemy je chronić, umieszczając przed nimi polecenie `\protect`. Polecenie `\protect` odnosi się wyłącznie do instrukcji znajdującej się tuż za nim, nie obejmuje swym działaniem nawet argumentów tej instrukcji. W większości wypadków nadmiarowe użycie `\protect` nie powoduje szkód.

```
\section{Jestem przezorny%
  \protect\footnote{i~chronię przypisy}}
```

Uwaga: wiele osób lubi dodawać przypisy do śródtytułów. Naszym zdaniem jest to zły i komplikujący życie zwyczaj; ostatecznie zawsze można umieścić przypis w pierwszym akapicie pod śródtytułem. Rób jak uważasz, ale czy wiesz, jak usunąć numer przypisu ze spisu treści?

## 2.14. Listy

Do pisania listów można użyć klasy `letter`. Struktura pliku źródłowego tej klasy różni się od dokumentów z klasy `article` czy `book`. Klasę `letter` zaprojektowano tak, by bezproblemowo dało się napisać zarówno pojedynczy list do przysłowiowej „cioci”, jak też setki listów do różnych osób (korespondencja seryjna).

Jeśli imię, nazwisko i adres nadawcy mają być identyczne we wszystkich listach, to deklarujemy je poleceniem:

```
\address{imię\\nazwisko\\ adres...}
```

Użycie instrukcji `\\` w adresie powoduje złamanie wiersza.

Polecenie `\signature` służy do zadeklarowania podpisu pod listem. Wewnątrz argumentu tego polecenia instrukcja `\\` służy do rozpoczęcia nowego wiersza, przykładowo:

Dyr. E.~K.~Tor,\\ Przewodniczący  
Zastępcy

Dyr. E. K. Tor, Przewodniczący Zastępcy
--

Do nagłówka listu automatycznie wstawiana jest bieżąca data. Aby wstawić inną, należy zastosować deklarację `\date`:

```
\date{16 Czerwca 1963~r.}
```

Deklaracje `\address`, `\signature` oraz `\date` umieszcza się zwykle w preambule, chociaż mogą one wystąpić również w części zasadniczej pliku źródłowego.

Treść listu powinniśmy wpisać wewnątrz otoczenia `letter`. Otoczenie to ma jeden argument, którym jest adres osoby, do której piszemy. Wewnątrz otoczenia `letter` można stosować kilka prostych poleceń służących do umieszczania w odpowiednim miejscu elementów typowego listu<sup>13</sup>. Do złożenia nagłówka listu używamy polecenia `\opening`, a do zakończenia – `\closing`. Ponadto są polecenia: `\ps` do wstawienia *post scriptum* oraz `\cc` do zdefiniowania wykazu osób, które mają otrzymać kopię listu. Oto pełny przykład listu:

```
%& --translate-file=il2-pl
\documentclass{letter}
\usepackage{polski}
\address{Dyr. E.~K.~Tor,\\ Przewodniczący Zastępcy\\
  Firma z~o.o.\\ w/m}
\signature{E.~K.~Thor}
\begin{document}
% pierwszy list
\begin{letter}{Henryk Potrykus\\ul.~Krótka\\Puck}
\opening{Szanowny Panie}
Z~przykrością zawiadamiam, że Pańskie podanie
zostało...
\closing{Z~poważaniem}
\cc{cc: Józef Wujke}
\end{letter}

% drugi list
\begin{letter}{Zofia Potrykus\\ul.~Szkolna\\Reda}
\opening{Szanowna Pani}
Odpowiadając na Pani pismo...
...
\end{letter}
\end{document}
```

---

<sup>13</sup>Elementy nietypowe zawsze można umieścić, korzystając z innych poleceń poznanych w tym rozdziale.



## Rozdział 3

# Wyrażenia matematyczne

Nareszcie! W tym rozdziale poznasz najlepszą stronę  $\text{\TeX}$ a, czyli skład wzorów matematycznych. Ostrzegamy jednak, że przedstawimy tu jedynie absolutne podstawy. Chociaż wystarczają one większości użytkowników, to nie załamuj rąk, jeśli nie poradzisz sobie z jakimś skomplikowanym wzorem, lecz zapoznaj się z możliwościami  $\text{\LaTeX}$ a lub innego wyspecjalizowanego pakietu<sup>1</sup>.

### 3.1. Wstęp

Do składu wyrażeń matematycznych mamy w  $\text{\LaTeX}$ u specjalny *tryb matematyczny*. Wzory wpisuje się między znakami  $\$$  i  $\$$ , między parami znaków  $\backslash ($  i  $\backslash )$  albo między  $\backslash \text{begin}\{\text{math}\}$  oraz  $\backslash \text{end}\{\text{math}\}$ .

$\$a\$$  do kwadratu plus  $\$b\$$   
do kwadratu równa się  $\$c\$$   
do kwadratu. Albo, stosując  
bardziej matematyczne  
podejście:  $\$c^{\{2\}}=a^{\{2\}}+b^{\{2\}}\$$ .

$\{\text{\TeX}\}$  należy wymawiać jako  
 $\$\tau\epsilon\chi\$$ .  $\backslash\backslash[6\text{pt}]$   
 $100\text{\~m}\$^{\{3\}}\$$  wody.  $\backslash\backslash[6\text{pt}]$   
To płynie z~mojego~ $\$\heartsuit\$$ .

$a$  do kwadratu plus  $b$  do kwadratu równa  
się  $c$  do kwadratu. Albo, stosując bardziej  
matematyczne podejście:  $c^2 = a^2 + b^2$ .

$\text{\TeX}$  należy wymawiać jako  $\tau\epsilon\chi$ .

$100\text{ m}^3$  wody.

To płynie z mojego  $\heartsuit$ .

Składając większe wzory, powinniśmy je eksponować, to znaczy wstawiać między akapitami, w osobnym wierszu. Takie wzory umieszcza się albo między parami znaków  $\backslash [$  i  $\backslash ]$ , albo wewnątrz otoczenia  $\text{displaymath}$ . Ta ostatnia konstrukcja dotyczy tworzenia wzorów bez numeracji:

<sup>1</sup>Pod egidą Amerykańskiego Towarzystwa Matematycznego (*American Mathematical Society*) powstało istotne rozszerzenie  $\text{\LaTeX}$ a. Wiele przykładów w tym rozdziale korzysta z tego rozszerzenia, które jest dołączane do wszystkich współczesnych dystrybucji  $\text{\TeX}$ a. Jeśli w twojej go nie ma, to znajdziesz je pod adresem [CTAN://macros/latex/packages/amslatex](http://ctan://macros/latex/packages/amslatex).

`$a$` do kwadratu plus `~$b$`  
do kwadratu równa się `~$c$`  
do kwadratu. Albo,  
bardziej matematycznie:  
`\begin{displaymath}`  
`c^{\{2\}}=a^{\{2\}}+b^{\{2\}}`  
`\end{displaymath}`  
Pierwszy wiersz po wzorze.

$a$  do kwadratu plus  $b$  do kwadratu równa się  $c$  do kwadratu. Albo, bardziej matematycznie:

$$c^2 = a^2 + b^2$$

Pierwszy wiersz po wzorze.

Do uzyskiwania wzorów numerowanych stosujemy otoczenie `equation`. Instrukcji `\label` możemy wówczas użyć do zapamiętania numeru wzoru, a polecenia `\ref` albo pochodzącego z pakietu `amsmath` `\eqref` – do przywołania w dokumencie tego numeru:

`\begin{equation}`  
`\epsilon > 0 \label{eq:eps}`  
`\end{equation}`  
Ze wzoru (`\ref{eq:eps}`)  
otrzymujemy `\ldots`

$$\epsilon > 0 \quad (3.1)$$

Ze wzoru (3.1) otrzymujemy ...

Zwróćmy uwagę na różnicę w wyglądzie wzorów złożonych wewnątrz akapitu i w wersji eksponowanej:

`$\lim_{n \to \infty}`  
`\sum_{k=1}^n \frac{1}{k^2}`  
`= \frac{\pi^2}{6}$`

$$\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

`\begin{displaymath}`  
`\lim_{n \to \infty}`  
`\sum_{k=1}^n \frac{1}{k^2}`  
`= \frac{\pi^2}{6}`  
`\end{displaymath}`

$$\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

Między *trybem matematycznym*  $\text{\LaTeX}$ a a *trybem tekstowym* istnieją znaczne różnice. Na przykład, w *trybie matematycznym*:

1.  $\text{\LaTeX}$  ignoruje prawie wszystkie odstępy oraz znaki końca linii; wszystkie odstępy we wzorach wynikają bądź z kontekstu, bądź z użycia specjalnych poleceń, takich jak: `\,` lub `\quad` (por. punkt 6.3.3, str. 110).
2. Puste linie są niedozwolone. Nie ma czegoś takiego, jak podział wzorów na akapity.
3. Litery we wzorach służą do oznaczania nazw zmiennych; zmienne składamy inaczej niż zwykły tekst. Jeżeli częścią wzoru ma być zwykły tekst, to należy się posłużyć instrukcją `\text{rm}\{...\}`.

`\begin{equation}`  
`\forall x \in \mathbf{R} \colon`  
`\quad x^2 \geq 0`  
`\end{equation}`

$$\forall x \in \mathbf{R}: \quad x^2 \geq 0 \quad (3.2)$$

```
\begin{equation}
x^2 \geq 0 \quad \quad
\text{dla każdego } x \in \mathbf{R}
\end{equation}
```

$$x^2 \geq 0 \quad \text{dla każdego } x \in \mathbf{R} \quad (3.3)$$

Matematycy potrafią być niezwykle staranni w doborze symboli. Na przykład we wzorach, w których występują oznaczenia zbiorów (jak powyższy), często stosuje się krój, w którym te oznaczenia przypominają odmianę „grubą”, pisaną kredą na tablicy ( $\mathbb{A}, \mathbb{B}, \mathbb{C} \dots$ ). Symbole takie wstawiamy do wzoru poleceniem `\mathbb{b}` z pakietu `amssymb` lub `amssymb`. Ostatni przykład wygląda wtedy następująco:

```
\begin{displaymath}
x^2 \geq 0 \quad \quad
\text{dla każdego } x \in \mathbb{R}
\end{displaymath}
```

$$x^2 \geq 0 \quad \text{dla każdego } x \in \mathbb{R}$$

## 3.2. Grupowanie

Argumentem większości instrukcji do składu matematyki jest tylko jeden znak – ten, który następuje tuż po instrukcji. Jeżeli polecenie ma dotyczyć grupy znaków, to należy je umieścić wewnątrz pary nawiasów klamrowych `{...}`:

```
\begin{equation}
a^{x+y} \neq a^{x+y}
\end{equation}
```

$$a^x + y \neq a^{x+y} \quad (3.4)$$

## 3.3. Części składowe wyrażeń matematycznych

W tym podrozdziale opiszemy ważniejsze instrukcje do składu wyrażeń. Zestawienie wszystkich dostępnych symboli i znaków znajduje się w punkcie 3.10 na stronie 58.

**Małe litery alfabetu greckiego** wprowadzamy, używając instrukcji typu: `\alpha`, `\beta`, `\gamma` itd., a **duże**<sup>2</sup>: `\Delta`, `\Gamma` itd.:

```
\lambda, \xi, \pi, \mu, \Phi, \Omega
```

$$\lambda, \xi, \pi, \mu, \Phi, \Omega$$

**Indeksy górne i wykładniki** otrzymujemy za pomocą znaku `^`, a **dolne** – stosując podkreślenie `_`:

<sup>2</sup>Obecnie brakuje dużej litery *Alpha* i wygląda ona identycznie jak pierwsza litera *A* alfabetu łacińskiego. Sytuacja ta ma się zmienić po wprowadzeniu nowego sposobu kodowania symboli matematycznych.

`$a_{1} x^{2} e^{-\alpha t}`  
`a^{3}_{ij} e^{x^2} \neq {e^x}^2$`

$$a_1 x^2 e^{-\alpha t} a_{ij}^3 e^{x^2} \neq e^{x^2}$$

**Pierwiastek kwadratowy** składamy poleceniem `\sqrt`. Wielkość znaku pierwiastka jest przez L<sup>A</sup>T<sub>E</sub>Xa ustalana automatycznie. Zapis samego znaku pierwiastka umożliwia instrukcja `\surd`<sup>3</sup>, natomiast pierwiastek stopnia  $n$  składamy konstrukcją `\sqrt[n]`:

`$\sqrt{x} \sqrt{x^2+\sqrt{y}}`  
`\sqrt[3]{2} \surd[x^2 + y^2]$`

$$\sqrt{x} \sqrt{x^2 + \sqrt{y}} \sqrt[3]{2} \sqrt{x^2 + y^2}$$

Polecenia `\overline` oraz `\underline` umieszczają nad i pod wyrażeniami poziome kreski:

`$\overline{m+n} \underline{x+y}$`

$$\overline{m+n} \underline{x+y}$$

Instrukcje `\overbrace` oraz `\underbrace` umieszczają nad i pod wyrażeniami poziome klamry:

`$\underbrace{a+b+\cdots+z}_{26}$`

$$\underbrace{a+b+\cdots+z}_{26}$$

Akcenty matematyczne, takie jak daszki czy tyldy nad zmiennymi, umieszczamy we wzorze poleceniami z tabeli 3.1. Szerokie daszki i tyldy, obejmujące wiele symboli, wstawiamy za pomocą instrukcji `\widetilde` oraz `\widehat`. Znakiem ' oznaczamy symbol „prim”:

`\begin{displaymath}`  
`\hat y=x^2\quad y'=2x''`  
`\end{displaymath}`

$$\hat{y} = x^2 \quad y' = 2x''$$

**Wektory** oznacza się niekiedy akcentem w postaci strzałki nad nazwą zmiennej. Służy do tego polecenie `\vec`. Natomiast do oznaczenia wektora od punktu  $A$  do punktu  $B$  korzystamy z poleceń `\overrightarrow` oraz `\overleftarrow`:

`\begin{displaymath}`  
`\vec a\quad\overrightarrow{AB}`  
`\end{displaymath}`

$$\vec{a} \quad \overrightarrow{AB}$$

Nazwy funkcji typu „logarytm” należy składać odmianą prostą, nie zaś kursywą, zarezerwowaną dla nazw zmiennych. Oto lista poleceń L<sup>A</sup>T<sub>E</sub>Xa służących do składu rozmaitych funkcji matematycznych:

<sup>3</sup>Taki zapis jest wykorzystywany raczej w literaturze anglosaskiej.

`\arccos \cos \csc \exp \ker \limsup \min \sinh`  
`\arcsin \cosh \deg \gcd \lg \ln \Pr \sup`  
`\arctan \cot \det \hom \lim \log \sec \tan`  
`\arg \coth \dim \inf \liminf \max \sin \tanh`

`\[\lim_{x \rightarrow 0}`  
`\frac{\sin x}{x}=1\]`

$$\lim_{x \rightarrow 0} \frac{\sin x}{x} = 1$$

W Polsce nazwy niektórych funkcji trygonometrycznych różnią się od anglosaskich. Pakiet `polski` – po dołączeniu do dokumentu – zmienia na życzenie standardowe funkcje `LaTeXa` na zgodne ze zwyczajami polskimi. Oto angielskie oryginały: `tan`, `coth`, `tanh`, `arccos`, `arcsin` i ich polskie odpowiedniki: `tg`, `ctgh`, `tgh`, `arc cos`, `arc sin`.

Dla funkcji typu modulo istnieją dwie instrukcje: `\bmod` dla binarnego operatora „ $a \bmod b$ ” oraz `\pmod` do składu takich wyrażeń jak „ $x \equiv a \pmod{b}$ ”.

`$a\bmod b$`  
`$x\equiv a \pmod{b}$`

$$a \bmod b$$

$$x \equiv a \pmod{b}$$

**Ułamki piętrowe** składa się poleceniem `\frac{...}{...}`. Do ułamków zwykłych czasami lepiej stosować kreskę ukośną, zwłaszcza w wypadku niewielkich porcji materiału ułamkowego, jak „ $1/2$ ”:

`$1\frac{1}{2}$~godziny`  
`\begin{displaymath}`  
`\frac{x^2}{k+1} \quad`  
`x^{\frac{2}{k+1}} \quad x^{1/2}`  
`\end{displaymath}`

$$1\frac{1}{2} \text{ godziny}$$

$$\frac{x^2}{k+1} \quad x^{\frac{2}{k+1}} \quad x^{1/2}$$

Do składu dwumianów lub podobnych konstrukcji możemy użyć polecenia `{... \choose ...}` albo `{... \atop ...}`. Instrukcja `\atop` daje efekt taki jak `\choose`, tyle że bez nawiasów:

`\begin{displaymath}`  
`{n \choose k} \quad x \atop y+2`  
`\end{displaymath}`

$$\binom{n}{k} \quad x \atop y+2$$

W pakiecie `amsmath` do składu dwumianu Newtona dostępne jest polecenie `\binom`:

`\begin{displaymath}`  
`\binom{n}{k} \quad \mathrm{C}_n^k`  
`\end{displaymath}`

$$\binom{n}{k} \quad \mathrm{C}_n^k$$

Do uzyskiwania symboli relacji binarnych może się przydać instrukcja `\stackrel{!}{=}`. Składa ona swój pierwszy argument czcionką pomniejszoną, jaka stosowana jest do indeksów, i umieszcza go nad drugim argumentem, złożonym czcionką normalnej wielkości:

```
\begin{displaymath}
\int f_N(x) \stackrel{!}{=} 1
\end{displaymath}
```

$$\int f_N(x) \stackrel{!}{=} 1$$

**Znak całki** składamy poleceniem `\int`, **znak sumowania** instrukcją `\sum`, zaś **operator iloczynu** za pomocą instrukcji `\prod`. Górne granice całkowania i sumowania określamy za pomocą `^`, a dolne – znakiem `_`, czyli podobnie jak w wypadku indeksów górnych i dolnych<sup>4</sup>:

```
\begin{displaymath}
\sum_{i=1}^n \int_0^{\frac{\pi}{2}} \prod_{\epsilon}
\end{displaymath}
```

$$\sum_{i=1}^n \int_0^{\frac{\pi}{2}} \prod_{\epsilon}$$

Pakiet `amsmath` zawiera dwa dodatkowe narzędzia do sterowania położeniem indeksów w złożonych wyrażeniach: instrukcję `\substack` i otoczenie `subarray`:

```
\begin{displaymath}
\sum_{\substack{0 < i < n \\ 1 < j < m}} P(i,j) =
\sum_{\begin{subarray}{l} i \in I \\ 1 < j < m \end{subarray}} Q(i,j)
\end{displaymath}
```

$$\sum_{\substack{0 < i < n \\ 1 < j < m}} P(i,j) = \sum_{\substack{i \in I \\ 1 < j < m}} Q(i,j)$$

Do składu **nawiasów** i innych **ograniczników** typu `( [ < || ↓` mamy różnorodność symboli. Nawiasy okrągłe i kwadratowe wstawiamy bezpośrednio z klawiatury. Do nawiasów klamrowych stosujemy `\{` oraz `\}`. Wszystkie inne ograniczniki wstawiamy, używając specjalnych poleceń, np. `\updownarrow`. Zestawienie dostępnych ograniczników znajduje się w tabeli 3.7 na stronie 61.

```
\begin{displaymath}
a, b, c \neq \{a, b, c\}
\end{displaymath}
```

$$a, b, c \neq \{a, b, c\}$$

Poprzedzenie otwierającego ogranicznika poleceniem `\left`, a zamykającego poleceniem `\right` powoduje automatyczne ustalenie jego rozmiaru w zależności od wielkości zawartego między nimi wyrażenia. Uwaga: każde użycie

<sup>4</sup> `\mathcal{S-LAT}_{\text{E}}\text{X}` ma dodatkowo wielolinijkowe indeksy dolne i górne.

`\left` oraz ogranicznika wymaga nawiasu zamykającego poprzedzonego poleceniem `\right`. Gdy ogranicznik ma się pojawić tylko po jednej stronie, wówczas po drugiej *należy* użyć konstrukcji z kropką: `\left.` po lewej albo `\right.` po prawej:

```
\begin{displaymath}
1 + \left( \frac{1}{1-x^2} \right)
\right) ^3
\end{displaymath}
```

$$1 + \left( \frac{1}{1-x^2} \right)^3$$

W pewnych sytuacjach trzeba samemu określić właściwą wielkość ogranicznika. Do tego celu służą instrukcje `\big`, `\Big`, `\bigg` oraz `\Bigg`, poprzedzające odpowiedni ogranicznik<sup>5</sup>:

```
$\Big( (x+1) (x-1) \Big) ^2$\\
$\big(\Big(\bigg(\Bigg(\quad
$\big\}\Big\}\bigg\}\Bigg\}\quad
$\big\|\Big\|\bigg\|\Bigg\|$\
```

$$\left( (x+1)(x-1) \right)^2$$

$$\left( \left( \left( \left\{ \right\} \right) \right) \right) \left\| \left\| \left\| \left\| \right\| \right\| \right\|$$

**Wielokropek** w wyrażeniach matematycznych wprowadzamy poleceniem `\ldots`. Kropki pojawiają się wtedy na linii podstawowej, to znaczy na jednakowej wysokości z przecinkiem czy kropką. Instrukcja `\cdots` wstawia natomiast inny rodzaj wielokropka, w którym kropki znajdują się w osi znaków +, -, =:

```
\begin{displaymath}
x_{\{1\}}, \ldots, x_{\{n\}} \quad \qquad \qquad
x_{\{1\}} + \cdots + x_{\{n\}}
\end{displaymath}
```

$$x_1, \dots, x_n \qquad x_1 + \cdots + x_n$$

Są jeszcze instrukcje `\vdots` oraz `\ddots`. Pierwsza z nich generuje wielokropek pionowy, a druga – skośny (zobacz przykład w punkcie 3.5).

### 3.4. Odstępy w trybie matematycznym

Zdarzają się sytuacje, kiedy wielkość odstępów wewnątrz wyrażenia matematycznego jest niepoprawna. Można je skorygować odpowiednimi instrukcjami. Do wprowadzania niewielkich odstępów służy kilka poleceń: `\`, wstawia odstęp równy  $\frac{3}{18}$  em<sup>6</sup> (`\`), `\:` pozwala uzyskać odstęp równy  $\frac{4}{18}$  em (`\:`) a `\;` – odstęp równy  $\frac{5}{18}$  em (`\;`). Użycie instrukcji `\_` (spacja po znaku `\`) prowadzi

<sup>5</sup>Polecenia te nie działają zgodnie z oczekiwaniami, gdy uprzednio zmieniono stopień pisma, na przykład użyto opcji `11pt` lub `12pt`. W takiej sytuacji należy skorzystać z pakietu `exscale` albo pakietu `amsmath`.

<sup>6</sup>W programach komputerowych przyjęło się stosowanie jednostki *em* równej szerokości litery „M” w bieżącym kroju pisma. Por. też [20].

do utworzenia zwykłego odstępu międzywyrazowego; `\quad` – odstępu równego 1 em ( $\square$ ), a `\qquad` – dwóm em ( $\square\square$ ). Instrukcja `\!` wstawia odstęp „ujemny”, to znaczy zamiast zwiększać, zmniejsza odstęp między znakami. Wielkość tego odstępu wynosi  $-\frac{3}{18}$  em ( $\mathbb{U}$ ):

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\int\!\!\!\!\!\int\int_D g(x,y)
\, \, \, \ud x \, \, \, \ud y
\end{displaymath}
%
```

zamiast

```
\begin{displaymath}
\int\int_D g(x,y)\ud x \ud y
\end{displaymath}
```

zamiast

$$\iint_D g(x,y) \, dx \, dy$$

$$\int \int_D g(x,y) dx dy$$

Zwróćmy uwagę, że litera „d” w symbolu różniczki jest złożoną odmianą prostą pisma<sup>7</sup>.

Dzięki zdefiniowanym w pakiecie  $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\mathcal{L}\mathcal{A}\mathcal{T}\mathcal{E}\mathcal{X}$ <sup>8</sup> takim instrukcjom jak: `\iint`, `\iiint`, `\iiint` oraz `\idotsint` powyższy przykład można złożyć dużo prościej:

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\iint_D \, \, \, \ud x \, \, \, \ud y
\end{displaymath}
```

$$\iint_D dx \, dy$$

Więcej wiadomości na ten temat znajdziemy w pliku `testmath.tex`, który jest częścią pakietu  $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\mathcal{L}\mathcal{A}\mathcal{T}\mathcal{E}\mathcal{X}$  lub w rozdziale ósmym podręcznika [6].

### 3.5. Wyrównywanie w pionie

Do składania macierzy używa się otoczenia `array`. Działa ono podobnie do wcześniej omówionego otoczenia `tabular`. Używane w przykładzie polecenie `\\` oznacza przejście do nowego wiersza macierzy:

```
\begin{displaymath}
\mathbf{X} =
\left( \begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)
\end{displaymath}
```

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \dots \\ x_{21} & x_{22} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

<sup>7</sup>W Polsce do składu litery „d” w różniczkach stosuje się kursywę matematyczną.

<sup>8</sup>Ścisłej mówiąc, w pakiecie `amsmath`.



Otoczeniem `array` możemy się posłużyć także do składania wyrażeń zawierających tylko jeden ogranicznik, po prawej lub po lewej stronie, stosując konstrukcję z kropką `\right.` lub `\left.`:

```
\begin{displaymath}
y = \left\{ \begin{array}{l}
a & \text{\texttt{\textit{jeżeli } $d > c$}} \\
b+x & \text{\texttt{\textit{rano}}} \\
l & \text{\texttt{\textit{w ciągu dnia}}}
\end{array} \right.
\end{displaymath}
```

$$y = \begin{cases} a & \text{jeżeli } d > c \\ b+x & \text{rano} \\ l & \text{w ciągu dnia} \end{cases}$$

Podobnie jak w wypadku otoczenia `tabular`, także w otoczeniu `array` można wstawiać pionowe i poziome kreski, np. oddzielające poszczególne rubryki macierzy:

```
\begin{displaymath}
\left( \begin{array}{c|c}
1 & 2 \\ \hline
3 & 4
\end{array} \right)
\end{displaymath}
```

$$\left( \begin{array}{c|c} 1 & 2 \\ \hline 3 & 4 \end{array} \right)$$

Do składu wyrażeń wielowierszowych można użyć – zamiast otoczenia `equation` – otoczeń `eqnarray` lub `eqnarray*`. W otoczeniu `eqnarray` każdy wiersz zawartego w nim wyrażenia otrzymuje osobny numer; w otoczeniu `eqnarray*` wiersze nie są numerowane. Otoczenia `eqnarray` oraz `eqnarray*` działają jak trójkolumnowa tabela w układzie `{rcl}`. W takiej tabeli w środkowej kolumnie wstawiamy zwykle znaki równości lub nierówności. Poleceniem `\\` łamiemy zawartość otoczenia na wiersze:

```
\begin{eqnarray}
f(x) & = & \cos x \\
f'(x) & = & -\sin x \\
\int_0^x f(y)dy & = & \sin x
\end{eqnarray}
```

$$\begin{aligned} f(x) &= \cos x & (3.5) \\ f'(x) &= -\sin x & (3.6) \\ \int_0^x f(y)dy &= \sin x & (3.7) \end{aligned}$$

Zwróćmy uwagę, że odstęp po obu stronach znaku równości jest dość duży. Można go zmniejszyć poleceniem `\setlength\arraycolsep{2pt}`, które najlepiej jest umieścić w preambule dokumentu.

L<sup>A</sup>T<sub>E</sub>X nie dzieli automatycznie długich wyrażeń, niemieszczących się w jednym wierszu. Musimy to zrobić sami. Najczęściej stosuje się takie sposoby:

```
\setlength\arraycolsep{2pt}
\begin{eqnarray}
\sin x &= & x - \frac{x^3}{3!} + \frac{x^5}{5!} - \\
&+ \frac{x^7}{7!} - \cdots
\end{eqnarray}
```

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots \quad (3.8)$$

```
\begin{eqnarray}
\lefteqn{\cos x = 1 - \frac{x^2}{2!} +} \\
&+ \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots
\end{eqnarray}
```

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots \quad (3.9)$$

$\text{\LaTeX}$  nie generuje numeru w tym wierszu wyrażenia, w którym pojawia się polecenie `\nonumber`.

Złożenie w ten sposób dużych i skomplikowanych wyrażeń może się jednak okazać dość trudne. Lepiej wtedy użyć pakietu `amsmath`, w którym mamy do dyspozycji otoczenia `align`, `flalign`, `gather`, `multline` i `split`.

### 3.6. Fantomy

Fantomów nie można zobaczyć, mimo to zajmują one w naszych umysłach trochę miejsca. Nie inaczej jest w  $\text{\LaTeX}$ u, co pozwala robić z odstępami różne sztuczki.

Podczas wyrównywania w pionie tekstu z indeksami  $\sim$  bądź  $\_$   $\text{\LaTeX}$  bywa nadgorliwy. Polecenie `\phantom` pozwala rezerwować miejsce na znaki, które nie mają się pojawić w ostatecznym wydruku. Najłatwiej to zrozumieć, analizując przykład:

```
\begin{displaymath}
{}^{12}_{\phantom{1}}\text{C} \quad \text{versus} \quad {}^{12}_6\text{C}
\end{displaymath}
```

$${}^{12}_{\phantom{1}}\text{C} \quad \text{versus} \quad {}^{12}_6\text{C}$$

```
\begin{displaymath}
\Gamma_{ij}^{\phantom{k}} \quad \text{versus} \quad \Gamma_{ij}^k
\end{displaymath}
```

$$\Gamma_{ij}^{\phantom{k}} \quad \text{versus} \quad \Gamma_{ij}^k$$

### 3.7. Stopień pisma

W trybie matematycznym stopień pisma dobierany jest automatycznie, zależnie od kontekstu. Indeksy górne L<sup>A</sup>T<sub>E</sub>X składa na przykład mniejszą czcionką. Gdy wewnątrz wyrażenia matematycznego zachodzi potrzeba złożenia fragmentu normalnego tekstu, a użyjemy polecenia `\textrm`, to nie zadziała mechanizm przełączania stopni pisma. Będzie tak, ponieważ polecenie `\textrm` powoduje tymczasowe przejście do trybu tekstowego.

Zamiast `\textrm` można użyć polecenia `\mathrm`, które zachowuje mechanizm zmiany stopnia pisma. Pamiętajmy jednak, że działa ono poprawnie w zasadzie tylko dla pojedynczych wyrazów, ponieważ znaki odstępów są ignorowane. Ponadto nie działa zgodnie z oczekiwaniami mechanizm akcentowania<sup>9</sup>:

```
\begin{equation}
2^{\textrm{nd rd}}^{\textrm{th}} \quad \quad
2^{\mathrm{nd rd}}^{\mathrm{th}}
\end{equation}
```

$$2^{\text{nd rd}}^{\text{th}} \quad 2^{\text{ndrd}}^{\text{th}} \quad (3.10)$$

Czasami musimy wyraźnie określić stopień pisma, jakim chcemy się posłużyć. W trybie matematycznym możemy do tego stosować cztery następujące polecenia:

`\displaystyle` (123), `\textstyle` (123), `\scriptstyle` (123) oraz `\scriptscriptstyle` (123).

Zmiana stylu dotyczy także sposobu składania indeksów górnych i dolnych, jak granice sumowania czy całkowania:

```
\begin{displaymath}
\mathop{\mathrm{cov}}(X,Y)=
\frac{\displaystyle
\sum_{i=1}^n(x_i-\overline{x})
(y_i-\overline{y})}
{\displaystyle\biggl[
\sum_{i=1}^n(x_i-\overline{x})^2
\sum_{i=1}^n(y_i-\overline{y})^2
\biggr]^{1/2}}
\end{displaymath}
```

$$\text{cov}(X, Y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\left[ \sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2 \right]^{1/2}}$$

Powyższy przykład ilustruje sytuację, w której należy zastosować polecenie `\biggl` bądź `\biggr`, ponieważ nawiasy pochodzące z konstrukcji `\left[` oraz `\right]` byłyby zbyt małe.

<sup>9</sup>W pakiecie  $\mathcal{A}\mathcal{M}\mathcal{S}$ -L<sup>A</sup>T<sub>E</sub>X polecenie `\textrm` zostało poprawione i umożliwia automatyczną zmianę stopnia pisma zależnie od kontekstu. Pakiet ten nie tylko poprawia działanie `\textrm`, ale też definiuje instrukcję `\text` do wstawiania fragmentów „normalnego” tekstu wewnątrz wzorów.

### 3.8. Twierdzenia, definicje, itp.

W pracach matematycznych występuje potrzeba wyróżniania w składzie zapisu lematów, definicji, aksjomatów i tym podobnych elementów. Do zdefiniowania nowego typu elementu służy polecenie:

```
\newtheorem{nazwa}[nazwa']{tekst}[punkt]
```

Argument *nazwa* oznacza nazwę otoczenia, zaś *tekst* jest napisem, który zostanie wydrukowany; może to być „Twierdzenie”, „Definicja” itp. Argumenty w nawiasach kwadratowych są nieobowiązkowe. Za ich pomocą określamy sposób numerowania twierdzeń. Opcjonalny argument *nazwa'* to nazwa elementu uprzednio zdefiniowanego poleceniem `\newtheorem`. Jeśli ten argument podano, to otoczenia *nazwa* oraz *nazwa'* będą posiadały wspólną numerację. Argument *punkt* określa sposób numerowania twierdzeń: jeżeli umieścimy tam na przykład `chapter`, to elementy będą numerowane w obrębie rozdziałów. Domyślnie otoczenia definiowane za pomocą `\newtheorem` są numerowane w sposób ciągły w obrębie całego dokumentu.

Po umieszczeniu instrukcji `\newtheorem{nazwa}...` w preambule można otoczenie *nazwa* stosować w następujący sposób:

```
\begin{nazwa}[tekst]
```

Oto moje interesujące twierdzenie

```
\end{nazwa}
```

Instrukcja `\newtheoremstyle{style}` z pakietu `amsthm` pozwala określić sposób formatowania twierdzenia przez wybór spośród trzech predefiniowanych stylów: **definition** (wytluszczony tytuł, treść złożona pismem prostym), **plain** (wytluszczony tytuł, treść zapisana kursywą) oraz **remark** (tytuł zapisany kursywą, treść – pismem prostym).

Tyle teoria. Poniższe przykłady usuną, miejmy nadzieję, wszelkie wątpliwości i jednocześnie uświadomią, że działanie otoczenia `\newtheorem` niełatwo zrozumieć:

```
% definicje w~preamble
\newtheorem{twr}{Twierdzenie}
\newtheorem{lem}{Lemat}
% po \begin{document}
\begin{lem} Pierwszy
lemat\dots\label{lem:1} \end{lem}
\begin{twr}{Dyzma}
Przyjmując w~lemacie~\ref{lem:1},
że  $\epsilon=0$ \dots \end{twr}
\begin{lem}Trzeci lemat\end{lem}
```

**Lemat 1.** *Pierwszy lemat...*

**Twierdzenie 2 (Dyzma).** *Przyjmując w lemacie 1, że  $\epsilon = 0$ ...*

**Lemat 3.** *Trzeci lemat*

Elementy Twierdzenie i Lemat używają tego samego licznika. Argument nieobowiązkowy (wewnątrz nawiasów kwadratowych) służy do umieszczenia komentarza, w postaci nazwiska twórcy itp.

```
\newtheorem{mur}{Murphy}[section]
\begin{mur} Jeżeli coś można
wykonać na dwa lub więcej sposobów,
przy czym jeden z nich prowadzi do
katastrofy, to sposób ten zostanie
przez kogoś wybrany.\end{mur}
```

**Murphy 3.8.1.** *Jeżeli coś można wykonać na dwa lub więcej sposobów, przy czym jeden z nich prowadzi do katastrofy, to sposób ten zostanie przez kogoś wybrany.*

Numeracja twierdzenia „Murphy’ego” jest tu powiązana z numeracją kolejnych punktów. Można też do numerowania twierdzeń stosować inne jednostki podziału dokumentu, jak rozdziały czy podpunkty.

W pakiecie `amsthm` znajduje się też otoczenie `proof` do zapisywania dowodów:

```
\begin{proof}
Banalne. Użyj \[E=mc^2\]
\end{proof}
```

*Dowód.* Banalne. Użyj  
$$E = mc^2$$

□

Polecenie `\qedhere` pozwala wstawić symbol „końca dowodu” w określonym miejscu zamiast domyślnego umieszczania go w oddzielnym wierszu:

```
\begin{proof}
Banalne. Użyj \[E=mc^2 \qedhere\]
\end{proof}
```

*Dowód.* Banalne. Użyj  
$$E = mc^2$$

□

### 3.9. Symbole półgrube

Wstawianie symboli półgrubych jest w  $\text{\LaTeX}$ u zadaniem dość trudnym. Być może jest tak celowo, ponieważ składacze-amatorzy mają skłonność do ich nadużywania. Poleceniem `\mathbf` uzyskamy odmianę półgrubą. Nie będzie to niestety półgruba kursywa, jaką zwykle składane są symbole matematyczne. Istnieje co prawda polecenie `\boldmath`, ale można go użyć jedynie *poza trybem matematycznym*. Jego działanie obejmuje również symbole.

```
\begin{displaymath}
\mu, M \quad \mathbf{M} \quad \mu, M
\end{displaymath}
```

$\mu, M$      **M**      $\mu, M$

Zauważmy, że przecinek też został złożony w odmianie półgrubej, co z reguły jest niepożądanym efektem.

Pakiet `amsbsy`, dołączany przez `amsmath`, czyni zadanie dużo łatwiejszym. W pakiecie tym dostępne są polecenia `\boldsymbol` oraz `\pmb`. Instrukcja `\pmb` imituje znak półgruby przez wydrukowanie dwóch nakładających się na siebie znaków, złożonych w odmianie normalnej. Można tym

sposobem uzyskać symbole półgrube nawet wtedy, gdy w systemie brak odpowiednich fontów.

```
\begin{displaymath} \mu, M \quad \quad
\boldsymbol{\mu}, \boldsymbol{M} \quad \quad
\pmb{\mu}, \pmb{M} \end{displaymath}
```

$$\mu, M \quad \boldsymbol{\mu}, \boldsymbol{M} \quad \pmb{\mu}, \pmb{M}$$

### 3.10. Zestawienie symboli matematycznych

W poniższych tabelach zestawiono wszystkie symbole standardowo dostępne w *trybie matematycznym*. Symbole w tabelach 3.11–3.15<sup>10</sup> są dostępne, jeżeli mamy zainstalowane dodatkowe fonty matematyczne (AMS *math fonts*) i do dokumentu dołączymy pakiet `amssymb`. W razie braku fontów lub pakietu można je odnaleźć w [CTAN://macros/latex/required/amslatex](http://CTAN://macros/latex/required/amslatex). Ponadto dużo bardziej kompletne zestawienie symboli matematycznych można znaleźć w [CTAN://info/symbols/comprehensive](http://CTAN://info/symbols/comprehensive).

Tabela 3.1: Akcenty matematyczne

$\hat{a}$ <code>\hat{a}</code>	$\check{a}$ <code>\check{a}</code>	$\tilde{a}$ <code>\tilde{a}</code>	$\acute{a}$ <code>\acute{a}</code>
$\grave{a}$ <code>\grave{a}</code>	$\dot{a}$ <code>\dot{a}</code>	$\ddot{a}$ <code>\ddot{a}</code>	$\breve{a}$ <code>\breve{a}</code>
$\bar{a}$ <code>\bar{a}</code>	$\vec{a}$ <code>\vec{a}</code>	$\widehat{A}$ <code>\widehat{A}</code>	$\widetilde{A}$ <code>\widetilde{A}</code>

Tabela 3.2: Litery alfabetu greckiego

$\alpha$ <code>\alpha</code>	$\theta$ <code>\theta</code>	$o$ <code>o</code>	$\upsilon$ <code>\upsilon</code>
$\beta$ <code>\beta</code>	$\vartheta$ <code>\vartheta</code>	$\pi$ <code>\pi</code>	$\phi$ <code>\phi</code>
$\gamma$ <code>\gamma</code>	$\iota$ <code>\iota</code>	$\varpi$ <code>\varpi</code>	$\varphi$ <code>\varphi</code>
$\delta$ <code>\delta</code>	$\kappa$ <code>\kappa</code>	$\rho$ <code>\rho</code>	$\chi$ <code>\chi</code>
$\epsilon$ <code>\epsilon</code>	$\lambda$ <code>\lambda</code>	$\varrho$ <code>\varrho</code>	$\psi$ <code>\psi</code>
$\varepsilon$ <code>\varepsilon</code>	$\mu$ <code>\mu</code>	$\sigma$ <code>\sigma</code>	$\omega$ <code>\omega</code>
$\zeta$ <code>\zeta</code>	$\nu$ <code>\nu</code>	$\varsigma$ <code>\varsigma</code>	
$\eta$ <code>\eta</code>	$\xi$ <code>\xi</code>	$\tau$ <code>\tau</code>	
$\Gamma$ <code>\Gamma</code>	$\Lambda$ <code>\Lambda</code>	$\Sigma$ <code>\Sigma</code>	$\Psi$ <code>\Psi</code>
$\Delta$ <code>\Delta</code>	$\Xi$ <code>\Xi</code>	$\Upsilon$ <code>\Upsilon</code>	$\Omega$ <code>\Omega</code>
$\Theta$ <code>\Theta</code>	$\Pi$ <code>\Pi</code>	$\Phi$ <code>\Phi</code>	

W trybie matematycznym L<sup>A</sup>T<sub>E</sub>X wstawia dodatkowy mały odstęp po przecinku i średniku, natomiast w wypadku dwukropka wstawia odstęp

<sup>10</sup>Tabele przygotowano na podstawie pliku `symbols.tex` (David Carlisle), gruntownie zmodyfikowanego zgodnie z sugestiami Josefa Tkadleca.

Tabela 3.3: Symbole relacji

Odpowiednie symbole negacji można utworzyć, poprzedzając każde z poniższych poleceń instrukcją `\not`.

$<$	$<$	$>$	$>$	$=$	$=$
$\leq$	<code>\leq</code> lub <code>\le</code>	$\geq$	<code>\geq</code> lub <code>\ge</code>	$\equiv$	<code>\equiv</code>
$\ll$	<code>\ll</code>	$\gg$	<code>\gg</code>	$\doteq$	<code>\doteq</code>
$\prec$	<code>\prec</code>	$\succ$	<code>\succ</code>	$\sim$	<code>\sim</code>
$\preceq$	<code>\preceq</code>	$\succeq$	<code>\succeq</code>	$\simeq$	<code>\simeq</code>
$\subset$	<code>\subset</code>	$\supset$	<code>\supset</code>	$\approx$	<code>\approx</code>
$\subseteq$	<code>\subseteq</code>	$\supseteq$	<code>\supseteq</code>	$\cong$	<code>\cong</code>
$\sqsubset$	<code>\sqsubset</code> <sup>a</sup>	$\sqsupset$	<code>\sqsupset</code> <sup>a</sup>	$\bowtie$	<code>\Join</code> <sup>a</sup>
$\sqsubseteq$	<code>\sqsubseteq</code>	$\sqsupseteq$	<code>\sqsupseteq</code>	$\bowtie$	<code>\bowtie</code>
$\in$	<code>\in</code>	$\ni$	<code>\ni</code> , <code>\owns</code>	$\propto$	<code>\propto</code>
$\vdash$	<code>\vdash</code>	$\dashv$	<code>\dashv</code>	$\models$	<code>\models</code>
$ $	<code>\mid</code>	$\parallel$	<code>\parallel</code>	$\perp$	<code>\perp</code>
$\smile$	<code>\smile</code>	$\frown$	<code>\frown</code>	$\asymp$	<code>\asymp</code>
$:$	<code>:</code>	$\notin$	<code>\notin</code>	$\neq$	<code>\neq</code> lub <code>\ne</code>

<sup>a</sup> Dostępne po dołączeniu pakietu `latexsym`.

przed i za znakiem, bo traktuje dwukropek jako znak relacji. Oto ilustracja tej różnicy:

Nie `$f:A\to B$`,  
lecz `$f\colon A\to B$`

Nie  $f : A \rightarrow B$ , lecz  $f : A \rightarrow B$

Jeżeli przecinek oddziela część całkowitą liczby od części dziesiętnej, to wskazane jest zakazać  $\text{\LaTeX}$ -owi wstawiania dodatkowego odstępu, co zwykle robi w trybie matematycznym. Wystarczy w tym celu otoczyć przecinek parą nawiasów klamrowych. Porównajmy:

Zamiast `$22,115$` lepiej `$22\{, \}115$`

Zamiast 22,115 lepiej 22,115

Znaki mniejszy-lub-równy i większy-lub-równy mają kształt różny od stosowanego w krajach anglosaskich. Po dołączeniu pakietu `polski` standardowe w  $\text{\LaTeX}$ u polecenia `\leq` oraz `\geq` generują polskie wersje tych relacji, to znaczy  $\leq$  i  $\geq$  zamiast  $\leq$  i  $\geq$ .

Tabela 3.4: Symbole operacji dwuargumentowych

$+$	$+$	$-$	$-$	$\triangleleft$	<code>\triangleleft</code>
$\pm$	<code>\pm</code>	$\mp$	<code>\mp</code>	$\triangleright$	<code>\triangleright</code>
$\cdot$	<code>\cdot</code>	$\div$	<code>\div</code>	$\star$	<code>\star</code>
$\times$	<code>\times</code>	$\setminus$	<code>\setminus</code>	$\ast$	<code>\ast</code>
$\cup$	<code>\cup</code>	$\cap$	<code>\cap</code>	$\circ$	<code>\circ</code>
$\sqcup$	<code>\sqcup</code>	$\sqcap$	<code>\sqcap</code>	$\bullet$	<code>\bullet</code>
$\vee$	<code>\vee</code> , <code>\lor</code>	$\wedge$	<code>\wedge</code> , <code>\land</code>	$\diamond$	<code>\diamond</code>
$\oplus$	<code>\oplus</code>	$\ominus$	<code>\ominus</code>	$\uplus$	<code>\uplus</code>
$\odot$	<code>\odot</code>	$\oslash$	<code>\oslash</code>	$\amalg$	<code>\amalg</code>
$\otimes$	<code>\otimes</code>	$\bigcirc$	<code>\bigcirc</code>	$\dagger$	<code>\dagger</code>
$\triangle$	<code>\bigtriangleup</code>	$\nabla$	<code>\bigtriangledown</code>	$\ddagger$	<code>\ddagger</code>
$\triangleleft$	<code>\lhd<sup>a</sup></code>	$\triangleright$	<code>\rhd<sup>a</sup></code>	$\wr$	<code>\wr</code>
$\trianglelefteq$	<code>\unlhd<sup>a</sup></code>	$\trianglerighteq$	<code>\unrhd<sup>a</sup></code>		

Tabela 3.5: Symbole zmiennej wielkości

$\sum$	<code>\sum</code>	$\bigcup$	<code>\bigcup</code>	$\bigvee$	<code>\bigvee</code>	$\bigoplus$	<code>\bigoplus</code>
$\prod$	<code>\prod</code>	$\bigcap$	<code>\bigcap</code>	$\bigwedge$	<code>\bigwedge</code>	$\bigotimes$	<code>\bigotimes</code>
$\coprod$	<code>\coprod</code>	$\bigsqcup$	<code>\bigsqcup</code>			$\bigodot$	<code>\bigodot</code>
$\int$	<code>\int</code>	$\oint$	<code>\oint</code>			$\biguplus$	<code>\biguplus</code>

Tabela 3.6: Strzałki

$\leftarrow$	<code>\leftarrow</code> lub <code>\gets</code>	$\longleftarrow$	<code>\longleftarrow</code>	$\uparrow$	<code>\uparrow</code>
$\rightarrow$	<code>\rightarrow</code> lub <code>\to</code>	$\longrightarrow$	<code>\longrightarrow</code>	$\downarrow$	<code>\downarrow</code>
$\leftrightarrow$	<code>\leftrightarrow</code>	$\longleftrightarrow$	<code>\longleftrightarrow</code>	$\updownarrow$	<code>\updownarrow</code>
$\Leftarrow$	<code>\Leftarrow</code>	$\Longleftarrow$	<code>\Longleftarrow</code>	$\Uparrow$	<code>\Uparrow</code>
$\Rightarrow$	<code>\Rightarrow</code>	$\Longrightarrow$	<code>\Longrightarrow</code>	$\Downarrow$	<code>\Downarrow</code>
$\Leftrightarrow$	<code>\Leftrightarrow</code>	$\Longleftrightarrow$	<code>\Longleftrightarrow</code>	$\Updownarrow$	<code>\Updownarrow</code>
$\mapsto$	<code>\mapsto</code>	$\longmapsto$	<code>\longmapsto</code>	$\nearrow$	<code>\nearrow</code>
$\hookrightarrow$	<code>\hookrightarrow</code>	$\hookrightarrow$	<code>\hookrightarrow</code>	$\searrow$	<code>\searrow</code>
$\leftharpoonup$	<code>\leftharpoonup</code>	$\rightharpoonup$	<code>\rightharpoonup</code>	$\swarrow$	<code>\swarrow</code>
$\leftharpoondown$	<code>\leftharpoondown</code>	$\rightharpoondown$	<code>\rightharpoondown</code>	$\nwarrow$	<code>\nwarrow</code>
$\rightleftharpoons$	<code>\rightleftharpoons</code>	$\iff$	<code>\iff</code>	$\leadsto$	<code>\leadsto<sup>a</sup></code>

<sup>a</sup> Dostępne po dołączeniu pakietu `latexsym`.



Tabela 3.7: Ograniczniki

$($	$($	$)$	$\uparrow$	$\uparrow$	$\Uparrow$	$\Uparrow$
$[$	$\lbrack$	$\rbrack$	$\rangle$	$\rangle$	$\Downarrow$	$\Downarrow$
$\{$	$\{$	$\}$	$\langle$	$\langle$	$\Updownarrow$	$\Updownarrow$
$\}$	$\}$	$\}$	$\rangle$	$\rangle$	$\lvert$	$\lvert$
$\}$	$\}$	$\}$	$\lfloor$	$\lfloor$	$\lceil$	$\lceil$
$/$	$/$	$\backslash$	$\backslash$	$\backslash$	$\backslash$	$\backslash$

Tabela 3.8: Duże ograniczniki

$\left($	$\left($	$\right)$	$\left($	$\right)$	$\left($	$\right)$
$\left[$	$\left[$	$\right]$	$\left[$	$\right]$	$\left[$	$\right]$
$\left\{$	$\left\{$	$\right\}$	$\left\{$	$\right\}$	$\left\{$	$\right\}$
$\left\lceil$	$\left\lceil$	$\right\rceil$	$\left\lceil$	$\right\rceil$	$\left\lceil$	$\right\rceil$

Tabela 3.9: Różne symbole

$\dots$	$\cdots$	$\vdots$	$\ddots$
$\hbar$	$\imath$	$\jmath$	$\ell$
$\Re$	$\Im$	$\aleph$	$\wp$
$\forall$	$\exists$	$\mho^a$	$\partial$
$'$	$\prime$	$\emptyset$	$\infty$
$\nabla$	$\triangle$	$\Box^a$	$\Diamond^a$
$\bot$	$\top$	$\angle$	$\surd$
$\diamondsuit$	$\heartsuit$	$\clubsuit$	$\spadesuit$
$\neg$	$\flat$	$\natural$	$\sharp$

<sup>a</sup> Dostępne po dołączeniu pakietu latexsym.

Tabela 3.10: Symbole niematematyczne

$\dagger$	$\S$	$\copyright$	$\ddagger$	$\P$	$\pounds$
-----------	------	--------------	------------	------	-----------

Polecenia te są dostępne również w trybie tekstowym.

Tabela 3.11: Ograniczniki (pakiet AMS)

$\ulcorner$	$\urcorner$	$\llcorner$	$\lrcorner$
-------------	-------------	-------------	-------------

Tabela 3.12: Symbole Greckie i Hebrajskie (pakiet AMS)

$\digamma$	$\varkappa$	$\beth$	$\daleth$	$\gimel$
------------	-------------	---------	-----------	----------

Tabela 3.13: Symbole relacji (pakiet AMS)

$\lessdot$	$\gtrdot$	$\doteqdot$ lub $\Doteq$
$\leqslant$	$\geqslant$	$\risingdotseq$
$\leqslantless$	$\leqslantgtr$	$\fallingdotseq$
$\leqq$	$\geqq$	$\eqcirc$
$\lll$ lub $\llless$	$\ggg$ lub $\gggtr$	$\circeq$
$\lesssim$	$\gtrsim$	$\triangleq$
$\lessapprox$	$\gtrapprox$	$\bumpeq$
$\lessgtr$	$\gtrless$	$\Bumpeq$
$\lesseqgtr$	$\gtreqless$	$\thicksim$
$\lesseqqgtr$	$\gtreqqless$	$\thickapprox$
$\preccurlyeq$	$\succcurlyeq$	$\approxeq$
$\curlyeqprec$	$\curlyeqsucc$	$\backsim$
$\prec$	$\succ$	$\backsimeq$
$\precapprox$	$\succapprox$	$\vDash$
$\subseteq$	$\supseteq$	$\Vdash$
$\Subset$	$\Supset$	$\Vvdash$
$\sqsubset$	$\sqsupset$	$\backepsilon$
$\therefore$	$\because$	$\varpropto$
$\shortmid$	$\shortparallel$	$\between$
$\smallsmile$	$\smallfrown$	$\pitchfork$
$\vartriangleleft$	$\vartriangleright$	$\blacktriangleleft$
$\trianglelefteq$	$\trianglerighteq$	$\blacktriangleright$

Tabela 3.14: Strzałki (pakiet AMS)

$\dashleftarrow$	$\dashrightarrow$	$\multimap$
$\leftrightsquigarrow$	$\rightleftarrows$	$\Uparrow$
$\leftrightsquigarrow$	$\rightleftarrows$	$\Downarrow$
$\Lleftarrow$	$\Rrightarrow$	$\Uparrow$
$\twoheadleftarrow$	$\twoheadrightarrow$	$\Uparrow$
$\leftarrowtail$	$\rightarrowtail$	$\Downarrow$
$\leftrightharpoons$	$\rightleftharpoons$	$\Downarrow$
$\leftrightsquigarrow$	$\rightsquigarrow$	$\Lsh$
$\looparrowleft$	$\looparrowright$	$\Rsh$
$\curvearrowleft$	$\curvearrowright$	
$\circlearrowleft$	$\circlearrowright$	

Tabela 3.15: Negacje symbolów relacji i strzałek (pakiet AMS)

$\nless$	$\ngtr$	$\varsubsetneqq$
$\lneq$	$\gneq$	$\varsupsetneqq$
$\nleq$	$\ngeq$	$\nsubseteqeq$
$\nleqslant$	$\ngeqslant$	$\nsupseteqeq$
$\lneqq$	$\gneqq$	$\nmid$
$\lvertneqq$	$\gvertneqq$	$\nparallel$
$\nleqq$	$\ngeqq$	$\nshortmid$
$\lnsim$	$\gnsim$	$\nshortparallel$
$\lnapprox$	$\gnapprox$	$\nsim$
$\nprec$	$\nsucc$	$\ncong$
$\npreceq$	$\nsucceq$	$\nvdash$
$\nprecneqq$	$\nsuccneqq$	$\nvDash$
$\nprecnsim$	$\succnsim$	$\nVdash$
$\nprecnapprox$	$\succnapprox$	$\nVDash$
$\subsetneq$	$\supsetneq$	$\ntriangleleft$
$\varsubsetneq$	$\varsupsetneq$	$\ntriangleright$
$\nsubseteq$	$\nsupseteq$	$\ntrianglelefteq$
$\subsetneqq$	$\supsetneqq$	$\ntrianglerighteq$
$\nleftarrow$	$\rightarrow$	$\nleftrightarrow$
$\nLeftarrow$	$\Rightarrow$	$\nLeftrightarrow$

Tabela 3.16: Relacje dwuargumentowe (pakiet AMS)

$\dot{+}$	$\centerdot$	$\intercal$
$\ltimes$	$\rtimes$	$\divideontimes$
$\Cup$ lub $\doublecup$	$\veebar$	$\smallsetminus$
$\Cap$ lub $\doublecap$	$\bar{\wedge}$	$\doublebarwedge$
$\boxplus$	$\boxminus$	$\circleddash$
$\boxtimes$	$\boxdot$	$\circledcirc$
$\leftthreetimes$	$\curlyvee$	$\circledast$
$\rightthreetimes$	$\curlywedge$	

Tabela 3.17: Różne symbole (pakiet AMS)

$\hbar$ <code>\hbar</code>	$\hslash$ <code>\hslash</code>	$\Bbbk$ <code>\Bbbk</code>
$\square$ <code>\square</code>	$\blacksquare$ <code>\blacksquare</code>	$\textcircled{S}$ <code>\circledS</code>
$\triangle$ <code>\vartriangle</code>	$\blacktriangle$ <code>\blacktriangle</code>	$\complement$ <code>\complement</code>
$\nabla$ <code>\triangledown</code>	$\blacktriangledown$ <code>\blacktriangledown</code>	$\oslash$ <code>\oslash</code>
$\lozenge$ <code>\lozenge</code>	$\blacklozenge$ <code>\blacklozenge</code>	$\bigstar$ <code>\bigstar</code>
$\angle$ <code>\angle</code>	$\measuredangle$ <code>\measuredangle</code>	$\sphericalangle$ <code>\sphericalangle</code>
$\diagup$ <code>\diagup</code>	$\diagdown$ <code>\diagdown</code>	$\backprime$ <code>\backprime</code>
$\nexists$ <code>\nexists</code>	$\Finv$ <code>\Finv</code>	$\varnothing$ <code>\varnothing</code>
$\eth$ <code>\eth</code>	$\mho$ <code>\mho</code>	

Tabela 3.18: Kroje pisma dostępne w trybie matematycznym

Przykład	Polecenie	Wymagany pakiet
$\mathrm{ABCdef}$	<code>\mathrm{ABCdef}</code>	
$\mathit{ABCdef}$	<code>\mathit{ABCdef}</code>	
$\mathnormal{ABCdef}$	<code>\mathnormal{ABCdef}</code>	
$\mathcal{ABC}$	<code>\mathcal{ABC}</code>	
$\mathcal{ABC}$	<code>\mathcal{ABC}</code>	eucal z opcją <code>mathcal</code> lub
$\mathscr{ABC}$	<code>\mathscr{ABC}</code>	eucal z opcją <code>mathscr</code>
$\mathfrak{ABCdef}$	<code>\mathfrak{ABCdef}</code>	eufrak
$\mathbf{ABC}$	<code>\mathbf{ABC}</code>	amsfonts lub amssymb

## Rozdział 4

# Rysunki, skorowidze, generowanie plików PDF...

*Kolej teraz na opis możliwości  $\text{\LaTeX}$ a przydatnych w pracy nad większymi dokumentami, takich jak: dołączania rysunków w dokumencie, tworzenie skorowidzów i spisów literatury. Bardziej szczegółowy opis tych i pokrewnych zagadnień można znaleźć w [12] oraz [6]. Pod koniec niniejszego rozdziału jest też mowa o tym, jak  $\text{\LaTeX}$  może generować pliki PDF.*

### 4.1. Włączanie grafiki w formacie EPS

Najprościej przygotować ilustrację w wyspecjalizowanym programie graficznym w rodzaju xfig, CorelDraw!, Freehand, gnuplot, itp., a później włączyć gotowy rysunek do dokumentu. Chociaż można to zrobić na wiele sposobów, tutaj przedstawimy jedynie sposób na dołączanie grafiki w formacie EPS (*Encapsulated PostScript*), jako że jest to technika prosta i szeroko stosowana<sup>1</sup>. Do pracy z grafiką w formacie EPS potrzebujemy albo drukarki wyposażonej w język Postscript, albo programu ghostscript, dostępnego na przykład pod adresem <http://www.cs.wisc.edu/~ghost/>. Program ghostscript oraz ułatwiające posługiwanie się nim graficzne nakładki, takie jak: ghostview, gv czy gsview są dostępne (pod wyżej wymienionym adresem) na wszystkie popularne platformy systemowe.

Wielu poleceń przydatnych do włączania rysunków dostarcza pakiet **graphicx** (autor: D. P. Carlisle), będący częścią zestawu o nazwie „graphics”<sup>2</sup>. Włączenie grafiki do dokumentu za jego pomocą możemy przedstawić w następujący sposób:

---

<sup>1</sup>Użytkownicy programów typu Office, którzy w tym momencie być może po raz pierwszy usłyszeli, że istnieje coś takiego jak język Postscript i jego wariant EPS, mogą być tym stwierdzeniem zdziwieni. Tak jednak jest w istocie: inne standardy obowiązują w biurze, a inne w przemyśle poligraficznym. Więcej na temat grafiki można znaleźć w [19].

<sup>2</sup>Zestaw „graphics” jest obowiązkową (ang. *required*) częścią każdej dystrybucji  $\text{\LaTeX}$ a, można go też znaleźć w katalogu `CTAN://macros/latex/required/graphics`.

1. W programie graficznym zachowujemy rysunek w formacie EPS<sup>3</sup> lub konwertujemy rysunek na format EPS, jeżeli dysponujemy już gotową grafiką ale w innym formacie<sup>4</sup>.
2. Dołączamy pakiet `graphicx` do preambuły dokumentu:

```
\usepackage[dvi-ps]{graphicx}
```

gdzie *dvi-ps* oznacza nazwę programu do konwersji pliku wyjściowego .dvi na plik postscriptowy. Najczęściej używanym do tego celu programem jest dvips. Nazwa sterownika jest tu konieczna, gdyż brakuje standardu dotyczącego dołączania grafiki postscriptowej w dokumentach TeXowych. Kierując się nazwą sterownika, pakiet `graphicx` potrafi do wynikowego pliku .dvi włączyć informację potrzebną do tego, by interpreter Postscriptu (w drukarce bądź w komputerze) poprawnie umieścił rysunek na wydruku.

3. Instrukcją:

```
\includegraphics[klucz=wartość,...]{plik}
```

włączamy *plik* do dokumentu. Parametr opcjonalny jest listą oddzielonych przecinkami *kluczy*, o określonych przez nas *wartościach*. Klucze wykorzystujemy do zmiany parametrów dołączanego rysunku, takich jak szerokość, wysokość czy kąt obrotu. W tabeli 4.1 zamieszczono najważniejsze klucze.

Tabela 4.1: Znaczenie ważniejszych kluczy polecenia `\includegraphics`

<code>width=w</code>	skalowanie rysunku do podanej szerokości <i>w</i>
<code>height=h</code>	skalowanie rysunku do podanej wysokości <i>h</i>
<code>angle=a</code>	obrót o kąt <i>a</i> (przeciwnie do ruchu wskazówek zegara)
<code>scale=s</code>	równomierne przeskalowanie w skali <i>s</i>

Poniższy przykład pomoże zrozumieć całą ideę:

<sup>3</sup>Jeżeli używany przez nas program graficzny na to *nie pozwala*, to spróbujmy zainstalować sterownik do drukarki postscriptowej, jak Apple Laser Writer, i drukować tym sterownikiem do pliku. Przy odrobinie szczęścia otrzymamy dokument w formacie EPS. Pamiętajmy, że plik EPS może zawierać tylko jedną stronę. Sterowniki niektórych drukarek można jawnie ustawić do tworzenia plików wyjściowych właśnie w tym formacie.

<sup>4</sup>W wypadku grafiki obwiedniowej i rastrowej, można do tego celu użyć programu ImageMagick (<http://www.imagemagick.org/>). Dobrym programem do konwersji, ale tylko grafiki rastrowej, jest gimp (<http://www.gimp.org>). Do konwersji zdjęć albo innych grafik w formacie JPG wygodnym narzędziem jest jpeg2ps <http://gnuwin32.sourceforge.net/packages/jpeg2ps.htm>. Niestety konwersja plików zapisanych w promowanym przez firmę Microsoft formacie WMF/EMF (MS Visio, MS Office, itp.) często daje opłakane rezultaty, ale to już jest wina wspomnianej firmy i stosowanej przez nią „strategii biznesowej”...

```
\begin{figure}
%\begin{center} zamiast \begin{center} użyj lepiej
\centering % bo \centering nie wstawia dodatkowego odstępu
\includegraphics[angle=90,width=0.5\textwidth]{sowauszata.eps}
\end{figure}
```

W powyższym przykładzie do dokumentu jest dołączany rysunek z pliku `sowauszata.eps`. Rysunek najpierw obracamy o  $90^\circ$  w kierunku przeciwnym do ruchu wskazówek zegara, a następnie przeskalowujemy tak, by nadać mu szerokość równą połowie szerokości szpalty. Skalowanie grafiki jest równomierne z uwagi na brak klucza `height`. Szerokość i wysokość rysunku możemy też określić w jednostkach bezwzględnych, takich jak punkty czy centymetry. W tabeli 6.5 zestawiono jednostki miar w  $\text{\LaTeX}$ u. Więcej informacji na temat powyższych zagadnień znajdziemy w [2] i [22].

Z powodów opisanych w punkcie 4.7.3, *zaleca się* podawać nazwę dołączanego pliku graficznego *bez rozszerzenia*, to znaczy lepiej zapisać powyższe polecenie `\includegraphics` następująco:

```
\includegraphics[angle=90,width=0.5\textwidth]{sowauszata}
```

W takiej sytuacji  $\text{\LaTeX}$  będzie szukał pliku `sowauszata` o rozszerzeniu adekwatnym do możliwości zadeklarowanego w poleceniu `\usepackage` sterownika. Przykładowo, jeżeli `graphicx` było wywołane z opcją `dvips`, to szukany będzie plik `sowauszata.eps`, jeżeli zaś użyjemy opcji `pdftex`, to będą szukane pliki o rozszerzeniach `.pdf`, `.jpg` i `.png`, a pierwszy znaleziony zostanie dołączony.

Rysunki najlepiej jest umieszczać w oddzielnym katalogu, będącym podkatalogiem tego, w którym jest dokument  $\text{\LaTeX}$ a. Bez względu na używany system operacyjny katalogi w ścieżce dostępu należy oddzielać znakiem <sup>5</sup>:

```
\includegraphics[width=0.5\textwidth]{rys/sowy/sowauszata}
```

Czasami wewnątrz jednego otoczenia `figure` chcemy umieścić kilka plików graficznych. W tym celu wystarczy, że umieścimy je obok siebie i w miarę potrzeby odpowiednio przeskalujemy:

```
\begin{figure}
\centering
%% http://pl.wikipedia.org/wiki/Ptaki_Polski
\includegraphics[width=.3\textwidth]{sowauszata}
\includegraphics[width=.3\textwidth]{puszczykmszarny}
\includegraphics[width=.3\textwidth]{bubobubo}%puchacz
\end{figure}
```

W powyższym przykładzie trzy rysunki zostaną umieszczone jeden obok drugiego. Ich łączna szerokość wyniesie  $3 \times 0,3 = 0,9$  szerokości kolumny

<sup>5</sup>Wszystkie uwagi dotyczące nazw plików opisane w punkcie 1.8 dla polecenia `\include` dotyczą także polecenia `\includegraphics`.

(`\textwidth` w języku  $\text{\LaTeX}$ ), między rysunkami wstawiony zostanie odstęp wielkości zwykłej spacji, co wynika z zasady  $\text{\LaTeX}$ a, że pojedyncza zmiana wiersza jest traktowana jak odstęp. Całość zostanie wyśrodkowana w osi szpalty. Pamiętajmy, żeby do tego stosować polecenie `\centering` a nie otoczenie `center`, gdyż to drugie wstawia dodatkowy a zbędny odstęp pionowy.

Aby efekt końcowy na wydruku był zadowalający, rysunki powinny mieć identyczne wymiary. Inaczej, albo poszczególne obrazki będą optycznie niezgodnie (np. sowa uszata będzie 3 razy większa od puchacza), albo też poszczególne rysunki składowe będą miały różne wymiary (rysunek z sową uszatą będzie dwa razy mniejszy niż rysunek z puchaczem).

Czasami rysunki czy tabele zmieściłyby się na stronie, gdyby obrócić je o  $90^\circ$ . W tym celu można skorzystać z pakietu `rotating`. Do obracania dowolnego fragmentu tekstu, pudełka (por. punkt 6.6), tabeli itp. należy używać otoczenia `rotate`:

```
\begin{rotate}{kąt} ... \end{rotate}
```

Do obrócenia obiektu o kąt  $90^\circ$  należy użyć otoczenia `sideways`:

```
\begin{sideways} ... \end{sideways}
```

Do obrócenia tabeli łącznie z podpisem pod tabelą należy zastosować otoczenie `sidewaystable` (uwaga: tabela jest umieszczana na osobnej stronie):

```
\begin{sidewaystable} ... \end{sidewaystable}
```

Do obrócenia rysunku łącznie z podpisem pod rysunkiem należy użyć otoczenia `sidewaysfigure` (uwaga: rysunek jest umieszczany na osobnej stronie):

```
\begin{sidewaysfigure} ... \end{sidewaysfigure}
```

## 4.2. Spis literatury

Do przygotowania spisu literatury służy otoczenie `thebibliography`. Każda pozycja w tym otoczeniu ma postać polecenia:

```
\bibitem{etykieta}
```

*Etykiety* można następnie użyć do zacytowania w dokumencie tej pozycji, czyli książki, artykułu bądź pracy konferencyjnej:

```
\cite{etykieta}
```

Numerowanie pozycji literaturowych jest automatyczne (polecenia `\bibitem` i `\cite` działają podobnie jak opisane już instrukcje `\label` i `\ref`). Otoczenie `thebibliography` ma jeden parametr, który powinien zawierać tekst przynajmniej tak szeroki jak najszersza etykieta ze spisu. W poniższym przykładzie zapis 99 oznacza, że numery pozycji w spisie będą co najwyżej dwucyfrowe.



Partl~\cite{pa} zaproponował,  
żeby \ldots

```
\begin{thebibliography}{99}
\bibitem{pa} H.~Partl:
\emph{German \TeX},
TUGboat Vol.~9, No.~1 ('88)
\end{thebibliography}
```

Partl [1] zaproponował, żeby ...

### Bibliografia

- [1] H. Partl: *German T<sub>E</sub>X*, TUGboat Vol. 9,  
No. 1 ('88)

Do większych projektów przydaje się program o nazwie BibT<sub>E</sub>X. Program ten znajduje się w każdej współczesnej dystrybucji T<sub>E</sub>Xa. BibT<sub>E</sub>X korzysta z bazy bibliograficznej (biblioteki), z której wybiera tylko te pozycje literaturowe, które były cytowane w dokumencie. Sposób formatowania spisów literatury jest sterowany za pomocą specjalnych szablonów, których modyfikacja umożliwia zmianę układu graficznego spisu.

Po przetworzeniu pliku, na podstawie zawartości etykiet zapisanych przez L<sup>A</sup>T<sub>E</sub>Xa do pliku `.aux`, BibT<sub>E</sub>X tworzy spis literatury obejmujący tylko te pozycje z biblioteki, które cytowano w dokumencie. Zwykle plik ten ma rozszerzenie `.bib`. Format spisu zależy od specyfikacji szablonu znajdującej się w pliku o rozszerzeniu `.bst`, a jest zapisywany do pliku o rozszerzeniu `.bb1`. Do poprawnego sformatowania bibliografii i cytowań konieczne jest przynajmniej trzykrotne przetworzenie dokumentu L<sup>A</sup>T<sub>E</sub>Xem. Więcej informacji na temat przygotowywania spisu literatury za pomocą BibT<sub>E</sub>Xa zawiera [1].

## 4.3. Skorowidze

Niezwykle użytecznym elementem wielu książek jest skorowidz. Można go utworzyć stosunkowo łatwo za pomocą L<sup>A</sup>T<sub>E</sub>Xa oraz programu narzędziowego o nazwie `makeindex`<sup>6</sup>. W tym wprowadzeniu omówimy jedynie podstawowe polecenia dotyczące skorowidzów. Jak zawsze, więcej informacji znajdziemy w [6].

Generować hasła do skorowidza można dopiero po załadowaniu w preambule dokumentu pakietu o nazwie `makeidx`:

```
\usepackage{makeidx}
```

oraz wstawieniu (także w obrębie preambuły) instrukcji:

```
\makeindex
```

<sup>6</sup>Albo `makeidx`, jeśli nasz system operacyjny nie pozwala używać nazw dłuższych niż 8 znaków.

Hasło wstawiamy do skorowidza poleceniem:

```
\index{hasło}
```

gdzie *hasło* oznacza pozycję w skorowidzu. Polecenie `\index{hasło}` umieszczamy w pliku źródłowym bezpośrednio w miejscu związanym z określonym hasłem. W tabeli 4.2 przedstawiono przykłady użycia *hasel*.

Tabela 4.2: Przykłady składni polecenia `\index`

Przykład	Hasło	Uwagi
<code>\index{kot}</code>	kot, 1	hasło pierwszego stopnia
<code>\index{kot!rudy}</code>	rudy, 3	hasło drugiego stopnia
<code>\index{Sam@\textsl{Sam}}</code>	<i>Sam</i> , 2	hasło sformatowane
<code>\index{Lin@\textbf{Lin}}</code>	<b>Lin</b> , 7	hasło sformatowane
<code>\index{Cadiz@C\'adiz}</code>	Cádiz, 77	poprawne sortowanie
<code>\index{Jenny textbf}</code>	Jenny, <b>3</b>	formatowanie numeru strony
<code>\index{Joe textit}</code>	Joe, 5	formatowanie numeru strony
<code>\index{kot see{felis}}</code>	kot, zob. felis	relacja pomiędzy hasłami

Aby wstawić do skorowidza literalnie znak `!`, `@` albo `|` należy poprzedzić go znakiem `"`.

Podczas przetwarzania pliku źródłowego przez  $\text{\LaTeX}$  każda instrukcja `\index` powoduje, że do pliku pomocniczego zostaje zapisana odpowiednia pozycja skorowidza wraz z bieżącym numerem strony. Plik pomocniczy nosi tę samą nazwę co główny plik źródłowy, ma jednak rozszerzenie `.idx`. Trzeba go następnie przetworzyć programem `makeindex`, w taki oto sposób:

```
makeindex plik
```

W rezultacie program `makeindex` tworzy posortowany skorowidz i zapisuje go do pliku o nazwie identycznej z nazwą głównego pliku źródłowego i o rozszerzeniu `.ind`. Jeżeli jeszcze raz przetworzymy plik źródłowy, to tym razem skorowidz zostanie włączony w miejscu wystąpienia polecenia:

```
\printindex
```

Pakiet `showidx`, wchodzący w skład standardowej dystrybucji  $\text{\LaTeX 2}_{\epsilon}$ , drukuje pozycje skorowidza na lewym marginesie szpalty. Jest on dość przydatny do sprawdzania i korygowania skorowidza.

Reguły sortowania skorowidza zależą oczywiście od języka, w którym piszemy dokument. Pod tym względem program `makeindex` jest nieprzydatny dla dokumentów polskojęzycznych, gdyż sortuje hasła jedynie według zasad języka angielskiego. Wersją programu `makeindex` przystosowaną do polskich reguł sortowania wyrazów jest `plmindex` (autor: Włodzimierz Macewicz).

Potrafi on tworzyć skorowidz według reguł angielskich jak i polskich. Program ten jest dostępny na przykład pod adresem <http://www.ia.pw.edu.pl/~wujek/tex/idx/plmindex.zip>. Więcej informacji na temat polskich zasad tworzenia skorowidzów można znaleźć w [17].

Umieszczając w dokumencie polecenie `\index`, powinniśmy zwrócić uwagę na odstępy. Oto przykład, jak może to wpłynąć na skład:

Moje słowo `\index{słowo}`. Inaczej  
niż `słowo\index{słowo}`. Zwróćmy  
uwagę na pozycję kropki.

Moje słowo . Inaczej niż słowo. Zwróćmy  
uwagę na pozycję kropki.

## 4.4. Paginy górne i dolne

Pakiet `fancyhdr` (autor Piet van Oostrum), który można znaleźć w katalogu: [CTAN://macros/latex/contrib/fancyhdr/](http://CTAN://macros/latex/contrib/fancyhdr/), udostępnia polecenia do definiowania zawartości pagin. Zwróćmy uwagę na różnice w wyglądzie pagin na stronach: bieżącej i poprzedniej. Oprócz numeru strony, w paginie górnej na stronie parzystej znajduje się tytuł rozdziału, a na stronie nieparzystej dodatkowo tytuł punktu. Pagine, której zawartość podąża za treścią dokumentu, nazywa się fachowo żywą paginą.

---

```
\documentclass{book}
\usepackage{fancyhdr}
\pagestyle{fancy}
% zmiana liter w żywej paginie na małe
\renewcommand{\chaptermark}[1]{\markboth{#1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection\ #1}}
\fancyhf{} % usuń bieżące ustawienia pagin
\fancyhead[LE,R0]{\small\bfseries\thepage}
\fancyhead[LO]{\small\bfseries\rightmark}
\fancyhead[RE]{\small\bfseries\leftmark}
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}
\addtolength{\headheight}{0.5pt} % pionowy odstęp na kreskę
\fancypagestyle{plain}{%
\fancyhead{} % usuń p. górne na stronach pozbawionych
% numeracji (plain)
\renewcommand{\headrulewidth}{0pt} % pozioma kreska
}
```

---

Rysunek 4.1: Przykład wykorzystania pakietu `fancyhdr`

Za pomocą  $\text{\LaTeX}$ a można sobie łatwo poradzić z automatycznym umieszczaniem odpowiedniej informacji w żywej paginie. Rozwiązanie jest następujące. W definicjach poleceń składających paginy wykonanie instrukcji  $\text{\rightmark}$  oraz  $\text{\leftmark}$  wstawia odpowiedni tytuł rozdziału, punktu lub cokolwiek innego. Poleceniom  $\text{\rightmark}$  i  $\text{\leftmark}$  jest nadawane nowe znaczenie (nowe wartości) za każdym wykonaniem instrukcji składania tytułu rozdziału i punktu ( $\text{\chapter}$ ,  $\text{\section}$ ).

W rzeczywistości,  $\text{\chapter}$  oraz inne polecenia podziału logicznego dokumentu nie zmieniają znaczenia poleceń  $\text{\rightmark}$  i  $\text{\leftmark}$ . Odwołują się one natomiast do poleceń  $\text{\chaptermark}$ ,  $\text{\sectionmark}$  oraz  $\text{\subsectionmark}$ . Dopiero użycie tych instrukcji powoduje zmianę definicji poleceń  $\text{\rightmark}$  i  $\text{\leftmark}$ .

Do zmiany postaci tytułu rozdziału w paginie wystarczy modyfikacja polecenia  $\text{\chaptermark}$ . Rysunek 4.1 przedstawia takie wykorzystanie pakietu, że paginy górne będą wyglądać mniej więcej tak jak w tym podręczniku. Kompletny opis pakietu znajduje się w jego dokumentacji.

## 4.5. Pakiet verbatim

Pakiet `verbatim` udostępnia poprawioną wersję standardowego otoczenia `verbatim` (opisanego na stronie 38). Oprócz wielu drobnych, chociaż istotnych ulepszeń pakiet udostępnia polecenie:

$\text{\verbatiminput}\{plik\}$

które dołącza do dokumentu plik tekstowy tak, jakby jego zawartość znajdowała się wewnątrz otoczenia `verbatim`.

Pakiet `verbatim` jest częścią grupy pakietów o nazwie „tools”. Wchodzi one w skład standardowej dystrybucji  $\text{\LaTeX}$ a. Więcej szczegółów znajdziemy w [25].

## 4.6. Instalowanie dodatkowych pakietów

Ten punkt jest adresowany raczej do tych czytelników, którzy mają już doświadczenie w  $\text{\LaTeX}$ u, do składania dokumentów używają własnych poleceń i otoczeń, a odczuwają potrzebę uporządkowania sobie środowiska pracy bądź udostępnienia go innym. Czytelnicy początkujący mogą bez wielkiej straty pominąć ten punkt i kontynuować czytanie od następnego punktu.

W większość instalacji  $\text{\LaTeX}$ a wbudowany jest bogaty zestaw pakietów dodatkowych, jeszcze ich więcej można znaleźć w sieci. Głównym miejscem, gdzie można je znaleźć, jest archiwum CTAN (<http://www.ctan.org/>).

Pakiety takie jak `geometry` albo `hyphenat`, a także wiele innych, składają się na ogół z dwóch plików: jednego o rozszerzeniu `.ins` i drugiego

o rozszerzeniu `.dtx`. Często jest też dołączany plik `readme.txt` z krótkim opisem pakietu. Ma się rozumieć, zawsze warto zaczynać od przeczytania tego właśnie pliku.

Po skopiowaniu plików pakietu na nasz komputer trzeba je w ten czy inny sposób przetworzyć, aby po pierwsze wprowadzić do dystrybucji  $\text{T}_{\text{E}}\text{X}$ a informację o nowym pakiecie, a po drugie uzyskać dokumentację. Oto, jak osiąga się pierwszy z tych celów:

1. Uruchamiamy  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ a na pliku `.ins`. To powoduje wygenerowanie pliku `.sty`.
2. Plik `.sty` kopiujemy w miejsce, w którym nasza dystrybucja  $\text{T}_{\text{E}}\text{X}$ a potrafi go odnaleźć. Zazwyczaj jest to katalog o nazwie `.../localtexmf/tex/latex` (użytkownicy systemów MS Windows oraz OS/2 wiedzą, że w ich systemie trzeba używać znaku ukośnika pochyłonego przeciwnie).
3. Odświeżamy zawartą w dystrybucji bazę danych nazw plików. Odpowiednie polecenie zależy od dystrybucji  $\text{T}_{\text{E}}\text{X}$ a: w  $\text{t}_{\text{E}}\text{X}$ u i  $\text{f}_{\text{P}}\text{P}_{\text{E}}\text{X}$ u jest to `maktexlsr`; a w  $\text{Mik}_{\text{E}}\text{X}$ u – `initexmf -update-fndb`. W dystrybucjach wyposażonych w środowisko graficzne (np.  $\text{Mik}_{\text{E}}\text{X}$ ) do odświeżenia bazy nazw wystarczy wybrać odpowiednią pozycję z menu.

Można teraz wygenerować dokumentację z pliku `.dtx`:

1. Uruchamiamy  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ a na pliku `.dtx`, w wyniku czego powstanie plik `.dvi`. Niewykluczone, że zanim w dokumentacji uporządkują się numery odsyłaczy, będziemy musieli uruchomić  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ a kilka razy.
2. Sprawdzamy, czy wśród wielu plików, które wygenerował  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ , jest plik `.idx`. Jeśli go nie widać, to możemy przejść do kroku 5.
3. Aby wygenerować skorowidz, piszemy linijkę rozkazową:

```
makeindex -s gind.ist nazwa
```

gdzie *nazwa* oznacza nazwę głównego spośród przetwarzanych plików, bez rozszerzenia.

4. Ponownie uruchamiamy  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ a na pliku `.dtx`.
5. Na koniec coś równie ważnego: aby uprzyjemnić sobie czytanie, tworzymy plik `.ps` albo `.pdf`.

Czasami wśród wygenerowanych plików znajdziemy jeszcze plik `.glo`. W takiej sytuacji między krokami 4 i 5 powinniśmy wykonać:

```
makeindex -s gglo.ist -o name.gls name.glo
```

Zawsze przed przejściem do kroku 5 musimy mieć pewność, że w poprzednim kroku uruchomiliśmy  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ a na pliku `.dtx`.

## 4.7. Tworzenie plików PDF: pdf $\LaTeX$ i hyperref

PDF (*Portable Document Format*) to format hipertekstowy. Podobnie jak na stronie internetowej, niektóre słowa są w dokumencie zaznaczone jako hiperłącza. Łączą one miejsce w dokumencie z innymi jego miejscami bądź nawet z innymi dokumentami; gdy klikamy w takie hiperłącze, jesteśmy przeniesieni do miejsca docelowego. W kontekście  $\LaTeX$ a oznacza to, że wszystkie wystąpienia poleceń `\ref` i `\pageref` stają się hiperłączami. Na dodatek spis treści, skorowidz i wszystkie podobne struktury stają się zestawami łączy.

Większość spotykanych dziś w Internecie stron jest napisana w języku HTML (*HyperText Markup Language*). Kiedy przychodzi do pisania pracy naukowej, format ten ma dwie wady:

1. Nie ma dobrej metody włączania wzorów matematycznych. Chociaż został zdefiniowany odpowiedni standard, to jednak większość przeglądarek go nie obsługuje, brakuje też w komputerach odpowiednich fontów.
2. Dokumenty HTML można co prawda drukować, ale wydruki wyglądają różnie w różnych przeglądarkach, a także na różnych platformach. Od jakości, do której przyzwyczajają nas  $\LaTeX$ , wydruki te dzieli przepaść.

Próbowano już na wiele sposobów napisać program tłumaczący język  $\LaTeX$ a na HTML. Niektóre z tych programów dają nawet niezłe wyniki – w tym sensie, że są w stanie ze źródła w standardowym  $\LaTeX$ u wygenerować czytelny dokument HTML. Wszystkim jednak zawsze czegoś brakuje. Gdy tylko zachodzi konieczność użycia bardziej wyrafinowanych możliwości  $\LaTeX$ a bądź jego pakietów zewnętrznych, jakość spada. Autorom, którym zależy na publikowaniu w sieci WWW, a jednocześnie chcą utrzymać wysoką jakość typograficzną, pozostaje w zasadzie wyłącznie PDF. Większość współczesnych przeglądarek sieciowych potrafi wyświetlać dokumenty w tym formacie.

Chociaż przeglądarki do plików w formatach DVI i PS są dostępne na prawie każdej platformie, to jeszcze powszechniej spotyka się przeglądarki do plików PDF, takie jak Acrobat Reader i xpdf. Tak więc, jeśli naszą pracę udostępnimy w postaci PDF, to stanie się ona łatwiej dostępna dla grona potencjalnych czytelników.

### 4.7.1. Tworzenie dokumentów PDF w $\LaTeX$ u

Są dwa sposoby<sup>7</sup> zamiany plików źródłowych  $\LaTeX$ a na dokument PDF. Pierwszy z nich polega na zamianie pliku DVI na dokument PS (zwykle za pomocą programu dvips), który następnie jest transformowany do formatu PDF bądź programem ghostscript, bądź z użyciem komercyjnego narzędzia Acrobat Distiller, firmy Adobe. Drugi sposób to bezpośrednie generowanie

<sup>7</sup>Tych sposobów jest więcej, ale przedstawione dwa są zdecydowanie najczęściej używane.

dokumentu PDF za pomocą programu pdfT<sub>E</sub>X. Pierwszy z wymienionych sposobów, mimo że na pierwszy rzut oka wydaje się bardziej pracochłonny, ma tę przewagę nad drugim, że umożliwia wykorzystanie wielu wartościowych pakietów stosujących język Postscript, choćby takich jak `rotate` czy `pstricks`. Z kolei utworzenie pliku PDF bezpośrednio z pliku źródłowego, za pomocą programu pdfT<sub>E</sub>X, jest prostsze. Którąkolwiek z metod wybierzesz, w większości przypadków efekt końcowy będzie identyczny.

Program pdfT<sub>E</sub>X, opracowany przez Hàn Thê Thànha, generuje dokument PDF na podobnej zasadzie, jak T<sub>E</sub>X produkuje plik DVI. Istnieje też pdfL<sup>A</sup>T<sub>E</sub>X, który tworzy pliki PDF ze źródeł L<sup>A</sup>T<sub>E</sub>Xowych. Zarówno pdfT<sub>E</sub>X, jak i pdfL<sup>A</sup>T<sub>E</sub>X wchodzi w skład wszystkich współczesnych dystrybucji T<sub>E</sub>Xa, takich jak: teT<sub>E</sub>X, fpT<sub>E</sub>X, MikT<sub>E</sub>X, T<sub>E</sub>XLive czy CMacT<sub>E</sub>X.

Do wygenerowania pliku PDF zamiast DVI wystarczy zastąpić polecenie `latex file.tex` instrukcją `pdflatex file.tex`. W zintegrowanym środowisku graficznym, takim jak TeXnicCenter czy Kile, uruchomienie obu programów polega na wybraniu odpowiedniego przycisku z menu.

Tworzenie dobrej jakości dokumentów PDF wymaga: użycia *odpowiednich* fontów, dołączenia grafiki we *właściwym* formacie oraz zdefiniowania hiperłączy dla elementów takich jak odsyłacze, spisy czy skorowidze. W kolejnych punktach omówimy te zagadnienia szczegółowo. Jeszcze bardziej detaliczny opis można znaleźć w [24].

#### 4.7.2. Fonty bitmapowe i obwiedniowe

Fonty są bardzo ważną częścią każdego systemu składu. Font to w istocie program komputerowy, zawierający opisy znaków oraz dodatkowe informacje określające sposób ich drukowania i pozycjonowania na stronie. Większość fontów jest komercyjna i nie może być swobodnie rozpowszechniana. Z tego powodu system T<sub>E</sub>X nie używa fontów systemowych, na przykład fontów dostępnych w systemie MS Windows, lecz wykorzystuje alternatywne kroje pisma o statusie Oprogramowania Otwartego. Inaczej, dokumentu przygotowanego na przykład w systemie MS Windows nie można by poprawnie przetworzyć w systemie Linux albo, jeżeli nawet dałoby się go przetworzyć, to otrzymalibyśmy dokument wyglądający inaczej<sup>8</sup>.

Z punktu widzenia „technologii komputerowej” fonty dzielimy na bitmapowe i obwiedniowe, przy czym te pierwsze – za wyjątkiem systemu T<sub>E</sub>X – mają znaczenie historyczne. Istnieje kilka standardów tworzenia fontów obwiedniowych: fonty postscriptowe (tzw. fonty Type 1 albo *typu pierwszego*), fonty TrueType oraz najnowsze OpenType. „Klasyczne” bitmapowe fonty T<sub>E</sub>Xa określa się akronimem PK.

<sup>8</sup>Inne na przykład byłoby łamanie wierszy i podział na strony. W edytorze MS Word nie jest to zresztą uważane za błąd, bo w tamtym programie dokument jest przenośny, ale tylko razem z komputerem, bo przesłanie dokumentu na inny komputer z reguły powoduje, że jest on złożony inaczej.



L<sup>A</sup>T<sub>E</sub>X umie korzystać zarówno z fontów PK, jak i z fontów Type 1, zaś pdfL<sup>A</sup>T<sub>E</sub>X potrafi dodatkowo używać fontów TrueType i OpenType. Jednakże dokument z tradycyjnymi fontami PK bardzo źle wygląda w przeglądarce Acrobat Reader. Najlepiej używać wyłącznie fontów Type 1, bo wówczas dokumenty zawsze wyświetlają się poprawnie. Współczesne instalacje T<sub>E</sub>Xa są domyślnie skonfigurowane tak, by używane były fonty Type 1<sup>9</sup>. W zależności od sposobu polonizacji dokumentu (por. punkt 2.5.1) L<sup>A</sup>T<sub>E</sub>X albo wykorzysta obwiedniowe wersje fontów PL, albo przełączy się na fonty z pakietu cm-super, które są implementacją fontów EC w technologii Type 1.

Ostatnio dostępna stała się wysokiej jakości rodzina fontów obwiedniowych LM (*Latin Modern*). Jeśli dysponujemy świeżej daty instalacją T<sub>E</sub>Xa, to jest całkiem możliwe, że zawiera ona już tę rodzinę. Wystarczy w takim wypadku dodać do preambuły:

```
\usepackage{lmodern}
\usepackage[T1]{fontenc}
```

Uwaga! Rodzina LM nie zawiera na razie fontów matematycznych, dlatego do składu formuł i tak potrzebne są albo fonty PL, albo oryginalne fonty L<sup>A</sup>T<sub>E</sub>Xowe.

Uwaga! Jeśli w pliku o rozszerzeniu .log, zawierającym raport z kompilacji, pojawiają się komunikaty w rodzaju:

```
Warning: pdftex (file eurmo10): Font eur... not found
```

to oznacza, że któryś z fontów potrzebnych do przetworzenia dokumentu nie został odnaleziony. Problem taki wymaga rzetelnej naprawy, gdyż przeglądarka plików PDF *w ogóle nie wyświetli stron ze znakami z brakującego fontu*.

#### 4.7.3. Dołączanie grafiki

Do włączania grafiki najlepiej korzystać z pakietu **graphicx** (zobacz str. 65). Po wpisaniu **pdftex** jako opcji *sterownik* pakiet zadziała też w pdfL<sup>A</sup>T<sub>E</sub>Xu:

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

W tym przykładzie pojawia się również pakiet **color**, jako że stosowanie kolorów w sieci jest naturalne.

Tyle dobrych wieści. Złe są takie, że grafika w formacie EPS nie działa w pdfL<sup>A</sup>T<sub>E</sub>Xu. Jeśli w argumencie instrukcji **\includegraphics** pominiemy rozszerzenie nazwy pliku, to pakiet **graphicx** będzie szukał pliku w formacie zależnym od opcji *sterownik*. W programie pdfT<sub>E</sub>X dopuszczalny jest jeden z formatów: .png, .pdf, .jpg, .mps (METAPOST), ale *nie* .eps.

<sup>9</sup>Aby się przekonać, czy tak jest w wypadku twojej instalacji, utwórz plik PDF (na przykład za pomocą pdfL<sup>A</sup>T<sub>E</sub>X) i wyświetl go w oknie przeglądarki Acrobat Reader. Jeżeli fonty na ekranie są nieczytelne i mają poszarpane brzegi, to masz problem. W nowych dystrybucjach L<sup>A</sup>T<sub>E</sub>Xa problem ten nie powinien wystąpić, więc jeżeli go napotkałeś, to przypuszczalnie masz zainstalowaną zbyt okrojoną wersję systemu T<sub>E</sub>X.



Prostym sposobem wyjścia z tej sytuacji jest konwersja plików EPS na format PDF programem epstopdf, dostępnym na wielu platformach. Dla grafiki wektorowej (rysunków) jest to wspaniałe rozwiązanie. Dla mappek bitowych (zdjęcia, skany) nie jest ono idealne, bo format PDF z natury obsługuje włączanie obrazów w formatach PNG i JPEG. Format PNG jest dobry do zdjęć z ekranów i innych obrazów z niewielką liczbą kolorów. Format JPEG jest świetny do zdjęć, jako że zużywa na nie mało pamięci.

Niekiedy zamiast rysowania figur geometrycznych korzystniejsze jest opisywanie ich w wyspecjalizowanym języku poleceń, na przykład takim, jaki jest stosowany w programie METAPOST, który – wraz z obszernym podręcznikiem – wchodzi w skład większości dystrybucji T~~E~~Xa.

#### 4.7.4. Łączy hipertekstowe

O kierowanie wewnętrznych odsyłaczy we właściwe miejsca dokumentu za-  
dąba pakiet hyperref. Aby zadziałał poprawnie, zaleca się umieścić polecenie `\usepackage{hyperref}` jako *ostatnie* w preambule dokumentu. Sposobem działania pakietu hyperref można sterować za pomocą wielu opcji, które po-  
dajemy bądź „tradycyjne”, czyli jako listę rozdzielonych przecinkami opcji po `\usepackage`, wewnątrz nawiasów kwadratowych, bądź jako argument polecenia `\hypersetup{opcje}`.

Opcje `pdftex` i `dvips` są kluczowe, bo określają metodę generowania dokumentu PDF: bezpośrednio za pomocą programu pdfT~~E~~X albo pośrednio poprzez zamianę wynikowego pliku DVI na PS, a potem PDF.

W poniższym wykazie wartości domyślne są podane pismem prostym.

`bookmarks (=true, false)` w trakcie wyświetlania dokumentu Acrobatem  
pokaż bądź ukryj pasek zakładek;  
`unicode (=false, true)` pozwól w zakładkach Acrobatów używać znaków  
z alfabetów nielacińskich;  
`pdftoolbar (=true, false)` pokaż bądź ukryj pasek narzędziowy Acro-  
bata;  
`pdfmenubar (=true, false)` pokaż bądź ukryj menu Acrobatów;  
`pdf-fit-window (=true, false)` dostosuj wielkość wyświetlanego PDF-a do  
wielkości okna;  
`pdftitle (= {napis})` tytuł dokumentu;  
`pdfauthor (= {napis})` nazwisko autora;  
`pdf-new-window (=true, false)` określa, czy w wypadku gdy łączy prowadzi  
poza dokument, ma być otwierane nowe okno;  
`colorlinks (=false, true)` określa, czy otoczyć hiperłącza kolorowymi  
ramkami (`false`) czy kolorować same hiperłącza (`true`). Kolory można  
konfigurować za pomocą następujących opcji (w nawiasach kolory  
domyślne):  
`linkcolor (=red)` kolor łączy wewnętrznych (rozdziałów, punktów,  
stron itp.),

`citecolor (=green)` kolor cytowań, czyli odsyłaczy do bibliografii,  
`filecolor (=magenta)` kolor odsyłaczy do plików,  
`urlcolor (=cyan)` kolor łączy typu URL (adresy poczty elektronicznej bądź sieciowe).

Jeśli zadowolają nas ustawienia domyślne, to wpisujemy:

```
\usepackage[pdftex]{hyperref} %ew. dvips zamiast pdftex
```

Jeśli przeglądarka ma pokazać listę zakładek, a łączy mają być kolorowane (wartości domyślnych `=true` nie trzeba podawać), to:

```
\usepackage[pdftex,bookmarks,colorlinks]{hyperref}
```

Gdy tworzymy dokumenty PDF przeznaczone do druku czarno-białego, to kolorowanie łączy nie jest najlepszym pomysłem, bo w wydrukach wychodzą one szare, co utrudnia czytanie. Zamiast tego możemy użyć kolorowych ramek, które nie są drukowane:

```
\usepackage{hyperref}
\hypersetup{colorlinks=false}
```

albo dla łączy używać koloru czarnego:

```
\usepackage{hyperref}
\hypersetup{colorlinks,citecolor=black,%
filecolor=black,linkcolor=black,urlcolor=black,pdftex}
```

Jeśli chcemy dodać informacji o dokumencie PDF, które Acrobat wyświetla w oknie *Document Properties*:

```
\usepackage[pdftauthor={P. Desproges},pdftitle={Des femmes
qui tombent},pdftex]{hyperref}
```

Oprócz hiperłączy tworzonych automatycznie możemy umieszczać własne:

```
\href{url}{text}
```

Przykładowo, w wyniku wykonania następującego fragmentu:

Na stronie `\href{http://www.ctan.org}{CTAN}`.

zostanie wydrukowane słowo „CTAN”, kliknięcie w to słowo przeniesie nas na internetową stronę CTAN-u.

Jeśli docelowe miejsce łączy nie jest URL-em, lecz plikiem lokalnym, to możemy użyć polecenia `\href` w następującej postaci:

Pełny dokument jest w `\href{manual.pdf}{pliku}`.

Generuje to tekst „Pełny dokument jest w [pliku](#)”. Klikając w słowo „pliku”, otworzymy plik `manual.pdf`. (Nazwa pliku jest interpretowana względem położenia dokumentu PDF).

Autorka dokumentu może zachęcić czytelników do korespondencji elektronicznej, wpisując instrukcję `\href` jako fragment polecenia `\author` na stronie tytułowej dokumentu:

```
\author{Maria Oetiker $<$\href{mailto:mary@oetiker.ch}%
{mary@oetiker.ch}$>$}
```

Zauważmy, że łącze zapisano tu tak, iż adres elektroniczny pojawia się zarówno w łączu, jak i na stronie. Gdyby podać go tak:

```
\href{mailto:mary@oetiker.ch}{Maria Oetiker}
```

to łącze działałoby w programie Acrobat, lecz adres nie byłby widoczny w wydruku.

#### 4.7.5. Problemy z łączami

Ukazaniu się komunikatu w rodzaju:

```
! pdfTeX warning (ext4): destination with the same
  identifier (name{page.1}) has been already used,
  duplicate ignored
```

towarzyszy reinicjalizacja licznika, na przykład w wyniku użycia polecenia `\mainmatter`, dostępnego w klasie dokumentów `book`. Ustala ono wartość licznika stron na 1 tuż przed pierwszym rozdziałem książki. Ale ponieważ również i wstęp do książki zawiera stronę 1, to odsyłacz „page.1” staje się niejednoznaczny, stąd notka „duplicate ignored”.

Możemy temu przeciwdziałać, dodając opcję `plainpages=false`. Pomaga to niestety tylko w odniesieniu do licznika stron. Radykalniejszym rozwiązaniem jest opcja `hypertextnames=false`, która jednak powoduje, że przestają działać odsyłacze do stron w skorowidzu.

#### 4.7.6. Problemy z zakładkami

Tekst na zakładkach nie zawsze wygląda zgodnie z naszymi oczekiwaniami. Ponieważ zakładki są traktowane jako „czysto tekstowe”, może w nich wystąpić mniejszy zakres znaków niż w normalnym tekście L<sup>A</sup>T<sub>E</sub>Xowym. Pakiet `hyperref` zauważył tego typu problem i zasygnalizuje go komunikatem:

```
Package hyperref Warning:
Token not allowed in a PDFDocEncoded string:
```

Możemy ten problem obejść, podając napis przeznaczony na zakładkę, który ma zastąpić wywołujący trudność tekst:

```
\texorpdfstring{TEX tekst}{Napis na zakładkę}
```

Podstawowym kandydatem do zastosowania takiego postępowania są wzory matematyczne:

```
\section{\texorpdfstring{$E=mc^2$}{E=mc^2}}
```

Śródtytuł „ $E=mc^2$ ” przekształca się na zakładce w „E=mc2”.

Także zmiany kolorów nie przenoszą się dobrze na zakładki:

```
\section{\textcolor{czerwony}{Czerwony !}}
```

Na zakładce otrzymamy „redRed!”. Polecenie `\textcolor` jest tu ignorowane, chociaż wyświetlany jest jego argument (na czerwono). Lepszy wynik uzyskamy, wpisując:

```
\section{\texorpdfstring{\textcolor{red}{Red !}}{Red\ !}}
```

Jeśli w dokumencie używamy unikodu, a pakietowi `hyperref` podamy opcję `unicode`, to na zakładkach będą mogły wystąpić znaki unikodowe. Poszerza to zakres znaków, które można przekazać w poleceniu `\texorpdfstring`.

#### 4.7.7. Interpretacja pliku źródłowego przez $\text{\LaTeX}$ a i $\text{pdf\LaTeX}$ a

W idealnych warunkach dokument powinien się równie dobrze kompilować tak w  $\text{\LaTeX}$ u, jak i w  $\text{pdf\LaTeX}$ u. Problemy w tym względzie mogą wynikać przede wszystkim z włączania grafiki. Prosty sposób jest *systematyczne opuszczanie* rozszerzeń nazw plików w poleceniu `\includegraphics`. Powoduje to wyszukiwanie plików w formacie właściwym dla sposobu kompilacji. Reszta sprowadza się do wygenerowania odpowiednich wersji plików graficznych.  $\text{\LaTeX}$  poszuka plików `.eps`, podczas gdy  $\text{pdf\LaTeX}$  spróbuje włączyć plik o jednym z rozszerzeń `.png`, `.pdf`, `.jpg` lub `.mps` (w podanej kolejności).

Jeśli w  $\text{pdf\TeX}$ owej wersji dokumentu chcemy użyć innego kodu niż w  $\text{\LaTeX}$ owej, to do preambuły możemy dodać pakiet `ifpdf`<sup>10</sup>. W pakiecie zdefiniowana jest instrukcja `\ifpdf`, która ułatwia pisanie kodu warunkowego. W poniższym przykładzie wersja do druku, w formacie PS, ma być czarno-biała, co obniży koszty w drukarni, podczas gdy wersja PDF, przeznaczona do wyświetlania na ekranie, ma być kolorowa:

```
%& --translate-file=il2-pl
\RequirePackage{ifpdf}
% Sprawdzamy, czy działa pdfTeX:
\ifpdf \documentclass[a4paper,12pt,pdftex]{book}
\else \documentclass[a4paper,12pt,dvips]{book} \fi

\ifpdf \usepackage{lmodern} \fi
% dodajemy hiperłącza
\usepackage[bookmarks,colorlinks,plainpages=false]{hyperref}
\usepackage{polski}
...
```

Pakiet `hyperref` jest tu włączany także w wersji PS. Dzięki temu polecenie `\href` zadziała także w wypadku kompilacji zwykłym  $\text{\LaTeX}$ em.

Warto odnotować, że w niektórych współczesnych dystrybucjach  $\text{\TeX}$ a podstawowym programem używanym do kompilacji dokumentów jest właśnie `pdftex`. Przełącza się on na generowanie plików PDF albo DVI, zależnie

<sup>10</sup>Pełniejsze wyjaśnienie, kiedy ten pakiet się przydaje, można znaleźć w  $\text{\TeX}$ owym FAQ-u, w punkcie <http://www.tex.ac.uk/cgi-bin/texfaq2html?label=ifpdf>.

od ustawień podanych wraz z klasą dokumentu. Kod podany powyżej pozwala użyć zarówno instrukcji `pdflatex` do wygenerowania PDF-a, jak i `latex` do uzyskania pliku DVI.

#### 4.7.8. Wymiary kartki papieru

W `LATEX`u rozmiar papieru zadaje się w opcjonalnym argumencie polecenia `\documentclass`, na przykład `a4paper` albo `letterpaper`. Opcje te działają również w wypadku generowania dokumentu PDF, tyle że ich wykonanie nie wystarcza do stworzenia poprawnego pliku PDF. Jeśli korzystamy z pakietu `hyperref`, to niezbędne polecenia zostaną wykonane automatycznie. W przeciwnym razie musimy to zrobić ręcznie, umieszczając w preambule dokumentu następujący fragment:

```
\ifpdf %% działa tylko z pdftex:
  \pdfpagewidth=\paperwidth \pdfpageheight=\paperheight
  %% działa z programami latex+dvips:
\else \special{papersize=\paperwidth,\pageheight} \fi
```

### 4.8. Tworzenie prezentacji

Daniel Flipo <Daniel.Flipo@univ-lille1.fr>

Wyniki naszej pracy naukowej możemy przedstawiać kredą na tablicy, za pomocą rzutnika i przeźroczy (slajdów) bądź – posługując się odpowiednim oprogramowaniem – bezpośrednio z laptopa.

Program `pdfLATEX` w połączeniu z klasą `beamer` służy do tworzenia prezentacji w formacie PDF. Wyglądają one tak, jak gdybyśmy je wygenerowali – mając dobry dzień i sporo szczęścia – za pomocą PowerPointa, ale są bardziej przenośne, bo Acrobat Reader jest dostępny w większej liczbie systemów.

Klasa `beamer` używa pakietów `graphicx`, `color` oraz `hyperref` z opcjami zaadaptowanymi do prezentacji ekranowych.

Kompilując `pdfLATEX`em kod z rysunku 4.2, otrzymamy plik PDF ze stroną tytułową oraz jeszcze jedną stronę, zawierającą kilka punktów, które mają się odsłaniać w miarę, jak podczas prezentacji będziemy przechodzili do kolejnych jej kroków.

Jedną z zalet klasy `beamer` jest to, że generuje ona gotowy do użycia plik PDF, bez konieczności przechodzenia przez fazę postscriptową, jak to jest w wypadku pakietu `prosper` albo wymagającego dodatkowego przetworzenia pakietu `ppower4`. Korzystając z klasy `beamer`, możemy z tego samego pliku źródłowego generować kilka wersji prezentacji, tak zwanych trybów. Plik źródłowy może w nawiasach kątowych zawierać instrukcje przeznaczone do różnych trybów. Dostępne są następujące tryby: `beamer` – dla omówionych wyżej prezentacji PDF, `trans` – do slajdów, oraz `handout` – do wydruku. Domyślnym jest `beamer`, a inny tryb możemy zadać jako opcję

```
%& --translate-file=il2-pl
\documentclass[10pt]{beamer}
\mode<beamer>{\usetheme[hideothersubsections,%
    right,width=22mm]{Goettingen} }
\usepackage{polski}
\title{Prosta prezentacja}
\author[K. Wielki]{Karol Wielki}
\institute{Uniwersytet Karola Wielkiego}
%\titlegraphic{\includegraphics[width=20mm]{USTL}}
\date{2005}

\begin{document}
\begin{frame}<handout:0>
    \titlepage \end{frame}

\section{Przykład}

\begin{frame}
    \frametitle{Co robić w niedzielne popołudnie?}
    \begin{block}{Można:}
        \begin{itemize}
            \item pójść na spacer z psem\pause
            \item przeczytać książkę\pause
            \item pobawić się z kotem\pause
        \end{itemize}
    \end{block}
    i robić wiele innych rzeczy.
\end{frame}
\end{document}
```

Rysunek 4.2: Prosty kod dla klasy beamer

globalną, wpisując na przykład `\documentclass[10pt,handout]{beamer}` w celu wydrukowania materiałów do rozdania.

Wygląd ekranu prezentacji zależy od wybranego *tematu*. Możemy wskazać jeden z tematów dostarczanych wraz z klasą albo stworzyć nasz własny. Więcej informacji na ten temat można znaleźć w opisie klasy zawartym w pliku `beameruserguide.pdf`.

Przyjrzyjmy się bliżej kodowi z rysunku 4.2. Do wersji ekranowej prezentacji wybrano dla trybu `\mode<beamer>` temat *Goettingen*, w którym spisowi treści towarzyszy panel do nawigacji. Opcje tematu pozwalają określić rozmiar panelu (w tym wypadku 22 mm) oraz jego pozycję (z prawej strony głównego tekstu). Opcja *hideothersubsections* nakazuje pokazywać tytuły punktów, jednak tylko te, które pochodzą z bieżącego rozdziału. Dla

trybów `\mode<trans>` i `\mode<handout>` nie zadano żadnej opcji; mają się ukazywać w swoim układzie domyślnym.

Polecenia `\title`, `\author`, `\institute`, oraz `\titlegraphic` określają zawartość strony tytułowej. Opcjonalne argumenty instrukcji `\title` i `\author` pozwalają podać specjalną wersję tytułu oraz autora do wyświetlenia w panelu tematu *Goettingen*. Tytuły oraz podtytuły na panelu są tworzone przez zwykłe polecenia `\section` i `\subsection`, umieszczone *poza* otoczeniem `frame`.

Małe ikony u dołu ekranu służą do nawigowania po dokumencie. Można ich ukazywanie się zablokować poleceniem:

```
\setbeamertemplate{navigation symbols}{}

```

Zawartość każdego slajdu oraz ekranu należy umieścić w otoczeniu `frame`. W nawiasach kątowych `< i >` można podać opcjonalny argument, który pozwala ukryć slajd w jednym z trybów prezentacyjnych. W powyższym przykładzie pierwsza strona nie ukaże się w trybie materiałów do rozdania, gdyż w argumencie otoczenia `frame` podano argument `<handout:0>`.

Warto zatytułować każdy ze slajdów, nie zaś jedynie slajd tytułowy. Zauważmy, że polecenia `\section` i `\subsection` nie służą do tytułowania slajdów; służy do tego instrukcja `\frametitle`. Gdyby potrzebny był podtytuł, to – jak pokazano w przykładzie – można użyć otoczenia `block`.

Użycie polecenia `\pause` w otoczeniu `itemize` pozwala rozwijać punkty jeden po drugim, w miarę postępów prezentacji. Dodatkowe efekty prezentacyjne można osiągnąć za pomocą instrukcji: `\only`, `\uncover`, `\alt` oraz `\temporal`. Do dalszego sterowania prezentacją służą dopuszczalne w wielu miejscach opcje, podawane w nawiasach kątowych.

Cokolwiek mówić, aby uzyskać pełny obraz wszystkich dostępnych parametrów, trzeba przeczytać dokumentację `beameruserguide.pdf` klasy `beamer`. Pakiet ten jest ciągle rozwijany, dlatego warto po nowości zajrzeć na stronę <http://latex-beamer.sourceforge.net/>.

## 4.9. Pakiet pdfscreen

Pakiet `pdfscreen`, opracowany przez C.V. Radhakrishnana, pozwala tworzyć dokumenty PDF „zorientowane ekranowo”, to znaczy takie, które będzie się wygodnie czytało z ekranu monitora (inne wymiary kolumny tekstu, większy krój pisma). Takie dokumenty mogą też zawierać różne elementy nawigacyjne przeznaczone do poruszania się po dokumencie. Efektowne przykłady dokumentów przygotowanych za pomocą pakietu `pdfscreen` można znaleźć pod adresem <http://www.tug.org.in/tutorial/>. Szczegółowy polski opis pakietu zawiera [24].

## Rozdział 5

# Tworzenie grafiki matematycznej

Większość ludzi używa  $\text{\LaTeX}$ a do składania tekstów. Ponieważ jednak podejście strukturalno-logiczne do tworzenia dokumentów jest tak wygodne,  $\text{\LaTeX}$  oferuje pewną – fakt, że obarczoną ograniczeniami – możliwość generowania grafiki z opisów tekstowych w pliku źródłowym. Co więcej, powstało sporo rozszerzeń  $\text{\LaTeX}$ a przełamujących wspomniane ograniczenia. W tym rozdziale dowiesz się o kilku z tych rozszerzeń.

### 5.1. Przegląd

Otoczenie `picture` pozwala programować rysunki bezpośrednio w  $\text{\LaTeX}$ u. Szczegółowy jego opis można znaleźć w [12]. Metoda ta ma jednak ograniczenia, które wynikają z tego, że zarówno nachylenia odcinków, jak i średnice okręgów można wybierać jedynie spośród niewielkiej liczby wartości. Z drugiej strony w wersji  $\text{\LaTeX} 2_{\epsilon}$  można w otoczeniu `picture` używać polecenia `\qbezier`, gdzie „q” oznacza drugiego stopnia (ang. *quadratic*). Wiele często używanych krzywych, jak: okręgi, elipsy albo krzywe łańcuchowe można – niekiedy z odrobiną wysiłku matematycznego – zadowalająco przybliżać krzywymi Béziera<sup>1</sup>. Jeśli na dodatek do generowania  $\text{\LaTeX}$ owych bloków `\qbezier` użyje się języka programowania, na przykład Javy, to otoczenie `picture` ujawni całkiem pokazną moc.

Chociaż programowanie rysunków w  $\text{\LaTeX}$ u wiąże się z ograniczeniami i jest niekiedy nużące, to jednak są powody, by z tej możliwości korzystać. Dokumenty wytworzone w ten sposób są „małe” – w sensie liczby bajtów zajmowanych przez wynikowe pliki PDF lub DVI – a na dodatek nie trzeba do nich wczytywać dodatkowych plików graficznych.

---

<sup>1</sup>W ciekawym artykule [10] B.L. Jackowski zwraca uwagę, że krzywe Béziera są krzywymi giętymi stopnia trzeciego, nie zaś – jak sugeruje nazewnictwo  $\text{\LaTeX}$ owe – stopnia drugiego.



Pakiety takie jak: `epic` i `eepic` (opisane na przykład w [6]) bądź `pstricks` pomagają pokonywać ograniczenia krępujące oryginalne otoczenie `picture` i znacznie rozszerzają graficzną moc  $\text{\LaTeX}$ a.

O ile pierwsze dwa z tych pakietów jedynie rozszerzają otoczenie `picture`, to pakiet `pstricks` zawiera własne otoczenie rysujące – `pspicture`. Moc pakietu `pstricks` bierze się z tego, że istotnie wykorzystuje on możliwości Postscriptu.

Na dodatek napisano wiele pakietów realizujących konkretne cele. Jednym z nich jest `Xy-pic`, opisany na końcu tego rozdziału. Znaczną liczbę tych pakietów opisano w [7] (nie należy mylić z [6]).

Spośród narzędzi graficznych związanych z  $\text{\LaTeX}$ em największe chyba możliwości posiada `METAPOST`, autorstwa Johna D. Hobby, będący wariantem programu `METAFONT`, którego autorem jest z kolei Donald E. Knuth. `METAPOST` jest wyposażony w solidny i matematycznie wyrafinowany język programowania `METAFONT`a. Inaczej jednak niż `METAFONT`, który generuje mapy bitowe, `METAPOST` generuje pliki Postscriptowe, które można importować do  $\text{\LaTeX}$ a. Za wprowadzenie może posłużyć [8], a za podręcznik – [21]. Dostępne jest także wprowadzenie do programu `METAPOST` w języku polskim [8]<sup>2</sup>.

Obszerne omówienie zagadnień związanych z wykorzystaniem grafiki (oraz fontów) w  $\text{\LaTeX}$ u i  $\text{\TeX}$ u można znaleźć w [9].

## 5.2. Otoczenie picture

Urs Oswald <osurs@bluewin.ch>

### 5.2.1. Podstawowe polecenia

Otoczenie `picture`<sup>3</sup> można tworzyć poleceniem:

```
\begin{picture}(x,y)...\end{picture}
```

lub

```
\begin{picture}(x,y)(x_0,y_0)...\end{picture}
```

Liczby  $x$ ,  $y$ ,  $x_0$ ,  $y_0$  odnoszą się do wielkości `\unitlength`, którą można zmieniać w dowolnym momencie (jednak nie wewnątrz otoczenia `picture`) poleceniem takim jak:

```
\setlength{\unitlength}{1.2cm}
```

<sup>2</sup>Nie ma co ukrywać, że posługiwanie się `METAPOST`em wymaga posiadania przynajmniej minimum umiejętności programistycznych. Dla użytkowników którzy nie programują przydatny może być program `MetaGraf`, który jest okienkową nakładką do programu `METAPOST`, por. <http://w3.mecanica.upm.es/metapost/>.

<sup>3</sup>Otoczenie `picture` działa w standardowym  $\text{\LaTeX}$  2 $\epsilon$ ; nie trzeba łączyć żadnych dodatkowych pakietów.

Wartością domyślną `\unitlength` jest 1pt. Pierwsza para,  $(x, y)$ , to wymiary rezerwowanego wewnątrz dokumentu prostokątnego obszaru na rysunek. Opcjonalna druga para,  $(x_0, y_0)$ , to współrzędne przypisane dolnemu lewemu narożnikowi zarezerwowanego prostokąta.

Większość poleceń rysujących ma jedną z dwóch postaci:

```
\put(x,y){object}
```

lub

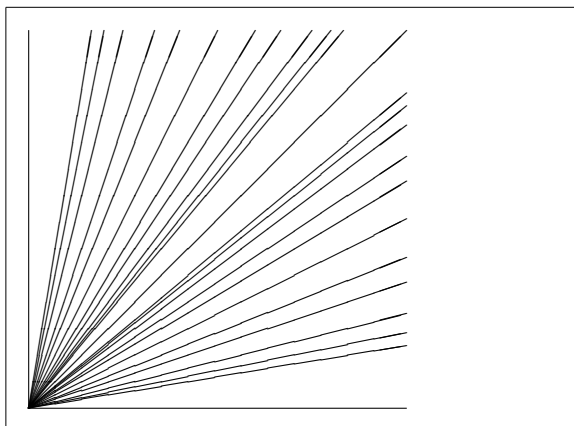
```
\multiput(x,y)(\Delta x,\Delta y){n}{object}
```

Wyjątkiem są krzywe Béziera, gdyż rysuje się je poleceniem:

```
\qbezier(x_1,y_1)(x_2,y_2)(x_3,y_3)
```

### 5.2.2. Odcinki

```
\setlength{\unitlength}{5cm}
\begin{picture}(1,1)
  \put(0,0){\line(0,1){1}}
  \put(0,0){\line(1,0){1}}
  \put(0,0){\line(1,1){1}}
  \put(0,0){\line(1,2){.5}}
  \put(0,0){\line(1,3){.3333}}
  \put(0,0){\line(1,4){.25}}
  \put(0,0){\line(1,5){.2}}
  \put(0,0){\line(1,6){.1667}}
  \put(0,0){\line(2,1){1}}
  \put(0,0){\line(2,3){.6667}}
  \put(0,0){\line(2,5){.4}}
  \put(0,0){\line(3,1){1}}
  \put(0,0){\line(3,2){1}}
  \put(0,0){\line(3,4){.75}}
  \put(0,0){\line(3,5){.6}}
  \put(0,0){\line(4,1){1}}
  \put(0,0){\line(4,3){1}}
  \put(0,0){\line(4,5){.8}}
  \put(0,0){\line(5,1){1}}
  \put(0,0){\line(5,2){1}}
  \put(0,0){\line(5,3){1}}
  \put(0,0){\line(5,4){1}}
  \put(0,0){\line(5,6){.8333}}
  \put(0,0){\line(6,1){1}}
  \put(0,0){\line(6,5){1}}
\end{picture}
```



Do rysowania odcinków służy polecenie:

```
\put(x,y){\line(x1,y1){length}}
```

Ma ono dwa argumenty: wektor kierunku i długość.

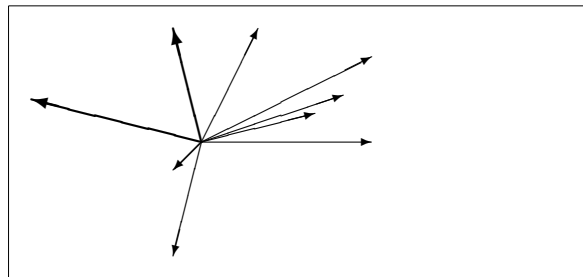
Jako składników wektora kierunku można użyć jedynie liczb całkowitych:

$$-6, -5, \dots, 5, 6,$$

Muszą one na dodatek być liczbami względnie pierwszymi (jedynym ich wspólnym dzielnikiem może być 1). Powyższy rysunek ilustruje wszystkie możliwe wartości nachylenia w pierwszej ćwiartce płaszczyzny. Długość jest podawana w jednostkach `\unitlength`. Argument długości oznacza współrzędną pionową w wypadku odcinka pionowego, zaś współrzędną poziomą we wszystkich pozostałych przypadkach.

### 5.2.3. Strzałki

```
\setlength{\unitlength}{0.75mm}
\begin{picture}(60,40)
  \put(30,20){\vector(1,0){30}}
  \put(30,20){\vector(4,1){20}}
  \put(30,20){\vector(3,1){25}}
  \put(30,20){\vector(2,1){30}}
  \put(30,20){\vector(1,2){10}}
  \thicklines
  \put(30,20){\vector(-4,1){30}}
  \put(30,20){\vector(-1,4){5}}
  \thinlines
  \put(30,20){\vector(-1,-1){5}}
  \put(30,20){\vector(-1,-4){5}}
\end{picture}
```



Strzałki rysuje się poleceniem:

```
\put(x,y){\vector(x1,y1){length}}
```

W wypadku strzałek na składniki wektora kierunku nałożone są jeszcze większe ograniczenia niż dla odcinków, bo jedynymi dopuszczalnymi liczbami całkowitymi są:

$$-4, -3, \dots, 3, 4.$$

Wartości składników i tu muszą być liczbami względnie pierwszymi (jedynym wspólnym dzielnikiem może być 1). Zauważ efekt działania polecenia `\thicklines` na dwie strzałki skierowane w stronę lewego górnego narożnika oraz `\thinlines` na strzałki w stronę narożnika prawego górnego.

### 5.2.4. Okręgi

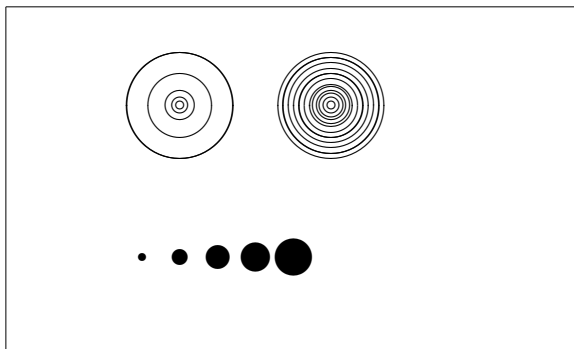
```

\setlength{\unitlength}{1mm}
\begin{picture}(60, 40)
  \put(20,30){\circle{1}}
  \put(20,30){\circle{2}}
  \put(20,30){\circle{4}}
  \put(20,30){\circle{8}}
  \put(20,30){\circle{16}}
  \put(20,30){\circle{32}}

  \put(40,30){\circle{1}}
  \put(40,30){\circle{2}}
  \put(40,30){\circle{3}}
  \put(40,30){\circle{4}}
  \put(40,30){\circle{5}}
  \put(40,30){\circle{6}}
  \put(40,30){\circle{7}}
  \put(40,30){\circle{8}}
  \put(40,30){\circle{9}}
  \put(40,30){\circle{10}}
  \put(40,30){\circle{11}}
  \put(40,30){\circle{12}}
  \put(40,30){\circle{13}}
  \put(40,30){\circle{14}}

  \put(15,10){\circle*{1}}
  \put(20,10){\circle*{2}}
  \put(25,10){\circle*{3}}
  \put(30,10){\circle*{4}}
  \put(35,10){\circle*{5}}
\end{picture}

```



Polecenie:

`\put( $x$ , $y$ ){\circle{ $diam$ }}`

rysuje okrąg o środku  $(x, y)$  i średnicy (nie promieniu) równym  $diam$ . Otoczenie `picture` dopuszcza średnice co najwyżej rzędu 14 mm, na dodatek nie wszystkie długości średnic są dozwolone. Polecenie `\circle*` generuje koła (wypełnione okręgi).

Podobnie jak w wypadku odcinków, można się uciec do dodatkowych pakietów, takich jak `eepic` bądź `pstricks`. Obszerny opis tych pakietów można znaleźć w [7].

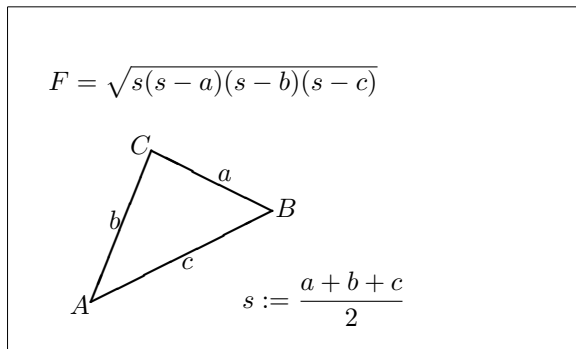
Otoczenie `picture` oferuje jeszcze jedną opcję. Jeśli nie boisz się wykonania niezbędnych obliczeń (być może nawet za pomocą programu), to okręgi oraz elipsy możesz złączać z krzywymi Béziera drugiego stopnia. Przykłady oraz źródłowe pliki w Javie można znaleźć w [21].

## 5.2.5. Tekst i wzory

```

\setlength{\unitlength}{0.8cm}
\begin{picture}(6,5)
  \thicklines
  \put(1,0.5){\line(2,1){3}}
  \put(4,2){\line(-2,1){2}}
  \put(2,3){\line(-2,-5){1}}
  \put(0.65,0.3){$A$}
  \put(4.05,1.9){$B$}
  \put(1.65,2.95){$C$}
  \put(3.1,2.5){$a$}
  \put(1.3,1.7){$b$}
  \put(2.5,1.05){$c$}
  \put(0.3,4){$F=$}
  \put(3.5,0.4){$\displaystyle$}
  \put(0.3,4){$\sqrt{s(s-a)(s-b)(s-c)}$}
  \put(3.5,0.4){$s:=\frac{a+b+c}{2}$}
\end{picture}

```



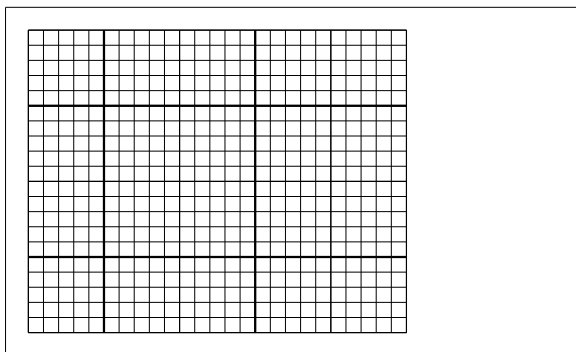
Jak widać w powyższym przykładzie, tekst oraz wzory można łatwo wprowadzać do otoczenia picture poleceniem `\put`.

5.2.6. Polecenia `\multiput` i `\linethickness`

```

\setlength{\unitlength}{2mm}
\begin{picture}(30,20)
  \linethickness{0.075mm}
  \multiput(0,0)(1,0){26}%
  {\line(0,1){20}}
  \multiput(0,0)(0,1){21}%
  {\line(1,0){25}}
  \linethickness{0.15mm}
  \multiput(0,0)(5,0){6}%
  {\line(0,1){20}}
  \multiput(0,0)(0,5){5}%
  {\line(1,0){25}}
  \linethickness{0.3mm}
  \multiput(5,0)(10,0){2}%
  {\line(0,1){20}}
  \multiput(0,5)(0,10){2}%
  {\line(1,0){25}}
\end{picture}

```



Polecenie:

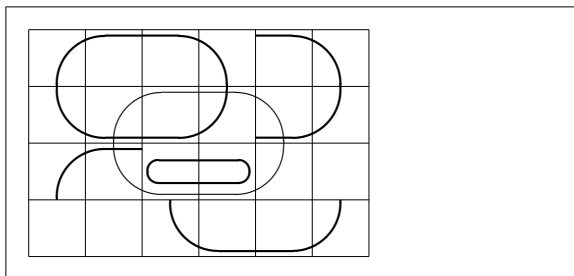
`\multiput(x,y)(\Delta x,\Delta y){n}{object}`

ma cztery argumenty: punkt początkowy, wektor przesunięcia z jednego obiektu do kolejnego, liczbę obiektów oraz rysowany obiekt. Instrukcja

`\linethickness` odnosi się do odcinków poziomych oraz pionowych, jednakże nie do odcinków ukośnych ani nie do okręgów. Stosuje się ona jednak również do krzywych Béziera drugiego stopnia.

### 5.2.7. Owale

```
\setlength{\unitlength}{0.75cm}
\begin{picture}(6,4)
  \linethickness{0.075mm}
  \multiput(0,0)(1,0){7}%
    {\line(0,1){4}}
  \multiput(0,0)(0,1){5}%
    {\line(1,0){6}}
  \thicklines
  \put(2,3){\oval(3,1.8)}
  \thinlines
  \put(3,2){\oval(3,1.8)}
  \thicklines
  \put(2,1){\oval(3,1.8)[t1]}
  \put(4,1){\oval(3,1.8)[b]}
  \put(4,3){\oval(3,1.8)[r]}
  \put(3,1.5){\oval(1.8,0.4)}
\end{picture}
```



Polecenie:

```
\put(x,y){\oval(w,h)}
```

jak też:

```
\put(x,y){\oval(w,h)[pozycja]}
```

generuje owal o środku  $(x, y)$ , szerokości  $w$  i wysokości  $h$ . Opcjonalne argumenty *pozycji*: **b**, **t**, **l** i **r** oznaczają odpowiednio: „top” (góra), „bottom” (dół), „left” (lewo) i „right” (pravo). Jak pokazuje przykład, można także używać ich kombinacji.

Grubość linii można sterować na dwa sposoby, z jednej strony poleceniem:

```
\linethickness{length}
```

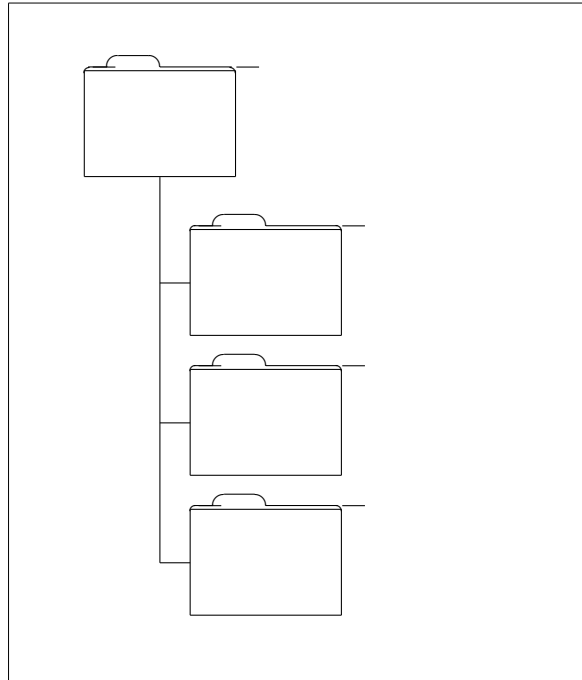
z drugiej – za pomocą `\thinlines` i `\thicklines`. O ile pierwszy ze sposobów odnosi się jedynie do linii poziomych oraz pionowych, a także do krzywych Béziera drugiego stopnia, o tyle `\thinlines` i `\thicklines` stosują się do odcinków ukośnych, jak też do okręgów i owali.

## 5.2.8. Wielokrotne użycie pudełek z rysunkami

```

\setlength{\unitlength}{0.5mm}
\begin{picture}(120,168)
\newsavebox{\foldera}
\savebox{\foldera}
  (40,32)[bl]{% definition
  \multiput(0,0)(0,28){2}
    {\line(1,0){40}}
  \multiput(0,0)(40,0){2}
    {\line(0,1){28}}
  \put(1,28){\oval(2,2)[tl]}
  \put(1,29){\line(1,0){5}}
  \put(9,29){\oval(6,6)[tl]}
  \put(9,32){\line(1,0){8}}
  \put(17,29){\oval(6,6)[tr]}
  \put(20,29){\line(1,0){19}}
  \put(39,28){\oval(2,2)[tr]}
}
\newsavebox{\folderb}
\savebox{\folderb}
  (40,32)[l]{% definition
  \put(0,14){\line(1,0){8}}
  \put(8,0){\usebox{\foldera}}
}
\put(34,26){\line(0,1){102}}
\put(14,128){\usebox{\foldera}}
\multiput(34,86)(0,-37){3}
  {\usebox{\folderb}}
\end{picture}

```



Pudełko rysunku można *zadeklarować* instrukcją:

```
\newsavebox{nazwa}
```

następnie *zachować* poleceniem:

```
\savebox{nazwa}(szerokość,wysokość)[pozycja]{treść}
```

i ostatecznie dowolnie często *rysować*, wywołując:

```
\put(x,y)\usebox{nazwa}
```

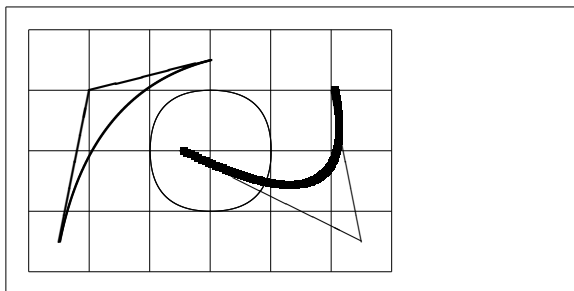
Opcjonalny parametr *pozycja* definiuje „punkt zaczepienia” zachowywanego pudełka (savebox). W przykładzie nadano mu wartość `bl`, co oznacza umieszczenie punktu zaczepienia w dolnym lewym narożniku pudełka. Pozycje można też oznaczać literami `t` (góra) i `r` (prawo).

Argument *nazwa* odnosi się do L<sup>A</sup>T<sub>E</sub>Xowego magazynu poleceń i dlatego ma naturę instrukcji (co w powyższym przykładzie objawia się choćby użyciem znaków '\'). Rysunki przechowywane w pudełkach można zagnieżdżać. W powyższym przykładzie wewnątrz definicji `\folderb` użyto `\foldera`.

Trzeba było użyć polecenia `\oval`, ponieważ instrukcja `\line` nie działa, gdy długość odcinka wynosi mniej niż około 3 mm.

### 5.2.9. Krzywe Béziera drugiego stopnia

```
\setlength{\unitlength}{0.8cm}
\begin{picture}(6,4)
  \linethickness{0.075mm}
  \multiput(0,0)(1,0){7}
    {\line(0,1){4}}
  \multiput(0,0)(0,1){5}
    {\line(1,0){6}}
  \thicklines
  \put(0.5,0.5){\line(1,5){0.5}}
  \put(1,3){\line(4,1){2}}
  \qbezier(0.5,0.5)(1,3)(3,3.5)
  \thinlines
  \put(2.5,2){\line(2,-1){3}}
  \put(5.5,0.5){\line(-1,5){0.5}}
  \linethickness{1mm}
  \qbezier(2.5,2)(5.5,0.5)(5,3)
  \thinlines
  \qbezier(4,2)(4,3)(3,3)
  \qbezier(3,3)(2,3)(2,2)
  \qbezier(2,2)(2,1)(3,1)
  \qbezier(3,1)(4,1)(4,2)
\end{picture}
```



Jak widać w przykładzie, podzielenie okręgu na cztery krzywe Béziera drugiego stopnia nie daje zadowalającego efektu; lepsze przybliżenie dałoby osiem. Przykład ponownie ilustruje wpływ instrukcji `\linethickness` na linie poziome i pionowe oraz poleceń `\thinlines` i `\thicklines` na odcinki pochyłe. Pokazuje on również, że oba te rodzaje poleceń oddziałują na krzywe Béziera i że kolejne użycie któregośkolwiek z nich przesłania poprzednie.

Niech symbole  $P_1 = (x_1, y_1)$ ,  $P_2 = (x_2, y_2)$  oznaczają punkty końcowe, zaś  $m_1, m_2$  – odpowiednie nachylenia krzywej Béziera drugiego stopnia. Pośredni punkt kontrolny  $S = (x, y)$  jest zatem opisany równaniami:

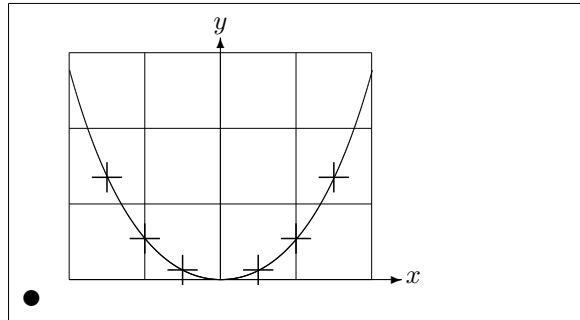
$$\begin{cases} x &= \frac{m_2 x_2 - m_1 x_1 - (y_2 - y_1)}{m_2 - m_1}, \\ y &= y_i + m_i(x - x_i) \quad (i = 1, 2). \end{cases} \quad (5.1)$$



W [21] można znaleźć program w Javie, który generuje odpowiednią linię polecenia `\qbezier`.

### 5.2.10. Krzywe łańcuchowe

```
\setlength{\unitlength}{1cm}
\begin{picture}(4.3,3.6)(-2.5,-0.25)
\put(-2,0){\vector(1,0){4.4}}
\put(2.45,-.05){\textit{x}}
\put(0,0){\vector(0,1){3.2}}
\put(0,3.35){\makebox(0,0){\textit{y}}}
\qbezier(0.0,0.0)(1.2384,0.0)
(2.0,2.7622)
\qbezier(0.0,0.0)(-1.2384,0.0)
(-2.0,2.7622)
\linethickness{.075mm}
\multiput(-2,0)(1,0){5}
{\line(0,1){3}}
\multiput(-2,0)(0,1){4}
{\line(1,0){4}}
\linethickness{.2mm}
\put(.3,.12763){\line(1,0){.4}}
\put(.5,-.07237){\line(0,1){.4}}
\put(-.7,.12763){\line(1,0){.4}}
\put(-.5,-.07237){\line(0,1){.4}}
\put(.8,.54308){\line(1,0){.4}}
\put(1,.34308){\line(0,1){.4}}
\put(-1.2,.54308){\line(1,0){.4}}
\put(-1,.34308){\line(0,1){.4}}
\put(1.3,1.35241){\line(1,0){.4}}
\put(1.5,1.15241){\line(0,1){.4}}
\put(-1.7,1.35241){\line(1,0){.4}}
\put(-1.5,1.15241){\line(0,1){.4}}
\put(-2.5,-0.25){\circle*{0.2}}
\end{picture}
```



Każdą z symetrycznych połówek wykresu cosinusa hiperbolicznego  $y = \cosh x - 1$  przybliżono na rysunku krzywą Béziera. Prawa połówka krzywej kończy się w punkcie  $(2, 2.7622)$ , w którym nachylenie ma wartość  $m = 3.6269$ . Używając ponownie równania (5.1), możemy wyliczyć pośrednie punkty kontrolne. Okazuje się, że są to:  $(1.2384, 0)$  i  $(-1.2384, 0)$ . Krzyżkami została zaznaczona „prawdziwa” krzywa. Błąd jest ledwie zauważalny, bo wynosi mniej niż jeden procent.

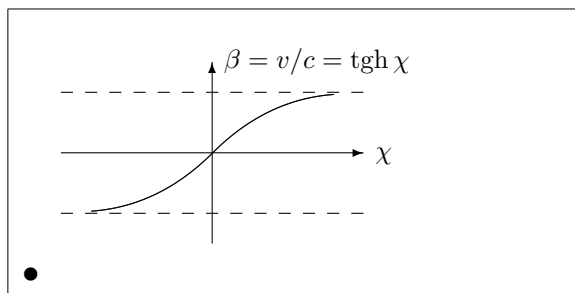
Ten przykład ilustruje też użycie opcjonalnego argumentu otoczenia `\begin{picture}`. Rysunek zdefiniowano w terminach wygodnych współrzędnych „matematycznych”, podczas gdy poleceniem:

```
\begin{picture}(4.3,3.6)(-2.5,-0.25)
```

jego lewemu dolnemu narożnikowi (oznaczonemu czarnym kółeczkiem) przypisano współrzędne  $(-2.5, -0.25)$ .

### 5.2.11. Prędkość w Szczególnej Teorii Względności

```
\setlength{\unitlength}{0.8cm}
\begin{picture}(6,4)(-3,-2)
\put(-2.5,0){\vector(1,0){5}}
\put(2.7,-0.1){\chi}
\put(0,-1.5){\vector(0,1){3}}
\multiput(-2.5,1)(0.4,0){13}
{\line(1,0){0.2}}
\multiput(-2.5,-1)(0.4,0){13}
{\line(1,0){0.2}}
\put(0.2,1.4)
{\beta=v/c=\tanh\chi}
\qbezier(0,0)(0.8853,0.8853)
(2,0.9640)
\qbezier(0,0)(-0.8853,-0.8853)
(-2,-0.9640)
\put(-3,-2){\circle*{0.2}}
\end{picture}
```



Punkty kontrolne dwóch krzywych Béziera wyliczono ze wzorów (5.1). Gałąź dodatnia jest określona wartościami  $P_1 = (0, 0)$ ,  $m_1 = 1$  oraz  $P_2 = (2, \tanh 2)$ ,  $m_2 = 1/\cosh^2 2$ . I znowu rysunek wyrażono w wygodnych matematycznie współrzędnych, a lewemu dolnemu narożnikowi przypisano współrzędne  $(-3, -2)$  (czarne kółeczko).

## 5.3. Xy-pic

Alberto Manuel Brandão Simões <albie@alfarrabio.di.uminho.pt>

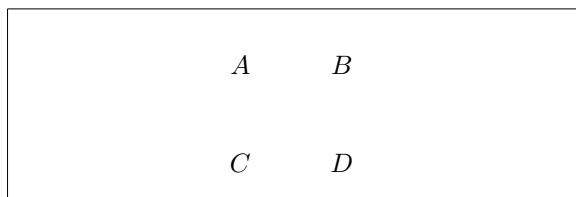
Pakiet Xy-pic służy do rysowania grafów. Aby uzyskać dostęp do jego funkcji, umieszczamy w preambule dokumentu wiersz:

```
\usepackage[opcje]{xy}
```

Parametr *opcje* jest listą funkcji pakietu Xy-pic, które mają zostać załadowane. Opcje te przydają się między innymi do szukania błędów w pakiecie. Zaleca się przekazywać opcję `all`, nakazującą L<sup>A</sup>T<sub>E</sub>Xowi załadować wszystkie polecenia pakietu.

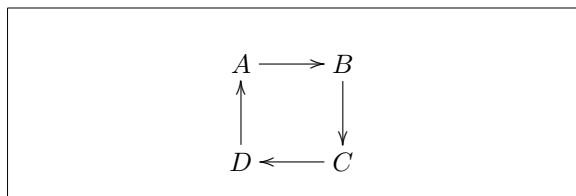
Graf rysuje się w Xy-pic na płótnie o strukturze macierzy, a każdy element grafu jest umieszczony w komórce tej macierzy:

```
\begin{displaymath}
\mathrm{xymatrix}{A & B \\
C & D }
\end{displaymath}
```



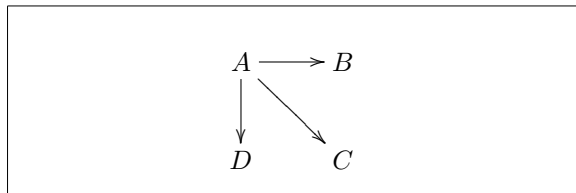
Polecenia `\xymatrix` można użyć jedynie w trybie matematycznym. Powyżej zadano dwa wiersze i dwie kolumny. Aby przekształcić tę macierz w graf, posługujemy się dodającym strzałki poleceniem `\ar`<sup>4</sup>:

```
\begin{displaymath}
\mathrm{xymatrix}{ A \ar[r] & B \ar[d] \\
D \ar[u] & C \ar[l] }
\end{displaymath}
```



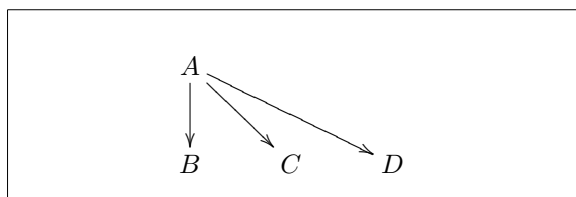
Polecenie `\ar` umieszczamy w komórce, w której strzałka ma się zaczynać. Argument oznacza kierunek strzałki (u – góra, d – dół, r – prawo i l – lewo):

```
\begin{displaymath}
\mathrm{xymatrix}{
A \ar[d] \ar[dr] \ar[r] & B \\
D & C }
\end{displaymath}
```



Aby uzyskać przekątne, używamy po prostu więcej niż jednego kierunku. Powtarzając kierunek, wydłużamy w istocie strzałkę:

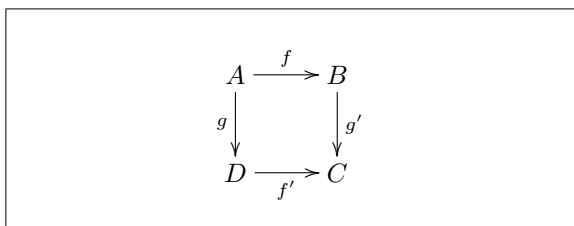
```
\begin{displaymath}
\mathrm{xymatrix}{
A \ar[d] \ar[dr] \ar[dr] & & \\
B & & C & D }
\end{displaymath}
```



Uzupełniając strzałki o etykiety, możemy uzyskać jeszcze ciekawsze grafy. Używamy do tego zwyczajnych operatorów indeksów górnych i dolnych:

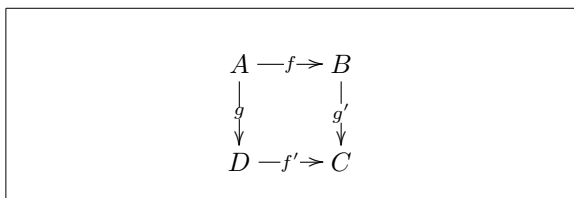
<sup>4</sup>Pakiety `Xy-pic` oraz `polski` są w konflikcie, bo oba definiują polecenie `\ar`. Jeżeli polecenie `\ar` ma działać tak, jak zdefiniowano to w pakiecie `Xy-pic`, to trzeba `xy` dołączyć po pakiecie `polski`. Oczywiście nie ma wtedy dostępu do polecenia `\ar` z pakietu `polski`.

```
\begin{displaymath}
\xymatrix{
  A \ar[r]^f \ar[d]_g &
  B \ar[d]^{g'} \\
  D \ar[r]_{f'} & C
}
\end{displaymath}
```



Jak widać, operatorów tych używa się tak samo jak w trybie matematycznym. Jedyną różnicą jest to, że indeks górny oznacza *nad*, zaś dolny – *pod* strzałką. Istnieje jeszcze trzeci operator – pionowej kreski |. Umieszcza on tekst *na* strzałce:

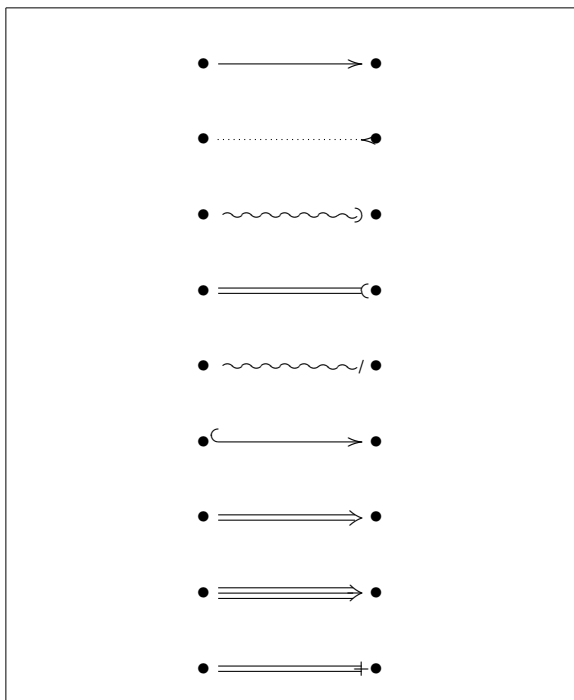
```
\begin{displaymath}
\xymatrix{
  A \ar[r]|f \ar[d]|g &
  B \ar[d]|{g'} \\
  D \ar[r]|{f'} & C
}
\end{displaymath}
```



Aby narysować strzałkę z dziurką w środku, możemy użyć polecenia `\ar[...]|hole`.

W niektórych sytuacjach trzeba używać różnych typów strzałek. Można je w tym celu oznaczać różnymi etykietami bądź nadawać im różny wygląd:

```
\begin{displaymath}
\xymatrix{
\bullet \ar@{->}[rr] && \bullet \\
\bullet \ar@{.<}[rr] && \bullet \\
\bullet \ar@{~}[rr] && \bullet \\
\bullet \ar@{=(}[rr] && \bullet \\
\bullet \ar@{~/}[rr] && \bullet \\
\bullet \ar@{^{\{ }->}[rr] && \bullet \\
\bullet \ar@{2->}[rr] && \bullet \\
\bullet \ar@{3->}[rr] && \bullet \\
\bullet \ar@{=+}[rr] && \bullet
}
\end{displaymath}
```

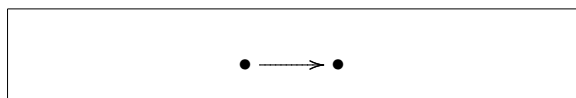


Zauważ różnicę między następującymi dwoma grafami:

```

\begin{displaymath}
\begin{matrix}
\bullet & \longrightarrow & \bullet \\
\bullet & \longrightarrow & \bullet
\end{matrix}
\end{displaymath}

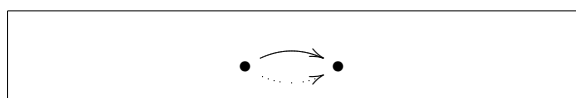
```



```

\begin{displaymath}
\begin{matrix}
\bullet & \curvearrowright & \bullet \\
\bullet & \curvearrowright & \bullet
\end{matrix}
\end{displaymath}

```



Symbole między ukośnikami określają tu sposób rysowania krzywych.

Pakiet  $\text{\texttt{Xy-pic}}$  oferuje wiele metod wpływania na sposób rysowania krzywych. Więcej na ten temat można przeczytać w dokumentacji pakietu [23].

## Rozdział 6

# Adaptowanie L<sup>A</sup>T<sub>E</sub>Xa

*Dokumenty składane za pomocą poznanych do tej pory poleceń będą się zapewne podobały zdecydowanej większości czytelników. Chociaż ich wygląd nie będzie może wyrafinowany, z pewnością jednak spełnią one podstawowe zasady składu, dzięki czemu będzie się je czytało łatwo i przyjemnie.*


*W niektórych wypadkach może się jednak okazać, że brakuje polecenia czy otoczenia, za pomocą którego moglibyśmy złożyć dany fragment tekstu w sposób odpowiadający potrzebom, albo też że sposób działania dostępnej w L<sup>A</sup>T<sub>E</sub>Xu instrukcji nie spełnia naszych wymagań.*

*W tym rozdziale przedstawimy, jak nauczyć L<sup>A</sup>T<sub>E</sub>Xa formatować dokumenty tak, aby wyglądały inaczej niż w wypadku korzystania jedynie ze standardowych klas i pakietów.*

### 6.1. Definiowane instrukcji i otoczeń

Czytelnicy zauważyli zapewne, że nowo wprowadzane w tej książce polecenia ukazują się w ramkach oraz że znajdują się one w skorowidzu. Aby to osiągnąć, nie korzystaliśmy za każdym razem z wbudowanych w L<sup>A</sup>T<sub>E</sub>Xa instrukcji, lecz utworzyliśmy własny pakiet, w którym zawarliśmy nowe, potrzebne nam polecenia i otoczenia. Dysponując takim pakietem, wystarczy po prostu napisać:

```
\begin{command}  
\ci{polecenie}  
\end{command}
```



`\polecenie`

W tym przykładzie użyliśmy zarówno nowego otoczenia o nazwie `command`, odpowiedzialnego za rysowanie ramek dookoła instrukcji, jak też nowego polecenia `\ci`, służącego do składu nazw poleceń i wprowadzania ich do skorowidza. Proponujemy Czytelnikom odszukanie hasła `\polecenie` w skorowidzu; przy hasle powinny być podane numery stron, na których ta instrukcja występuje w książce.

Gdy zdecydujemy się zaprzestać otaczać polecenia ramkami, to do zmiany wyglądu wystarczy, że zmienimy definicję otoczenia `command`. Jest to znacznie łatwiejsze od przebiegnięcia przez cały dokument w celu wyłapania w nim wszystkich standardowych poleceń  $\text{\LaTeX}$ , które służą do rysowania ramek wokół słów.

We wstępie do tego opracowania wspomnieliśmy, że w  $\text{\LaTeX}$ u możemy się skupić na logicznej strukturze dokumentów. Wskazane jest rozróżnić w tekście źródłowym wszystkie elementy logiczne dokumentu, nawet jeżeli ich formatowanie jest identyczne. Nierzadko bowiem to, co dzisiaj formatujemy w taki sam sposób, w przyszłości możemy chcieć rozróżnić.

Zwykło się na przykład składać adresy internetowe imitacją kroju maszynowego. Ponieważ w adresach mogą wystąpić znaki specjalne  $\text{\LaTeX}$ a, to można by do tego celu stosować instrukcję `\verb`. Lepiej jednak użyć specjalnej instrukcji, np. `\url`. W dokumencie papierowym nie ma to znaczenia, znakowanie logiczne pozwala jednak przedstawić dokument zarówno w formie drukowanej, jak i hipertekstowej, w formacie HTML lub PDF.

### 6.1.1. Instrukcje definiowane przez użytkownika

Do definiowania potrzebnych nam nowych poleceń możemy użyć instrukcji:

```
\newcommand{nazwa}[num]{tekst}
```

Wymaga ona podania dwóch argumentów. Pierwszy z nich, *nazwa*, oznacza nazwę nowej instrukcji, natomiast *tekst* to jej znaczenie, czyli tekst, który ma zostać wstawiony do składu w momencie wykonania instrukcji. Podawany w nawiasach kwadratowych argument *num* powinien być cyfrą od 1 do 9, określającą liczbę (obowiązkowych) argumentów instrukcji. Argument *num* jest opcjonalny, a jego pominięcie oznacza, że definiowana instrukcja jest bezargumentowa.

W części *tekst* wolno używać zarówno standardowych instrukcji  $\text{\LaTeX}$ a, jak też zdefiniowanych przez użytkownika. Nie wolno jednak korzystać z tych instrukcji, które same definiują inne polecenia, jak `\newcommand`, `\newenvironment` itp. Niedozwolona jest rekursja, nie wolno też w nazwach instrukcji umieszczać polskich liter diakrytycznych.

Następujące przykłady pomogą lepiej zrozumieć zagadnienie. W pierwszym z nich definiujemy instrukcję o nazwie `\kwle`, mającą być skrótem dla słów „Krótkie wprowadzenie do systemu  $\text{\LaTeX} 2_{\epsilon}$ ”. Takie polecenie mogłoby się przydać, gdyby tytuł książki miał w niej występować wielokrotnie.

```
\newcommand{\kwle}{Krótkie
wprowadzenie do systemu \LaTeXe}
% następnie po \begin{document}:
\kwle; \emph{\kwle}
```

```
Krótkie wprowadzenie do systemu \LaTeX 2ε;
Krótkie wprowadzenie do systemu \LaTeX 2ε
```

Następny przykład ilustruje sposób wykorzystania opcjonalnego argumentu *num*. Znacznik #1 oznacza pierwszy parametr formalny (#2 oznaczałby drugi, #3 – trzeci itd.). W trakcie wykonywania treści instrukcji w miejsce parametrów formalnych T<sub>E</sub>X wstawia argumenty podane w jej wywołaniu (parametry aktualne).

Instrukcja w poniższym przykładzie ma jeden parametr:

```
\newcommand{\wle}[1]
{\emph{#1} wprowadzenie
do systemu \LaTeXe}
% następnie po \begin{document}:
\wle{Krótkie}; \wle{Długie}
```

*Krótkie* wprowadzenie do systemu L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>;  
*Długie* wprowadzenie do systemu L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>

L<sup>A</sup>T<sub>E</sub>X nie pozwala zdefiniować instrukcji nazwanej tak samo jak wcześniej zdefiniowane polecenie. W wypadku gdy chcemy zmienić znaczenie już istniejącej instrukcji, powinniśmy użyć polecenia `\renewcommand`. Za wyjątkiem nazwy ma ono składnię identyczną jak `\newcommand`.

Czasami może się też przydać polecenie `\providecommand`. Działa ono jak `\newcommand`, z tym że jeśli istnieje już komenda o takiej samej nazwie, to nie zastępuje ono starej, zwyczajnie ignorując nową.

Nieco uwagi należy poświęcić temu, jaki skutek ma umieszczenie odstępów po komendzie L<sup>A</sup>T<sub>E</sub>Xa. Informacje na ten temat można znaleźć na stronie 5.

### 6.1.2. Otoczenia definiowane przez użytkownika

Odpowiednikiem definiującej nowe polecenie instrukcji `\newcommand` jest dla otoczeń instrukcja `\newenvironment`. Ma ona następującą składnię:

```
\newenvironment{nazwa}[num]{początek}{koniec}
```

Podobnie jak w wypadku `\newcommand`, można instrukcji `\newenvironment` użyć z argumentem opcjonalnym albo bez niego. L<sup>A</sup>T<sub>E</sub>X wstawia tekst *początek*, gdy w dokumencie napotyka napis `\begin{nazwa}`, a zawartość argumentu *koniec* – po napotkaniu napisu `\end{nazwa}`.

Poniższy przykład ilustruje sposób użycia instrukcji `\newenvironment`.

```
\newenvironment{zrodlo}
{Źródło: }{\par}
% następnie po \begin{document}:
\begin{zrodlo} Rocznik Statystyczny
GUS. \end{zrodlo}
```

Źródło: Rocznik Statystyczny GUS.

Znaczenie argumentu *num* jest takie samo jak w instrukcji `\newcommand`.

L<sup>A</sup>T<sub>E</sub>X nie pozwala zdefiniować otoczenia o już istniejącej nazwie. W razie potrzeby zastąpienia już istniejącego otoczenia powinniśmy użyć polecenia `\renewenvironment`, o składni takiej samej jak `\newenvironment`.



### 6.1.3. Nadmiarowe odstępy

Gdy tworzymy nowe otoczenie, problemem mogą być zbędne odstępy, które L<sup>A</sup>T<sub>E</sub>X wstawia do składu. Rozważmy przykład otoczenia, które ma się rozpoczynać od akapitu bez wcięcia, ponadto pierwszy akapit po otoczeniu także nie ma mieć wcięcia. Polecenie `\ignorespaces`, umieszczone jako ostatnie polecenie bloku *begin* otoczenia, spowoduje zignorowanie wszystkich odstępow występujących przed pierwszym akapitem otoczenia. Usunięcie drugiego wcięcia przez umieszczenie w bloku końcowym `\ignorespaces` jest niemożliwe, gdyż zawsze ostatnim poleceniem będzie `\end{otoczenie}`, które anuluje działanie `\ignorespaces`. W takiej sytuacji trzeba skorzystać z polecenia `\ignorespacesafterend`. Napotkawszy je, L<sup>A</sup>T<sub>E</sub>X wstawi `\ignorespaces` dopiero po wykonaniu zamykającego `\end{otoczenie}`.

```
\newenvironment{proste}%
{\noindent}%
{\par\noindent}

\begin{proste}
Zobacz odstępy\\z lewej strony.
\end{proste}
Tak samo\\tutaj.
```

Zobacz odstępy  
z lewej strony.  
Tak samo  
tutaj.

```
\newenvironment{poprawne}%
{\noindent\ignorespaces}%
{\par\noindent%
\ignorespacesafterend}

\begin{poprawne}
Bez odstępu\\z lewej strony.
\end{poprawne}
Tak samo\\tutaj.
```

Bez odstępu  
z lewej strony.  
Tak samo  
tutaj.

### 6.1.4. Własne pakiety

W wypadku definiowania wielu nowych poleceń i otoczeń preambuła dokumentu może się znacznie wydłużyć. Dobrze w takiej sytuacji stworzyć pakiet zawierający definicje tych instrukcji i otoczeń. Taki pakiet można później dołączyć do dokumentu poleceniem `\usepackage`.

Tworzenie pakietu polega na skopiowaniu poleceń z preambuły do oddzielnego pliku o rozszerzeniu `.sty`. Na początku pakietu należy wpisać polecenie:

```
\ProvidesPackage{nazwa}
```

Dzięki instrukcji `\ProvidesPackage` L<sup>A</sup>T<sub>E</sub>X poznaje nazwę pakietu, a to pozwala mu na przykład ostrzec użytkownika w wypadku powtórnego do-

---

```
% Przykładowy pakiet ***
\ProvidesPackage{demopack}
\newcommand{\kwle}{Krótkie wprowadzenie do systemu LATEX}
\newcommand{\wle}[1]{\emph{#1} wprowadzenie
do systemu LATEX}
\newenvironment{zrodlo}{Źródło: }{\par}
```

---

Rysunek 6.1: Przykładowy pakiet

łączenia pakietu do dokumentu. Rysunek 6.1 przedstawia niewielki pakiet z instrukcjami z powyższych przykładów.

## 6.2. Fonty

### 6.2.1. Instrukcje przełączające stopień pisma

L<sup>A</sup>T<sub>E</sub>X automatycznie dobiera krój, odmianę i stopień pisma<sup>1</sup> dla różnych elementów dokumentu (tytułów rozdziałów, punktów, przypisów itp.). Czasami zachodzi jednak potrzeba „ręcznego” przełączenia kroju bądź stopnia pisma. Można do tego użyć poleceń zestawionych w tabelach 6.1 i 6.2. Stopień pisma jest kwestią układu graficznego dokumentu i zależy od wybranej klasy dokumentu oraz ustawienia odpowiednich opcji. W tabeli 6.3 zestawiono stopnie pisma w jednostkach absolutnych dla poleceń zmieniających wielkość kroju w standardowych klasach dokumentów.

```
{\small Nieliczni lecz
\textbf{odważni} Rzymianie rządili}
{\Large wielką \textit{Italia}}.}
```

Nieliczni lecz <b>odważni</b> Rzymianie rządili wielką <i>Italia</i> .
---

Bieżący font jest w L<sup>A</sup>T<sub>E</sub>Xu scharakteryzowany przez pięć elementów: układ (zestaw znaków), krój (rodzinę), grubość i szerokość, odmianę oraz stopień

---

<sup>1</sup>*Pismo drukarskie* to pismo utrwalone na nośniku, tj. materiale, na którym umieszczono negatywy lub pozytywy znaków pisma. *Krój pisma* to obraz pisma drukarskiego o jednolitych cechach graficznych, niezależnych od stopnia i odmiany pisma. Każdy krój pisma posiada swoją nazwę (na przykład Times New Roman, Computer Modern czy Garamond). *Odmiana kroju* pisma różnicuje pisma jednego kroju ze względu na grubość, szerokość i pochylenie. *Stopień pisma* określa z kolei wielkość znaków. Zestawy metalowych *czcionek*, czyli kawałków metalu, w których utrwalano znaki pisma, drukarze przechowywali w *kasztach*. Angielską, a właściwie amerykańską nazwą zestawu czcionek jednego kroju i wielkości, powszechnie dziś używaną w terminologii komputerowej, jest *font* (zobacz też [3]).

Fonty L<sup>A</sup>T<sub>E</sub>Xa, takie jak PL, EC czy LM, są optycznie identyczne, bo wszystkie są replikami kroju *Computer Modern*, różnią się jednak od najczęściej używanego w edytorach biurowych, takich jak MS Word, kroju *Times New Roman*.

i interlinię. Każdy z nich można dobrać niezależnie od ustawienia pozostałych. Oznacza to na przykład, że zmiana stopnia pisma nie powoduje zmiany jego kroju ani odmiany.

Tabela 6.1: Polecenia wyboru krojów i odmian

<code>\textrm{...}</code>	krój szeryfowy	<code>\textsf{...}</code>	krój bezszeryfowy
<code>\texttt{...}</code>	grotesk, tj. pismo o jednakowej szerokości znaków		
<code>\textmd{...}</code>	pismo jasne	<code>\textbf{...}</code>	<b>pismo grube</b>
<code>\textup{...}</code>	odmiana prosta	<code>\textit{...}</code>	<i>kursywa</i>
<code>\textsl{...}</code>	<i>odmiana pochyła</i>	<code>\textsc{...}</code>	KAPITALIKI
<code>\emph{...}</code>	<i>wyróżnienie</i>		
<code>\textnormal{...}</code>	główny font dokumentu		

Tabela 6.2: Polecenia jednoczesnego wyboru stopnia pisma i interlinii

<code>\tiny</code>	mikroskopijny	<code>\Large</code>	wiekszy
<code>\scriptsize</code>	bardzo mały	<code>\LARGE</code>	bardzo duży
<code>\footnotesize</code>	mniejszy	<code>\huge</code>	ogromny
<code>\small</code>	mały	<code>\Huge</code>	największy
<code>\normalsize</code>	normalny		
<code>\large</code>	duży		

Przy okazji omawiania poleceń dotyczących fontów trzeba wspomnieć o koncepcji *grupowania*. Grupa zaczyna się od znaku {, a kończy znakiem }. Grupy służą do ograniczania zasięgu działania poleceń L<sup>A</sup>T<sub>E</sub>Xa. Przyjrzyjmy się następującemu przykładowi:

Lubię {\LARGE duże oraz  
{\small małe} litery} i~cyfry.

Lubię duże oraz małe litery i cyfry.

Pierwszy nawias klamrowy rozpoczyna grupę, potem polecenie `\LARGE` zmienia stopień pisma na *bardzo duży*, w którym zostanie złożony napis „duże oraz”. Kolejny otwierający nawias klamrowy zaczyna następną grupę. W jej obrębie polecenie `\small` zmienia stopień pisma na mały. Do złożenia w tym stopniu przewidziano jedynie słowo „małe”, bo nawias } za tym słowem zamyka grupę. Po zamknięciu grupy następuje powrót do stopnia pisma aktualnego przed jej rozpoczęciem, czyli `\LARGE`. W nim zostanie złożone słowo „litera”. Zamknięcie tej grupy powoduje, że resztę tekstu L<sup>A</sup>T<sub>E</sub>X

Tabela 6.3: Wielkość stopnia pisma w klasach standardowych

<i>Stopień</i>	<i>10pt</i>	<i>opcja 11pt</i>	<i>opcja 12pt</i>
<code>\tiny</code>	5pt	6pt	6pt
<code>\scriptsize</code>	7pt	8pt	8pt
<code>\footnotesize</code>	8pt	9pt	10pt
<code>\small</code>	9pt	10pt	11pt
<code>\normalsize</code>	10pt	11pt	12pt
<code>\large</code>	12pt	12pt	14pt
<code>\Large</code>	14pt	14pt	17pt
<code>\LARGE</code>	17pt	17pt	20pt
<code>\huge</code>	20pt	20pt	25pt
<code>\Huge</code>	25pt	25pt	25pt

złoży w wyjściowym stopniu pisma. Jak widać, grupy można zagnieżdżać, nawet wielokrotnie.

Polecenia zmieniające stopień pisma zmieniają także interlinię. Dzieje się tak jednak tylko wtedy, gdy *przed* zamknięciem odpowiedniej grupy kończony jest akapit – przez wstawienie pustego wiersza *lub* polecenia `\par`. Zwróćmy uwagę na miejsce, w którym umieszczono instrukcję `\par` w poniższych dwóch przykładach.

```
{\Large Zdanie, które ma więcej  
niż pięć słów, nie ma sensu!}\par}
```

Zdanie, które ma więcej niż pięć  
słów, nie ma sensu!

```
{\Large Zdanie, które ma więcej  
niż pięć słów, nie ma sensu!}\par}
```

Zdanie, które ma więcej niż pięć  
słów, nie ma sensu!

Jeśli zachodzi konieczność zmiany stopnia pisma dla całego akapitu lub jeszcze dłuższego tekstu, to możemy skorzystać ze składni przyjętej dla otoczeń:

```
\begin{Large} Zdanie, które ma  
więcej niż pięć słów, nie ma sensu!  
\end{Large}
```

Zdanie, które ma więcej niż pięć  
słów, nie ma sensu!

Zapis taki pozwala unikać łatwych do popełnienia błędów, wynikających z opuszczania nawiasów otwierających lub zamykających grupy.

W *trybie matematycznym* w celu złożenia fragmentu wzoru innym niż pochyłe krojem pisma można zastosować polecenia zestawione w tabeli 6.4.

Tabela 6.4: Polecenia wyboru fontów w trybie matematycznym

<i>Polecenie</i>	<i>Przykład</i>	<i>Wynik</i>
<code>\mathcal{...}</code>	<code>\$\mathcal{B}=c\$</code>	$\mathcal{B} = c$
<code>\mathrm{...}</code>	<code>\$\mathrm{K}_2\$</code>	$K_2$
<code>\mathbf{...}</code>	<code>\$\sum x=\mathbf{v}\$</code>	$\sum x = \mathbf{v}$
<code>\mathsf{...}</code>	<code>\$\mathsf{G\times R}\$</code>	$G \times R$
<code>\mathtt{...}</code>	<code>\$\mathtt{L}(b,c)\$</code>	$L(b, c)$
<code>\mathnormal{...}</code>	<code>\$\mathnormal{R_{19}}\backslash\neq R_{19}\$</code>	$R_{19} \neq R_{19}$
<code>\mathit{...}</code>	<code>\$\mathit{ffi}\backslash\neq ffi\$</code>	$\mathit{ffi} \neq ffi$

### 6.2.2. Uwaga, niebezpieczeństwo!

Jak zaznaczyliśmy na początku rozdziału, nie należy instrukcji zmiany fontu wstawiać *explicite* do pliku źródłowego. Byłoby to niezgodne z podstawową ideą  $\text{\LaTeX}$ a, jaką jest oddzielenie formy od treści dokumentu i posługiwanie się formatowaniem logicznym, a nie wizualnym. Jeżeli fragment tekstu ma zostać wyróżniony przez złożenie go innym krojem lub stopniem pisma, to należy zdefiniować odpowiednie polecenie i potem właśnie jego używać w treści dokumentu.

```
% w~preamble albo pakiecie
\newcommand{\uwaga}[1]{\textbf{#1}}
% po \begin{document}
\uwaga{Bacność!} Przewody sieci
trakcyjnej są pod napięciem.
Dotknięcie grozi \uwaga{śmiercią}.
```

**Bacność!** Przewody sieci trakcyjnej są pod napięciem. Dotknięcie grozi **śmiercią**.

Niewątpliwą zaletą tego podejścia jest to, że kiedy później będziemy chcieli wyróżnić wszystkie elementy, na które czytelnik powinien zwrócić *szczególną uwagę*, w sposób inny niż składając je pismem półgrubym, to nie musimy przeglądać całego pliku w celu sprawdzenia, czy dane wystąpienie `\textbf` dotyczy tekstu, na który ma zostać zwrócona *szczególna uwaga*, czy też wstawione zostało w zupełnie innym celu.

**Na zakończenie rada z gatunku estetycznych:** nie należy przesadzać ze stosowaniem wielu różnych krojów pisma w jednym dokumencie.

### 6.2.3. Użycie alternatywnych krojów pisma

Większość dokumentów jest składanych w  $\text{\LaTeX}$ u z użyciem domyślnego kroju, będącego repliką *Computer Modern*. Jeżeli wygląd znaków z rodziny CM nam się znuży, to możemy złożyć dokument innym krojem. Musimy jednak pamiętać, że na ogół alternatywne kroje nie są tak kompletne jak

rodzina *Computer Modern*, często na przykład brakuje w nich kompletu symboli matematycznych, znaków z alfabetów nielacińskich, takich jak greka lub alfabet cyrylicy, albo niektórych odmian, jak na przykład kapitalików.

Z drugiej strony krój CM ma też wady: kreski znaków są cieńsze, a względna wysokość małych liter<sup>2</sup> jest mniejsza niż w wielu innych krojach. Te cechy kroju CM powodują, że jest mniej czytelny w wypadku, gdy jest reprodukowany na nośniku o niskiej rozdzielczości, lub – mówiąc wprost – nie najlepiej się nadaje do dokumentów, które będą wyświetlane na ekranach komputerów, np. dokumentów w formacie PDF.

Pakiet `qtimes` umożliwia skład dokumentu w kroju QTimes, który jest klonem znanego kroju Times New Roman autorstwa Stanleya Morisona. Jeżeli dokument zawiera wzory matematyczne, to aby znaki w formułach były optycznie zgodne z otaczającym je tekstem, należy także dołączyć pakiety `txfonts` oraz `qtxmath`:

```
%& --translate-file=il2-pl
\documentclass[a4paper]{article}
\usepackage{polski}
\usepackage{txfonts,qtimes,qtxmath}
\usepackage{qswiss,qcourier}
\usepackage{sfheaders}
\author{Wanda Przechlewska}
\title{Test pakietu txfonts}
\begin{document} ...
```

Z kolei pakiet `qpalatin` wraz z pakietami `qpxmath` oraz `pxfonts` umożliwia skład dokumentu krojem QPalatino (łącznie ze wzorami matematycznymi), który jest klonem znanego kroju Palatino autorstwa Hermanna Zapfa:

```
%& --translate-file=il2-pl
\documentclass[a4paper]{article}
\usepackage{polski}
\usepackage{pxfonts,qpalatin,qpxmath}
\usepackage{qswiss,qcourier}
\usepackage{sfheaders}
\author{Maria Matysek}
\title{Test pakietu qpxfonts}
\begin{document} ...
```

W powyższych przykładach dołączono także pakiety `qswiss`, `sfheaders` oraz `qcourier`. Pakiet `qswiss` przedstawia domyślną odmianę bezszeryfową na krój Helvetica, a `sfheaders` przeddefiniowuje śródtytuły tak, że składane są krojem bezszeryfowym. Z kolei `qcourier` zamienia domyślny krój o stałej szerokości znaków na QCourier. W rezultacie pierwszy z dokumentów będzie podobny

<sup>2</sup>Względna w porównaniu do nominalnego stopnia pisma; wysokość ta wynosi dla kroju CM 43%, a dla kroju Palatino – 46%; dla dużych liter z tych krojów jest to odpowiednio: 68,9% i 68,5%.

do typowego dokumentu składanego na przykład edytorem MS Word; drugi z kolei będzie się doskonale nadawał do dokumentu PDF, który chcemy umieścić w sieci WWW, z uwagi na to, że krój Palatino – reprodukowany w niskiej rozdzielczości – jest dużo bardziej czytelny niż CM, a nawet *Times New Roman*. Uwaga: aby powyższe deklaracje zadziałały, instalacja L<sup>A</sup>T<sub>E</sub>Xa musi zawierać ww. fonty<sup>3</sup>.

## 6.3. Odstępy

### 6.3.1. Zmiana wielkości interlinii

Wielkość odstępów między wierszami dokumentu można zmienić, umieszczając w preambule polecenie `\linespread`, postaci:

```
\linespread{czynn timer}
```

Parametr *czynn timer* określa powiększenie odstępu między wierszami. Znany z maszyn do pisan ia efekt podwójnej interlinii, czyli podwojenie odstępu, uzyskamy za pomocą `\linespread{1.6}`. Aby otrzymać odstęp wielkości 1,5, powinniśmy wpisać `\linespread{1.3}`. Pojedynczemu i zarazem domyślnemu odstępowi odpowiada wartość 1.

Polecenie `\linespread` wpływa na odstępy międzywierszowe w całym dokumencie. Jeśli są wyraźne powody do zmiany odstępu tylko w pewnym fragmencie dokumentu, to lepsza może się okazać instrukcja:

```
\setlength{\baselineskip}{1.5\baselineskip}
```

Poniższy przykład ilustruje wykorzystanie `\baselineskip`:

```
{\setlength{\baselineskip}%
  {1.5\baselineskip}
Ten akapit jest składany
z~\ci{\baselineskip} ustawionym
na 1,5 dotychczasowej wartości.
Zwróćmy uwagę na wystąpienie
\ci{par} na końcu akapitu.\par}
```

Przeznaczenie tego akapitu jest jasne. Ilustruje on, że po zamykającym nawiasie klamrowym następuje powrót do normalnego składu.

Ten akapit jest składany z `\baselineskip` ustawionym na 1,5 dotychczasowej wartości. Zwróćmy uwagę na wystąpienie `\par` na końcu akapitu. Przeznaczenie tego akapitu jest jasne. Ilustruje on, że po zamykającym nawiasie klamrowym następuje powrót do normalnego składu.

<sup>3</sup>Wchodzą one w skład współczesnych dystrybucji L<sup>A</sup>T<sub>E</sub>Xa. Jeżeli ich nie masz, to powinien je doinstalować. W dokumentacji dystrybucji powinno być opisane, jak się to robi.

### 6.3.2. Odstępy wokół akapitów

Dwa dodatkowe parametry określają w L<sup>A</sup>T<sub>E</sub>Xu wielkość, odpowiednio, wcięcie akapitowego oraz odstępu między akapitami. Wpisując na przykład do preambuły dokumentu:

```
\setlength{\parindent}{0pt}
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}
```

ustalamy wielkość wcięcia akapitowego na 0 pt (co powoduje, że akapity będą się zaczynać bez wcięć), a odstęp między akapitami ustalamy na 1 ex plus 0,5 ex minus 0,2 ex. Drugi zapis oznacza, że normalny odstęp między akapitami, wynoszący 1,0 ex (jednostki miary w L<sup>A</sup>T<sub>E</sub>Xu podaje tabela 6.5 na stronie 109), może się zwiększyć do  $1,0 + 0,5 = 1,5$  ex lub zmniejszyć do  $1,9 - 0,2 = 0,8$  ex. W Europie kontynentalnej akapity składa się czasami bez wcięcia akapitowego, a jedynie z dodatkowym odstępem między nimi. Ale uwaga! Ten efekt pojawi się także w spisie treści, tabel i rysunków, gdzie poszczególne pozycje spisu będą od siebie bardziej oddalone (w spisach większość akapitów jest jednowierszowa). Aby uniknąć tego trochę śmiesznego efektu, należy usunąć powyższe instrukcje `\setlength` z preambuły dokumentu i wstawić je w części głównej, po poleceniach `\tableofcontents` itp. Najlepiej jednak wcale nie korzystać z tego sposobu, gdyż znakomita większość książek jest składana z wcięciem akapitowym, a nie z dodatkowymi odstępami między akapitami<sup>4</sup>.

Wcięcie akapitowe na początku akapitu wstawiamy poleceniem<sup>5</sup>:

```
\indent
```

Wstawienie `\indent`, kiedy wartość `\parindent` wynosi zero, nie przyniesie oczywiście żadnego efektu.

Aby uzyskać akapit bez wcięcia, należy przed nim umieścić polecenie:

```
\noindent
```

### 6.3.3. Odstępy poziome

Wielkość odstępów między słowami oraz między zdaniami L<sup>A</sup>T<sub>E</sub>X ustala automatycznie. Dodatkowy odstęp poziomy (przez *odstęp poziomy* rozumiemy odstęp między wyrazami, przez *odstęp pionowy* – odstęp między wierszami i akapitami) możemy wstawić poleceniem:

```
\hspace{odległość}
```

<sup>4</sup>Jednoczesne użycie wcięć i powiększonych odstępów między akapitami uważa się w Polsce za poważny błąd typograficzny.

<sup>5</sup>Dla uzyskania efektu wcięcia w pierwszym akapicie po tytule rozdziału, punktu itd. należy dołączyć do dokumentu pakiet `indentfirst` z zestawu pakietów „tools”.



Jeżeli taki odstęp, w wyniku złamania akapitu na wiersze, wypadnie na początku lub na końcu wiersza, to zostanie on usunięty – aby zapobiec justowaniu akapitu „w chorągiewkę”. Jeżeli  $\text{\LaTeX}$  ma wstawić odstęp także na początku lub na końcu wiersza, to zamiast `\hspace` należy użyć „gwiazdkowej” wersji `\hspace*`. Argument *odległość* oznacza wymiar  $\text{\LaTeX}$ owy. W najprostszej postaci jest to liczba wraz z jednostką odległości. Wykaz ważniejszych spośród dostępnych w  $\text{\LaTeX}$ u jednostek odległości znajduje się w tabeli 6.5.

To jest `\hspace{1.5cm}`odstęp  
równy 1,5~cm.

To jest                      odstęp równy 1,5 cm.

Tabela 6.5:  $\text{\LaTeX}$ owe jednostki miary

mm	milimetr $\approx 1/25$ cala	┐
cm	centymetr = 10 mm	┐
in	cal = 25,4 mm	┐
pt	punkt $\approx 1/72$ cala $\approx \frac{1}{3}$ mm	┐
em	w przybliżeniu szerokość „M” w bieżącym foncie	┐
ex	w przybliżeniu wysokość „x” w bieżącym foncie	┐

Często wygodnie jest użyć odległości „elastycznej”, zostawiając  $\text{\LaTeX}$ owi nieco swobody doboru takiej odległości, jaką uzna za najlepszą z punktu widzenia jakości składu. Taką elastyczną odległość zapisujemy następująco: *n* plus *p* minus *m*. Części „plus *p*” i „minus *m*” są opcjonalne (każda z nich można pominąć). Tego typu odległości mają naturalną wielkość *n* i mogą się kurczyć lub rozciągać w zakresie od *n* – *m* do *n* + *p*.

Omawiany wcześniej odstęp między akapitami (`\parskip`) jest przykładem  $\text{\LaTeX}$ owego wymiaru o zmiennej wielkości. Część wymiarów może mieć wartości zmienne, część jednak musi mieć wartość stałą. Powinno być na przykład zrozumiałe, że wcięcie akapitowe musi być wielkością stałą, podobnie jak szerokość i wysokość łamu.

Polecenie:

`\stretch{n}`

wstawia specjalny rozciągliwy odstęp, który potrafi wypełnić całą wolną przestrzeń w pionie lub w poziomie. Jeżeli na przykład wstawimy w wierszu dwa lub więcej poleceń `\hspace{\stretch{n}}`, to odstępy dzięki nim uzyskane będą miały wielkość według proporcji zadanych przez argument *n*. W poniższym przykładzie odstęp między *x* a *y* jest trzy razy mniejszy od odstępu między *y* a *z*.

`x\hspace{\stretch{1}}`  
`y\hspace{\stretch{3}}z`

x	y	z
---	---	---

Wielkość odstępów towarzyszących tekstowi warto dostosować do aktualnego rozmiaru czcionki. Można do tego użyć względnych jednostek miary `em` oraz `ex`:

`{\Large{}big\hspace{1em}y} \quad`  
`{\tiny{}tin\hspace{1em}y}`

big	y	tin	y
-----	---	-----	---

Przypominamy, że polecenia `\quad` i `\qquad` wstawiają odstęp poziomy o szerokości, odpowiednio, 1 em oraz 2 em.

### 6.3.4. Odstępy pionowe

Odstępy pionowe między akapitami, rozdziałami, punktami itp.  $\text{\LaTeX}$  wstawia automatycznie. Jeśli zachodzi potrzeba wstawienia dodatkowego odstępu pionowego, to należy zastosować polecenie:

`\vspace{odległość}`

Polecenie to należy oddzielić pustymi liniami od otaczającego je tekstu. Jeżeli w wyniku złamania strony odstęp taki znajdzie się na początku lub na końcu strony (będzie zaczynał lub też kończył kolumnę tekstu), to zostanie on usunięty. Jeżeli ma zostać wstawiony także na początku lub końcu strony, to należy użyć wersji „gwiazdkowej” `\vspace*`<sup>6</sup>. Argument *odległość* oznacza  $\text{\LaTeX}$ owy wymiar.

Do rozmieszczania tekstu kolumny w pionie można używać polecenia `\stretch`, łącznie z `\pagebreak`. W poniższym przykładzie tekst zostanie rozmieszczony tak, że odstęp u dołu będzie dwa razy mniejszy od odstępu u góry strony:

`\vspace{\stretch{1}}`

Tytuł i~autor

`\vspace{\stretch{2}}\pagebreak`

Dodatkowy odstęp między dwoma wierszami tego samego akapitu lub między wierszami tabeli możemy uzyskać poleceniem:

`\[odległość]`

Polecenia `\bigskip`, `\medskip` i `\smallskip` wstawiają odpowiednio odstępy „elastyczne” o następujących wielkościach: 12pt ± 4pt, 6pt ± 2pt oraz 3pt ± 1pt<sup>7</sup>.

<sup>6</sup>Zwróćmy uwagę, że w takim wypadku wysokość kolumny tekstu na sąsiednich stronach nie będzie jednakowa, stosujemy zatem polecenie `\vspace*` z pewną ostrożnością.

<sup>7</sup>Nie są to wielkości przypadkowe: 12 punktów to typowa odległość między liniami podstawowymi wierszy przy składzie pismem w stopniu 10 punktów.

## 6.4. Układ graficzny strony

Wymiary papieru można podać jako argumenty instrukcji `\documentclass`. Na podstawie zadeklarowanych wymiarów  $\text{\LaTeX}$  oblicza szerokość i wysokość kolumny, marginesy i inne parametry. Na rysunku 6.2 przedstawiono dostępne parametry graficznego układu strony. Do przygotowania rysunku użyliśmy pakietu `layout` z zestawu „tools”<sup>8</sup>. Jeżeli obliczone przez  $\text{\LaTeX}$ a wartości są z pewnych względów nieodpowiednie, to można je zmienić.

Zanim jednak zaczniemy eksperymentować, zwiększając na przykład szerokość szpalty, chwilę pomyślmy. Jak dla większości rzeczy w  $\text{\LaTeX}$ u, istnieją ważne powody, dlaczego szerokość szpalty jest taka a nie inna.

Z pewnością w porównaniu z wydrukiem przygotowanym za pomocą świeżo kupionego programu MS Word strona  $\text{\LaTeX}$ owa ma denerwująco wąską szpaltę. Ale spójrzmy na książkę z renomowanego wydawnictwa i policzmy na niej liczbę znaków w przeciętnym wierszu. Okazuje się, że wynosi ona około 66. Jeśli porównamy ją z wydrukiem złożonym przez  $\text{\LaTeX}$ a, to zapewne i tym razem będzie ona zbliżona do 66. Z doświadczeń wynika bowiem, że w miarę wzrostu liczby znaków w wierszu czytanie staje się męczące. Dzieje się tak, gdyż przy długich wierszach naszym oczom trudniej jest przenosić wzrok z końca jednego wiersza na początek następnego. Jest to jedna z przyczyn stosowania składu wielołamowego w gazetach i czasopiśmie.

Tak więc, jeśli zwiększamy szerokość kolumny, to pamiętajmy, że może to utrudnić odbiorcom czytanie naszej pracy. No, ale dość już kazań. Obiecaliśmy przecież wyjaśnić, jak można te rzeczy robić. . .

W  $\text{\LaTeX}$ u mamy dwie instrukcje do zmiany wielkości wymiarów, używane zazwyczaj w obrębie preambuły dokumentu. Pierwsza z nich *nadaje* parametrowi określoną wielkość:

`\setlength{parametr}{wielkość}`

Drugie polecenie *zwiększa* wartość parametru o określoną wielkość:

`\addtolength{parametr}{wielkość}`

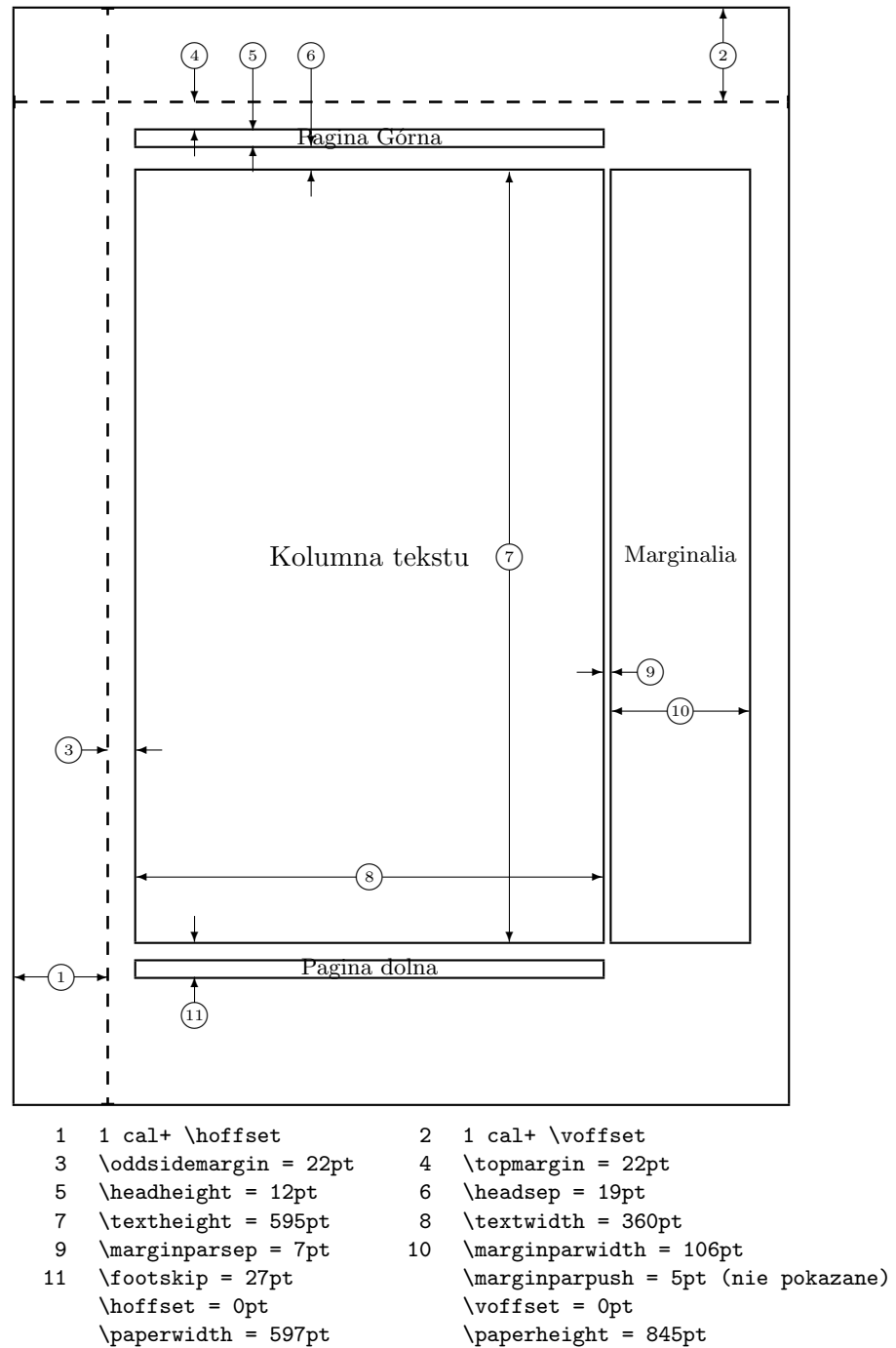
Z tej pary częściej stosowana jest druga instrukcja, ponieważ pozwala zmieniać wymiary. Przykładowo, aby zwiększyć szerokość szpalty o jeden centymetr, umieszczamy w preambule dokumentu następujące polecenia:

```
\addtolength{\hoffset}{-0.5cm}
\addtolength{\textwidth}{1cm}
```

Zmianę parametrów układu graficznego strony ułatwia pakiet `geometry`. W wykonywaniu operacji arytmetycznych na wymiarach pomaga pakiet `calc`.

---

<sup>8</sup>CTAN://macros/latex/packages/tools.



Rysunek 6.2: Parametry układu graficznego strony

## 6.5. Więcej o odległościach

Kiedy to tylko możliwe, unikajmy stosowania wymiarów zdefiniowanych w jednostkach absolutnych, takich jak punkty czy milimetry. Starajmy się raczej odnosić wymiary do już istniejących, takich jak wysokość czy szerokość kolumny. W poniższym przykładzie szerokość rysunku jest definiowana jako połowa szerokości bieżącej szpalty:

```
\includegraphics[width=0.5\textwidth]{sowauszata.eps}
```

Następujące trzy polecenia pozwalają określić szerokość, wysokość i głębokość *tekstu*:

```
\settoheight{nazwa}{tekst}
\settodepth{nazwa}{tekst}
\settowidth{nazwa}{tekst}
```

Oto przykład ilustrujący zastosowanie tych poleceń:

```
\newenvironment{vardesc}[1]{%
\settowidth{\parindent}{#1:\ }
\makebox[0pt][r]{#1:\ }}{}

\begin{displaymath} a^2+b^2=c^2
\end{displaymath}
```

```
\begin{vardesc}{gdzie}%
$a$, $b$ -- przyprostokątne, \par
$c$ -- przeciwprostokątna.
\end{vardesc}
```

$$a^2 + b^2 = c^2$$

gdzie:  $a$ ,  $b$  – przyprostokątne,  
 $c$  – przeciwprostokątna.

## 6.6. Pudełka

Każdą stronę  $\text{\LaTeX}$  tworzy z pudełek, które odpowiednio skleja. Elementarnymi pudełkami są litery, z których sklejać są słowa. Słowa są następnie łączone w wiersze, a wiersze – w akapity. Do łączenia używany jest specjalny klej, który dzięki elastyczności pozwala wyrazy ścisnąć lub rozciągnąć tak, by dokładnie wypełniały wiersze na stronie.

Trzeba przyznać, że takie ujęcie jest mocno uproszczoną wersją tego, co się naprawdę dzieje, chociaż zasadniczo biorąc, działanie  $\text{\TeX}$ a można jednak wyjaśnić właśnie w terminach pudełek oraz kleju (odstępu wstawianego między pudełkami). Pudełkami są nie tylko litery. Do pudełka można włożyć praktycznie wszystko, także inne pudełka. Każde pudełko  $\text{\LaTeX}$  traktuje jak pojedynczą literę.

Chociaż nie mówiliśmy o tym wprost, pudełka pojawiały się już w poprzednich rozdziałach. Na przykład polecenie `\includegraphics` albo otoczenie `tabular` tworzą pudełka. Dzięki temu dwa rysunki albo tabele można

łatwo zestawić obok siebie. Trzeba jedynie zadbać o to, by łączna szerokość połączonych obiektów nie przekraczała szerokości szpalty.

To samo odnosi się do akapitów, które – jeśli tego potrzebujemy – możemy składać w pudełka o zadanej szerokości:

```
\parbox[pos]{szerokość}{tekst}
```

Do tego samego celu można wykorzystać otoczenie:

```
\begin{minipage}[pos]{szerokość} tekst \end{minipage}
```

Argument *szerokość* to wymiar określający szerokość pudełka. Argument *pos* jest jednoliterowy i może przyjmować jedną z wartości: *c*, *t* lub *b*. Wartości te określają, jak L<sup>A</sup>T<sub>E</sub>X ma umieścić pudełko względem otaczającego tekstu. Wartość *c* oznacza umieszczenie środka wysokości pudełka na linii podstawowej, *t* – umieszczenie linii podstawowej pierwszego wiersza w pudełku na linii podstawowej otaczającego tekstu, natomiast *b* – umieszczenie dolnej krawędzi pudełka na linii podstawowej. Wynik zastosowania parametrów ilustruje poniższy przykład (linię podstawową oznaczono kreską):

```
\makebox[0pt][l]{\rule{66mm}{.4pt}}%
\parbox[c]{9mm}{5 5 5 5 5 5 5 5}
\parbox[t]{9mm}{6 6 6 6 6 6 6 6}
\parbox[b]{9mm}{8 8 8 8 8 8 8 8}
```

```

      8 8 8
5 5 5      8 8 8
5 5 5 6 6 6 8 8 8
5 5 5 6 6 6
      6 6 6
```

Polecenie `\parbox` składa tekst w pudełku, w razie potrzeby dzieląc tekst na linijki. Inaczej jest z pudełkami uzyskiwanymi poleceniem `\mbox`, których zawartość nigdy nie jest dzielona na wiersze. Polecenia tego używamy, gdy trzeba zapobiec dzieleniu wyrazu lub sekwencji wyrazów na wiersze. Polecenie `\mbox` jest uproszczoną wersją instrukcji `\makebox`, o składni:

```
\makebox[szerokość][pos]{tekst}
```

W opcjonalnym argumencie *szerokość* możemy zadać szerokość pudełka. Może się ona różnić od naturalnej szerokości tekstu w pudełku; może wynosić zero, a nawet być wielkością ujemną! W obrębie argumentu *szerokość* możemy się też posługiwać wielkościami `\width` (szerokość), `\height` (wysokość), `\depth` (głębokość) oraz `\totalheight` (suma wysokości i głębokości). Ponadto argument *pos* określa sposób umieszczenia tekstu. Litera *c* oznacza wyśrodkowanie, *l* – dosunięcie do lewej, *r* – dosunięcie do prawej, a *s* – wypsacjowanie zawartości.

Poniższy przykład ilustruje użycie polecenia `\width` w obrębie argumentu *szerokość*. Pierwsze pudełko ma szerokość równą połowie naturalnej szerokości tekstu:

```
\makebox[.5\width][l]{oooooooo}%
\makebox{xxxxxxxx}
```

```
ooooooooxxxx
```

Polecenie `\framebox` działa dokładnie jak `\makebox`, z tym że naokoło pudełka kreślona jest ramka.

Oto przykład zastosowania poleceń `\makebox` i `\framebox`:

```
\makebox[\textwidth]{%
p o ś r o d k u}\par
\makebox[\textwidth][s]{%
r o z s t r z e l o n y}\par
\framebox[1.1\width]{Teraz jestem Trochę tu jest za szeroko
obramowany!} \par
\framebox[0.8\width][r]{Trochę
tu jest za szeroko} \par
\framebox[1cm][l]{Nie
ma sprawy}
Da się to czytać?
```

```

                p o ś r o d k u
r o z s t r z e l o n y
Teraz jestem obramowany!
Trochę tu jest za szeroko
Nie ma sprawy
Da się to czytać?
```

Pudełka można też przesuwac w pionie. Służy do tego polecenie:

```
\raisebox{przesunięcie}[wysokość][głębokość]{tekst}
```

Argument *przesunięcie* określa wielkość przesunięcia w górę (lub w dół, jeżeli wielkość przesunięcia jest ujemna). Ponadto za pomocą parametrów opcjonalnych *wysokość* oraz *głębokość* można zadać nominalną wysokość oraz głębokość pudełka (L<sup>A</sup>T<sub>E</sub>X będzie traktował pudełko tak, jakby miało zadane wymiary, bez względu na wymiary naturalne). Wewnątrz parametrów można skorzystać z wielkości `\width`, `\height`, `\depth` oraz `\totalheight`.

```
\raisebox{0pt}[0pt][0pt]{\Large%
\textbf{Aaaa}\raisebox{-0.3ex}{a}%
\raisebox{-0.7ex}{aa}%
\raisebox{-1.2ex}{r}%
\raisebox{-2.2ex}{g}%
\raisebox{-4.5ex}{h}}}
Krzyczała, ale nikt nie zauważył,
że coś się jej przytrafiło.
```

```

Aaaaaaaar g h
Krzyczała, ale nikt nie za-
uważył, że coś się jej przytrafiło.
```

## 6.7. Kreski i podpory

W wyniku wykonania polecenia `\rule`, postaci:

```
\rule[przesunięcie]{szerokość}{wysokość}
```

otrzymujemy w składzie czarny prostokąt:

```
\rule{3mm}{.1pt}%
\rule[-1mm]{5mm}{1cm}%
\rule{3mm}{.1pt}%
\rule[1mm]{1cm}{5mm}%
\rule{3mm}{.1pt}
```



Polecenie `\rule` służy zwykle do rysowania kresek pionowych i poziomych. Na przykład gruba czarna krecha na stronie tytułowej niniejszego *Wprowadzenia* to wynik zadziałania instrukcji:

```
\rule[-1ex]{\textwidth}{5pt}
```

Parametr *przesunięcie* określa, jak wysoko przesunąć kreskę ponad linię podstawową (lub opuścić poniżej linii podstawowej, jeśli parametr jest ujemny).

Specjalnym przypadkiem jest kreska o zerowej szerokości, lecz o niezerowej wysokości. Taką kreskę nazywamy *podporą* (*strut*). Podpora to często stosowana metoda nadawania wszystkim elementom jednakowej wysokości. Spójrzmy na poniższy przykład. Dzięki wstawieniu podпоры drugi wiersz ma tę samą wysokość co pierwszy. Zawartość wiersza trzeciego jest identyczna jak drugiego, ale w drugim występuje niewidoczna podpora. Gdyby jej brakowało, to wysokość drugiego wiersza byłaby mniejsza.

```
\begin{tabular}{|c|} \hline
\rule{1pt}{4ex}Pittprop \ldots\\
\hline
\rule{0pt}{4ex}Strut\\ \hline
Strut\\ \hline
\end{tabular}
```

Pittprop ...
Strut
Strut

## 6.8. Więcej o składaniu tabel

### 6.8.1. Tabele o zadanej szerokości

W otoczeniu `tabular` szerokość poszczególnych kolumn tabeli jest ustalana automatycznie, a szerokość tabeli jest sumą szerokości kolumn i odstępów międzykolumnowych. Czasami jednak trzeba złożyć tabelę o z góry zadanej szerokości; jest to potrzebne przykładowo wtedy, gdy wszystkie tabele w dokumencie mają mieć tę samą szerokość.

Do tego celu można użyć albo otoczenia `tabular*`, albo otoczenia `tabularx` z pakietu o tej samej nazwie. Niestety każdy z tych sposobów ma poważne ograniczenia funkcjonalne. Rozpocznijmy od otoczenia `tabular*`, które różni się od `tabular` tylko jednym dodatkowym parametrem obowiązkowym, określającym szerokość tabeli:

```
\begin{tabular*}{szerokość}{spec-kolumn}
```



Rozważmy przykład tabeli zawierającej najważniejsze „parametry” słynnych wzniesień i przełęczy:

```
\begin{tabular}{|l|r|r|r|r|}\hline
Nazwa & \multicolumn{2}{|c|}{Wys. w m n.p.m.} & & \\
Długość & \multicolumn{2}{|c|}{Nachylenie \%} & & \\
\cline{2-3} \cline{5-6}% \cline można wstawiać wielokrotnie
& początek & koniec & w km & śr. & max \\ \hline
Col du Galibier & 1401 & 2646 & 18,1 & 6,9\% & 14,5 \\ \hline
Alpe D'Huez & 724 & 1815 & 14,2 & 7,7\% & 15,0 \\ \hline
Passo Gavia & 1734 & 2618 & 13,5 & 6,5\% & 20,0 \\ \hline
\end{tabular}
```

Złożone tabele wyglądają następująco:

Nazwa	Wys. w m n.p.m.		Długość w km	Nachylenie %	
	początek	koniec		śr.	max
Col du Galibier	1401	2646	18,1	6,9%	14,5
Alpe D'Huez	724	1815	14,2	7,7%	15,0
Passo Gavia	1734	2618	13,5	6,5%	20,0

Jeżeli tabela ma być złożona na przykład na szerokość łamu, to zastępujemy `tabular` jego wersją z gwiazdką oraz modyfikujemy preambułę tabeli:

```
\begin{tabular*}{\textwidth}%
{@{\extracolsep{\stretch{1}}}|l|r|r|r|r|}\hline ...
\end{tabular*}
```

Złożona tabela wygląda następująco:

Nazwa	Wys. w m n.p.m.		Długość w km	Nachylenie %	
	początek	koniec		śr.	max
Col du Galibier	1401	2646	18,1	6,9%	14,5
Alpe D'Huez	724	1815	14,2	7,7%	15,0
Passo Gavia	1734	2618	13,5	6,5%	20,0

Wewnątrz otoczenia `tabular`  $\text{\LaTeX}$  odziera poszczególne kolumny stałym odstępem równym `\tabcolsep`. Do składania tabel o określonej szerokości należy ten odstęp zamienić na taki, którego wielkość może się zmieniać (por. punkt 6.3.3). Do tego celu należy użyć wspomnianej w punkcie 2.11.6 (s. 39) instrukcji `@`. Ponadto polecenie `\extracolsep`, umieszczone wewnątrz instrukcji `@{...}`, wstawia *dodatkowy odstęp* między kolejnymi kolumnami – do odwołania poleceniem `\extracolsep` albo aż do końca tabeli.

W powyższym przykładzie polecenie `@{...}` usuwa domyślny odstęp międzykolumnowy, zastępując go specjalnym odstępem o zmiennej wielkości (taki odstęp wstawia polecenie `\stretch`). Manipulując odstępem międzykolumnowym,  $\text{\LaTeX}$  dopasowuje szerokość tabeli do żądanej wielkości. Otoczenie `tabular*` powinno zawierać w specyfikacji formatu kolumn co najmniej jedną konstrukcję `@{\extracolsep{\stretch{1}}}`. Uważny czytelnik dostrzeże jednak, że kreski poziome pod pierwszym wierszem między

kolumnami 2–3 oraz 5–6 nie są „dociągnięte” do lewego brzegu. Niestety nie da się tego naprawić – przynajmniej nie w prosty sposób. Otoczenie `tabular*` nadaje się do składania tabel, ale tylko wtedy, gdy nie korzystamy zbyt często z polecenia `\cline`.

Spróbujmy teraz zastosować otoczenie `tabularx`. W tym celu najpierw do preambuły dokumentu dołączamy pakiet `tabularx`. Następnie zmieniamy tabelę w taki sposób (kolumny, których szerokość ma być wyznaczona automatycznie, oznaczamy symbolem `X`):

```
\begin{tabularx}{\textwidth}{|X|X|X|X|X|X|}\hline
...
\end{tabularx}
```

Złożona tabela wygląda następująco:

Nazwa	Wys. w m n.p.m.		Długość w km	Nachylenie %	
	początek	koniec		śr.	max
Col du Galibier	1401	2646	18,1	6,9%	14,5
Alpe D'Huez	724	1815	14,2	7,7%	15,0
Passo Gavia	1734	2618	13,5	6,5%	20,0

Teraz wprowadzie wszystkie kreski są elegancko dociągnięte, ale jest problem z pierwszą kolumną: jest ona zbyt wąska, a przez to zawartość rubryk już się nie mieści i musi zostać przeniesiona<sup>9</sup>. Otoczenie `tabularx` automatycznie dzieli bowiem tabelę wyłącznie na kolumny *o równej szerokości*. Jest to jego największe ograniczenie funkcjonalne.

Specyfikacja kolumn tabeli oprócz `X` może zawierać także wartości „tradycyjne”, takie jak: `l`, `r` lub `c`. Odpowiadającym tym specyfikacjom kolumny mają naturalną szerokość. Przykładowo:

```
\begin{tabularx}{\textwidth}{|l|X|X|X|X|X|}\hline
...
\end{tabularx}
```

Akurat w naszym przykładzie osiągnęliśmy zadowalający rezultat, gotowa tabela wygląda bowiem następująco:

Nazwa	Wys. w m n.p.m.		Długość w km	Nachylenie %	
	początek	koniec		śr.	max
Col du Galibier	1401	2646	18,1	6,9%	14,5
Alpe D'Huez	724	1815	14,2	7,7%	15,0
Passo Gavia	1734	2618	13,5	6,5%	20,0

<sup>9</sup>Ponadto lepiej, by kolumny 2–6 były wyrównane do prawego, a nie do lewego brzegu rubryki.

Tabele złożone z użyciem otoczenia `tabular` nie są automatycznie dzielone między stronami. Do składu tabel, które nie mieszczą się na pojedynczej stronie, służy pakiet `longtable`, opisany w następnym punkcie.

### 6.8.2. Pakiet `longtable`

Pakiet definiuje otoczenie `longtable`, pozwalające składać tabele ciągnące się przez wiele kolejnych stron dokumentu. Aby zapewnić jednakowe szerokości rubryk na wszystkich stronach, wymagana jest dwukrotna kompilacja dokumentu. Oto przykład:

```
\begin{longtable}{|l|r|r|r|}
\caption{Tytuł tabeli}\\ \hline
\multicolumn{4}{|c|}{To jest nagłówek pierwszej strony}\\ \hline
nazwa & wysokość & długość & nachylenie \\ \hline
k-1 & k-2 & k-3 & k-4 \\ \hline
\endfirsthead
\hline
\multicolumn{4}{|c|}{To jest nagłówek następnych stron}\\
\hline k-1 & k-2 & k-3 & k-4\\ \hline
\endhead
\hline \multicolumn{4}{|c|}{Stopka tabeli}\\ \hline
\endfoot
\hline \multicolumn{4}{|c|}{Stopka na ostatniej stronie}\\
\hline
\endlastfoot
Col du T'el'egraphe & 1,566 & 12,0 & 6,7\% \\
Col du Galibier & 2,646 & 18,1 & 6,9\% \\
Col de la Madeleine & 2,000 & 25,4 & 6,1\% \\
%% ... 17 pominiętych wierszy ...
La Bola del Mundo & 2257 & 21,8 & 6,2\% \\
\end{longtable}
```

Po złożeniu efekt jest następujący<sup>10</sup>:

Tabela 6.6: Tytuł tabeli

To jest nagłówek pierwszej strony			
nazwa	wysokość	długość	nachylenie
k-1	k-2	k-3	k-4
Col du Télégraphe	1,566	12,0	6,7%
Col du Galibier	2,646	18,1	6,9%
Col de la Madeleine	2,000	25,4	6,1%
Stopka tabeli			

<sup>10</sup>Dane z tabeli pochodzą z katalogu <http://www.salite.ch>.

To jest nagłówek następnych stron			
k-1	k-2	k-3	k-4
Alpe D'Huez	1815	14,2	7,7%
Col de la Croix de Fer	2067	22,0	7,0%
Col de Portet d'Aspet	1069	4,4	9,6%
Mont Ventoux	1912	21,0	7,6%
Col de l'Izoard	2361	15,9	6,9%
Passo dello Stelvio	2758	24,3	7,4%
Col du Trébuchet	1143	14,6	4,3%
Passo Gavia	2618	13,5	6,5%
Col du Grand Colombier	1505	15,9	7,8%
Col de l'Iseran	2770	48,0	4,1%
Courchevel	2000	21,7	6,5%
Col de l'Aspin	1489	12,0	6,5%
Col de la Croix de l'Homme Mort	1163	18,0	4,3%
Col de Notre Dame des Abeilles	1000	13,9	5,1%
Paso del Morredero	1872	25,0	5,4%
Collado de la Caballar	1308	14,5	7,5%
Capilla de la Magdalena	1137	11,0	5,5%
La Bola del Mundo	2257	21,8	6,2%
Stopka na ostatniej stronie			

### 6.8.3. Pakiet array

Do składu tabel o bardziej skomplikowanym układzie graficznym lepiej używać pakietu `array`. Pakiet ten nie wprowadza żadnego nowego otoczenia, redefiniuje jedynie i rozszerza standardowe otoczenie `tabular`.

W „standardowym” otoczeniu `tabular` konstrukcja `p{szer-kolumn}` deklaruje kolumnę, w której zawartość każdej rubryki będzie składana w prostokąt o zadanej szerokości, z wyrównywaniem obu marginesów (odpowiednik `\parbox[t]{szer}`). Pakiet `array` wprowadza ponadto `b{szer}` – odpowiednik `\parbox[b]{szer}` – oraz `m{szer}` – odpowiednik `\parbox{szer}`. Przykład:

```
\begin{tabular}{|p{8mm}|p{8mm}|p{8mm}|} \hline
x y z x y z x y z x y z & x y z x y z x y y&
1 1 1 1 1 \\\hline \end{tabular}
\begin{tabular}{|m{8mm}|m{8mm}|m{8mm}|} \hline
x y z x y z x y z x y z & x y z x y z x y y&
2 2 2 2 2 \\\hline \end{tabular}
\begin{tabular}{|b{8mm}|b{8mm}|b{8mm}|} \hline
x y z x y z x y z x y z & x y z x y z x y y&
3 3 3 3 3 \\\hline
\end{tabular}
```

Złożone tabele wyglądają następująco:

x y z	x y z	1 1 1	x y z			x y z		
x y z	x y z	1 1	x y z	x y z	2 2 2	x y z		
x y z	x y y		x y z	x y z	2 2	x y z	x y z	
x y z			x y z	x y y		x y z	x y z	3 3 3
x y z			x y z			x y z	x y y	3 3

Kolejnym użytecznym rozszerzeniem zdefiniowanym w `array` jest możliwość bardziej szczegółowego określenia sposobu formatowania rubryk tabeli, niż ma to miejsce w standardowym otoczeniu `tabular`, gdzie w zasadzie możemy jedynie określić sposób justowania zawartości rubryk w poszczególnych kolumnach. Wracając do przykładu tabeli zawierającej różne „parametry” dotyczące przełęczy i wniesień, założmy, że nazwę wzniesienia chcemy złożyć kursywą, a długość – odmianą półgrubą. Można to oczywiście osiągnąć, wstawiając odpowiednie polecenia do każdej rubryki danej kolumny, ale sprawniej będzie zastosować zdefiniowane w pakiecie `array` konstrukcje:

`>{polecenia}` lub `<{polecenia}`

Konstrukcji `>{polecenia}` można użyć w preambule *przed* `c`, `l`, `r`, `p` oraz `m` i `b`. Jej działanie polega na wstawieniu *poleceń* na początku każdej rubryki w tej kolumnie. Podobnie, `<{polecenia}` można użyć *po* `c`, `l`, `r`, `p`, `m` i `b`. W rezultacie *polecenia* zostaną wstawione na końcu każdej rubryki w tej kolumnie.

Wracając do przykładu, oto preambuła tabeli wykorzystująca omawiane konstrukcje pakietu `array`:

```
\begin{tabular}{|>{\itshape}l|r|r|>{\bfseries}r|r|r|}\hline
Nazwa & \multicolumn{2}{|c|}{Wys. w m n.p.m.} & & & \\
Długość & \multicolumn{2}{|c|}{\{Nachylenie \%}\} \\
...
```

Wynik jest następujący:

Nazwa	Wys. w m n.p.m.		Długość w km	Nachylenie %	
	początek	koniec		śr.	max
<i>Col du Galibier</i>	1401	2646	<b>18,1</b>	6,9%	14,5
<i>Alpe D'Huez</i>	724	1815	<b>13,5</b>	7,7%	15,0
<i>Passo Gavia</i>	1734	2618	<b>13,5</b>	6,5%	20,0

Do zmiany głębokości konkretnego wiersza służy opcjonalny argument polecenia `\\`, np. `\\[2pt]`. Wysokość i głębokość wiersza można zmienić za pomocą niewidzialnej kreski, np. `\rule[-3mm]{0mm}{8mm}`. Przykładowo, w powyższym przykładzie tabela będzie wyglądała lepiej, jeżeli dodamy do pierwszego wiersza:

`Col du Galibier\rule[-3.5pt]{0pt}{15pt}`

Spowoduje to wstawienie dodatkowego odstępu między kreską nad pierwszym wierszem tabeli a zawartością wiersza.

#### 6.8.4. Pakiet tap

Pakiet `tap` (autorzy: Bogusław Jackowski, Piotr Pianowski i Piotr Strzelczyk) pozwala na eleganckie składanie nawet najbardziej ekstrawaganckich (jednostronicowych) tabel, z tym że stosuje niestandardowy zapis<sup>11</sup>, co może powodować problemy w integracji z innymi pakietami L<sup>A</sup>T<sub>E</sub>Xa, np. pakietem `colortbl`, służącym do kolorowania rubryk w tabelach. Oto przykład:

```
\input{tap.tex} %% <- dołączenie pakietu
\desiredwidth=\textwidth %% skład na szerokość łamu
\begin{table}
\begin{tableformat} \left " & \right \end{tableformat}
\=
\B!^ | @2\center{Wys. w~m~n.p.m.} | Długość |
      @2\center{Nachylenie \%} \E!
\B"- \center{Nazwa} " @2\center{Nazwa} " @2\center{Nazwa} \E"
\B"_ | początek | koniec | w~km | śr. | max \E!
\B! Col du Galibier | -- | 2645 | 17,5 | 6,9\% | 14,5 \E!
\B! Alpe D'Huez | -- | 1839 | 13,0 | 8,5\% | 15,0 \E!
\B! Passo Gavia | -- | 2621 | 17,3 | 7,9\% | 20,0 \E!
\=
\end{table}
```

Wynik jest imponujący:

Nazwa	Wys. w m n.p.m.		Długość w km	Nachylenie %	
	początek	koniec		śr.	max
Col du Galibier	1401	2646	18,1	6,9%	14,5
Alpe D'Huez	724	1815	14,2	7,7%	15,0
Passo Gavia	1734	2618	13,5	6,5%	20,0

Więcej informacji można odnaleźć w (polskojęzycznej) dokumentacji pakietu ([CTAN://macros/generic/tables/tap077.zip](http://CTAN://macros/generic/tables/tap077.zip)).

Na zakończenie uwaga: składanie tabel w L<sup>A</sup>T<sub>E</sub>Xu nie jest może aż tak proste jak w edytorach WYSIWYG, ale za to można tworzyć konstrukcje niezwykle trudne lub wręcz niemożliwe do wykonania w innych programach. Ponadto zadanie składania tabel można sobie znakomicie ułatwić, stosując skryptowe języki programowania, w rodzaju Perla czy Pythona. Już stosunkowo niewielka umiejętność programowania w wymienionych językach pozwala na szybką konwersję generowanych przez zewnętrzne aplikacje danych tabelarycznych do formatu L<sup>A</sup>T<sub>E</sub>Xa.

<sup>11</sup>Tak naprawdę, nie jest to nawet pakiet w rozumieniu L<sup>A</sup>T<sub>E</sub>Xa i zamiast poleceniem `\usepackage` należy go dołączać poleceniem `\input`.

# Bibliografia

- [1] Borzyszkowski Andrzej: Bib $\TeX$  – narzędzie do przygotowania bibliografii. Biuletyn GUST 1999 (13), Dostępny także w <http://www.ipipan.gda.pl/~andrzej/papers/bibtex.pdf>
- [2] Carlisle David P.: *Packages in the ‘graphics’ bundle*. Dokument dostępny w zestawie pakietów „graphics” w pliku `grfguide.tex`.
- [3] Chwałowski Robert: *Typografia typowej książki*, Helion 2001, ISBN: 83-7197-545-7, Por. też <http://www.typografia.ogme.pl/>.
- [4] Diller Antoni: *L $\TeX$  wiersz po wierszu*, tłum. Jan Jełowicki, Helion, Gliwice 2001, ISBN: 83-7197-341-1.
- [5] Eijkhout Victor: *T $\TeX$  by Topic, A T $\TeX$ nician’s Reference*, Addison-Wesley, Wokingham, England 1992 ISBN: 0-201-56882-9 Dostępny w <http://www.eijkhout.net/tbt/>.
- [6] Mittelbach Frank i inni: *L $\TeX$  Companion*, 2nd Edition, Addison-Wesley, Reading 2004, ISBN: 0201362996.
- [7] Goossens Michel, Rahtz Sebastian, Mittelbach Frank: *The L $\TeX$  Graphics Companion*. Addison-Wesley, Reading, Massachusetts, 1997, ISBN 0-201-85469-4.
- [8] Hobby John D.: *A User’s Manual for MetaPost*. Dostępny w <http://cm.bell-labs.com/who/hobby/>. Polskie tłumaczenie Joanny Marszałkowskiej jest dostępne w <ftp://ftp.gust.org.pl/pub/GUST/doc/mpint-pl.pdf>.
- [9] Hoenig Alan: *T $\TeX$  Unbound*. Oxford University Press, 1998, ISBN 0-19-509685-1+; 0-19-509686-X.
- [10] Jackowski Bogusław: *Co ma Bézier do B-spline’a?* Biuletyn GUST 2001 (17), ISSN: 1230-5650, Dostępny także w <ftp://ftp.gust.org.pl/pub/GUST/bulletin/17/jacko01.ps.gz>.
- [11] Knuth Donald E.: *The T $\TeX$ book*, Addison-Wesley Publishing Company 1984, ISBN 0-201-13448-9.
- [12] Lamport Leslie: *L $\TeX$ : A Document Preparation System*. 2nd ed., Addison-Wesley, Reading 1994, ISBN 0-201-52983-1. Polskie tłumaczenie Marii Wolińskiej i Marcina Wolińskiego *L $\TeX$  System opracowywania dokumentów. Podręcznik i przewodnik użytkownika*, WNT, Warszawa 2004, ISBN 83-204-2878-5.
- [13] L $\TeX$ 3 Project Team: *L $\TeX$  2 $\epsilon$  for authors*. Dokument dostępny w pliku `usrguide.tex` w dystrybucji L $\TeX$  2 $\epsilon$ .

- [14] L<sup>A</sup>T<sub>E</sub>X3 Project Team: *L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> for Class and Package writers*. Dokument dostępny w pliku `clsguide.tex` w dystrybucji L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>.
- [15] L<sup>A</sup>T<sub>E</sub>X3 Project Team: *L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> Font selection*. Dokument dostępny w pliku `fontguide.tex` w dystrybucji L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>.
- [16] *L<sup>A</sup>T<sub>E</sub>X Local Guide*: Każda wielodostępna instalacja L<sup>A</sup>T<sub>E</sub>X-owa powinna zawierać *L<sup>A</sup>T<sub>E</sub>X Local Guide*, w którym są opisane rzeczy specyficzne dla danej lokalnej instalacji. Dokument ten powinien być zawarty w pliku `local.tex`. W wielu wypadkach administratorzy nie udostępniają jednak użytkownikom takiego dokumentu. Pozostaje wtedy zwrócić się o pomoc do lokalnego L<sup>A</sup>T<sub>E</sub>X-owego guru.
- [17] Lichoński Bogusław: *T<sub>E</sub>X na indeksie*. Biuletyn GUST 1994 (3), ISSN: 1230-5650, Dostępny także w <ftp://ftp.gust.org.pl/pub/GUST/bulletin/03/02-bl.pdf>.
- [18] Macewicz Włodzimierz: Wirtualna Akademia T<sub>E</sub>Xowa. Dostępna w <http://www.ia.pw.edu.pl/~wujek/tex/>.
- [19] Myszka Wojciech: *Włączanie grafik do tekstów w L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>*, Dostępny w <http://www.immt.pwr.wroc.pl/~myszka/grafika/grafika.pdf>.
- [20] Nowacki Janusz M.: *T<sub>E</sub>Xnologia a typografia*. Biuletyn GUST 1995 (6), ISSN: 1230-5650, Dostępny także w <ftp://ftp.gust.org.pl/pub/GUST/bulletin/06/01-jmn.pdf>.
- [21] Oswald Urs: *Graphics in L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>*, containing some Java source files for generating arbitrary circles and ellipses within the `picture` environment, and *MetaPost – A Tutorial*. Dostępne w <http://www.ursoswald.ch>.
- [22] Reckdahl Keith: *Using EPS Graphics in L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> Documents*. Dostępny w [CTAN://info/epslatex.ps](http://ctan.org/info/epslatex.ps).
- [23] Rose Kristoffer H.: *X<sub>Y</sub>-pic User's Guide*. Dostępne w CTAN z pakietem X<sub>Y</sub>-pic.
- [24] Sapijaszko Grzegorz: *Tworzenie dokumentów pdf przy pomocy L<sup>A</sup>T<sub>E</sub>Xa*, Dostępny w <http://www.sapijaszko.net/pedeeefy.pdf>.
- [25] Schöpf Rainer, Raichle Bernd, Rowley Chris: *A New Implementation of L<sup>A</sup>T<sub>E</sub>X's verbatim Environments*. Dokument dostępny w zestawie pakietów „tools” w pliku `verbatim.dtx`.
- [26] Volovich Vladimir, Lemberg Werner i L<sup>A</sup>T<sub>E</sub>X3 Project Team: *Cyrillic languages support in L<sup>A</sup>T<sub>E</sub>X*. Rozpowszechniany w dystrybucji L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> w pliku `cyrguide.tex`.
- [27] Williams Graham: *The T<sub>E</sub>X Catalogue* (katalog pakietów dla T<sub>E</sub>Xa oraz L<sup>A</sup>T<sub>E</sub>Xa). Dokument dostępny w [CTAN://help/Catalogue/catalogue.html](http://ctan.org/help/Catalogue/catalogue.html).
- [28] Woliński Marcin: MWCLS *Moje własne klasy dokumentów dla L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>. Podręcznik użytkownika*. Dokument rozpowszechniany razem z zestawem klas `mwcls.zip`, por. <http://www.mimuw.edu.pl/~wolinski/mwcls.html>.



# Skorowidz

Uwaga: hasła wyróżnione imitacją pisma maszynowego, oznaczają polecenia (jeżeli są poprzedzone znakiem w-tył-ciacha) lub otoczenia; wartości opcji klas i pakietów oznaczono *odmianą pochylą* imitacji pisma maszynowego; programy zostały wyróżnione podkreśleniem zaś hasła złożone krojem bezszeryfowym oznaczają pakiety.

<code>\!</code> , 52	<code>\addcontentsline</code> , 32	<code>\bibitem</code> , 68
<code>\$</code> , 45	<code>\address</code> , 44	<code>\big</code> , 51
<code>\'</code> , 28	<code>\addtolength</code> , 111	<code>\Big</code> , 51
<code>\(</code> , 45	<code>æ</code> , 25	<code>\bigg</code> , 51
<code>\)</code> , 45	akcenty, 25	<code>\Bigg</code> , 51
<code>\,</code> , 46, 51	▷ matematyczne, 48	<code>\biggl</code> , 55
<code>\-</code> , 20	<code>align</code> , 54	<code>\biggr</code> , 55
..., zob. wielokropek	<code>all</code> , 94	<code>\bigskip</code> , 110
<code>\:</code> , 51	<code>\alt</code> , 83	<code>\binom</code> , 49
<code>\;</code> , 51	<code>amssbsy</code> , 57	block, 83
<code>\@</code> , 31	<code>amsmath</code> , 47, 64	<code>\bmod</code> , 49
<code>\[</code> , 45	<code>amsmath</code> , 46, 49–52, 54, 57	<code>\boldmath</code> , 57
<code>%</code> , 6	<code>amssymb</code> , 47, 58, 64	<code>\boldsymbol</code> , 57
<code>BibTeX</code> , 69	<code>amsthm</code> , 56, 57	Braams Johannes, 25
<code>\\</code> , 17, 36, 37, 39, 43, 110	<code>\and</code> , 33	<code>.bst</code> (plik), 69
<code>\\*</code> , 17	<code>\appendix</code> , 31, 32	calc, 111
WYSIWYG, 2, 3	<code>\ar</code> , 95	<code>\caption</code> , 41–43
<code>\]</code> , 45	argument, 6	Carlisle David, 58, 65
<code>^</code> , 47	▷ opcjonalny, 6	Casartelli Fabio, 120
<code>_</code> , 47	array, 120, 121	<code>\cc</code> , 44
<code>~</code> , 31	array, 52, 53	<code>\cdots</code> , 51
10pt, 10	arraycolsep, 53	center, 36, 68
11pt, 10, 51	<code>\atop</code> , 49	<code>\centering</code> , 68
12pt, 10, 51	<code>\author</code> , 33, 78, 83	<code>\chapter</code> , 31, 72
<code>\a'o</code> , 28	<code>.aux</code> (plik), 13, 32, 34, 69	<code>\chaptermark</code> , 72
<code>a4paper</code> , 10, 81	<code>b5paper</code> , 10	<code>\choose</code> , 49
<code>a5paper</code> , 10	babel, 11, 25–29	<code>\circle</code> , 88
<code>abstract</code> , 37	<code>\backmatter</code> , 33	<code>\circle*</code> , 88
<code>Acrobat</code> , 78, 79	backslash, 5	<code>\cite</code> , 68
<code>Acrobat Distiller</code> , 74	<code>\baselineskip</code> , 107	<code>\cleardoublepage</code> , 18, 42
<code>Acrobat Reader</code> , 74, 76, 81	<code>.bbl</code> (plik), 69	<code>\clearpage</code> , 18, 42
<code>acute</code> , 25	beamer, 9, 81–83	<code>\cline</code> , 40, 118
	<code>.bib</code> (plik), 69	<code>\closing</code> , 44

- .cls (plik), 13
- cmd, 8
- color, 76, 81
- colortbl, 122
- command, 8
- CorelDraw!, 65
- cp1250*, 8, 26
- czcionka, *zob.* pismo drukarskie
- \date, 33, 43, 44
- dcolumn, 39
- \ddots, 51
- \depth, 114, 115
- description, 36
- displaymath, 45
- \displaystyle, 55
- doc, 12
- \documentclass, 6, 9, 10, 13, 19, 29, 81
- draft*, 19
- .dtx (plik), 13, 73
- .dvi (plik), 9, 13, 66, 73
- dvipdf, 9
- dvips*, 67, 77
- dvips, 9, 13, 66, 74
- \dywiz, 22
- eepic, 85, 88
- \em, 35
- \emph, 35, 103
- enumerate, 36
- epic, 85
- .eps (plik), 76, 80
- EPS, 65, 66, 76
- epstopdf, 77
- eqnarray, 53
- eqnarray\*, 53
- \eqref, 46
- equation, 46, 53
- eucal, 64
- eufak, 64
- eurosym, 23
- \EURtm, 24
- executivepaper*, 10
- exscale, 12, 51
- \extracolsep, 117
- fancyhdr, 71
- \fbox, 20
- .fd (plik), 13
- figure, 40–42, 67
- flalign, 54
- fleqn*, 10
- flushleft, 36
- flushright, 36
- foiltex, 9
- \foldera, 92
- \folderb, 92
- font, 102
- fontenc, 12, 27, 29
- fonty
  - ▷ CM, 27
  - ▷ cm-super, 76
  - ▷ EC, 27, 76
  - ▷ LM, 27, 76
  - ▷ PL, 27, 76
- \footnote, 34, 43
- \footnotesize, 103
- \footskip, 112
- format, 25, 26, 28
- formatowanie
  - ▷ logiczne, 2
  - ▷ wizualne, 2
- \frac, 49
- frame, 83
- \framebox, 115
- \frametitle, 83
- Freehand, 65
- \frenchspacing, 31
- \frontmatter, 33
- \fussy, 19
- gather, 54
- geometry, 72, 111
- \geq, 59
- ghostscript, 9, 65, 74
- ghostview, 65
- gimp, 66
- .glo (plik), 73
- gnuplot, 65
- graphicx, 11, 65, 67, 76, 81
- grave, 25
- grupa, 103
- gsview, 65
- gv, 65
- \headheight, 112
- \headsep, 112
- \height, 114, 115
- hiperłącze, 74
- hipertekst, 74
- \hline, 39
- Hobby John D., 85
- \href, 78, 80
- \hspace, 108
- \hspace\*, 109
- HTML, 74
- \huge, 103
- \Huge, 103
- hyperref, 77, 79–81
- hyphenat, 72
- \hyphenation, 19, 20
- i („i” bez kropki), 25
- IDE, 75
- \idotsint, 52
- .idx (plik), 13, 70, 73
- ifpdf, 80
- \ifpdf, 80
- ifthen, 12
- \ignorespaces, 101
- \ignorespacesafterend, 101
- \iiint, 52
- \iint, 52
- \int, 52
- .ilg (plik), 13
- ImageMagick, 66
- \include, 14, 67
- \includegraphics, 66, 67, 76, 80, 113
- \includeonly, 14
- .ind (plik), 13, 70
- indeks
  - ▷ dolny, 47
  - ▷ górny, 47
- \indent, 108
- indentfirst, 17, 108
- \index, 70, 71
- \input, 14, 122
- inputenc, 12, 20, 26–29
- .ins (plik), 13, 72, 73
- \institute, 83
- \int, 50
- interlinia, 107
  - ▷ podwójna, 107
- \item, 36
- itemize, 36, 83
- j („j” bez kropki), 25

- Jackowski Bogusław, 20, 27, `\leq`, 59  
     84, 122  
jednostki miary, 109  
jpeg2ps, 66  
.jpg (plik), 67, 76, 80  
kasza, 102  
Kew Jonathan, 28  
Kile, 75  
klasa  
▷ article, 9  
▷ book, 10  
▷ letter, 10  
▷ mwart, 30  
▷ mwbook, 30  
▷ mwrep, 30  
▷ report, 10  
▷ slides, 9  
klej, 113  
Knuth Donald E., 1, 85  
kodowanie  
▷ LGR, 28  
▷ OT1, 27  
▷ OT4, 27  
▷ T1, 27  
▷ T2A, 28  
▷ T2B, 28  
▷ T2C, 28  
▷ X2, 28  
Kołodziejska Hanna, 20  
komentarz, 6  
kropka, 24  
\label, 34, 42, 46, 68  
    Lamport Leslie, 2  
\large, 103  
\Large, 103  
\LARGE, 103  
\LaTeX, 21  
     $\text{\LaTeX}$ 2.09, 2  
     $\text{\LaTeX}$ 2 $\epsilon$ , 2  
     $\text{\LaTeX}$ 3, 2, 4  
\LaTeXe, 21  
    latexsym, 12  
    latin2, 8, 26  
    layout, 111  
\ldots, 24, 51  
\left, 50, 51  
\leftmark, 72  
    legalpaper, 10  
        legno, 10  
        letter, 44  
        letterpaper, 10, 81  
        LGR, 28  
        ligatura, zob. spójka  
\line, 87, 92  
\linebreak, 17, 18  
\linespread, 107  
\linethickness, 90, 92  
    linia podstawowa, 51, 114, 116  
\listoffigures, 42  
\listoftables, 42  
.lof (plik), 13, 32  
.log (plik), 13, 76  
longtable, 119  
longtable, 119  
.lot (plik), 13, 32  
Macewicz Włodzimierz, 11, 70  
\mainmatter, 33, 79  
\makebox, 114, 115  
    makeidx, 12, 69  
\makeindex, 69  
    makeindex, 13, 69, 70  
\maketitle, 33  
    maktexsr, 73  
\marginparpush, 112  
\marginparsep, 112  
\marginparwidth, 112  
Marszałkowska Joanna, 123  
marvosym, 24  
math, 45  
\mathbb, 47  
\mathbf, 57, 105  
    mathcal, 64  
\mathcal, 105  
\mathit, 105  
\mathnormal, 105  
\mathrm, 55, 105  
    mathscr, 64  
\mathsf, 105  
\mathtt, 105  
\mbox, 20, 24, 114  
\medskip, 110  
METAPOST, 76, 77  
MeX, 30  
minipage, 114  
Mittelbach Frank, 2  
modulo, 49  
Morison Stanley, 106  
.mps (plik), 76, 80  
MS Office, 66  
MS Visio, 66  
\multicolumn, 39  
\multitup, 86, 89  
multiline, 54  
    nawias, 50  
\newcommand, 99  
\newenvironment, 99, 100  
\newline, 17, 18  
\newpage, 17, 18  
\newsavebox, 91  
\newtheorem, 56  
\newtheoremstyle, 56  
\noindent, 108  
\nolinebreak, 17  
    nomathsymbols, 30  
\nonumber, 54  
\nopagebreak, 17  
\normalsize, 103  
\not, 59  
    notitlepage, 10  
Nowacki Janusz, 27  
obracanie  
▷ rysunku, 68  
▷ tabeli, 68  
\oddsidemargin, 112  
odstęp, 4  
▷ na początku wiersza, 4  
▷ po instrukcji, 5  
▷ podwójny, 107  
▷ poziomy, 108  
▷ w trybie matematycznym, 51  
oe, 25  
ogranicznik, 50  
Olko Mariusz, 29  
onecolumn, 10  
oneside, 10  
\only, 83  
Oostrum Piet van, 71  
opcje, 10  
openany, 10, 18  
\opening, 44  
openright, 10, 18  
OpenType, 28

- operator
  - ▷ iloczynu, 50
  - ▷ sumowania, 50
- OT1*, 27, 29
- OT4*, 27, 29
- otoczenie, 35
- `\oval`, 90, 92
- `\overbrace`, 48
  - overfull hbox, 19
- `\overleftarrow`, 48
- `\overline`, 48
- `\overrightarrow`, 48
- `\pagebreak`, 17, 18, 110
- `\pageref`, 34, 74
- `\pagestyle`, 11
  - pakiet, 10
- `\paperheight`, 112
- `\paperwidth`, 112
- `\par`, 104, 107
- `\paragraph`, 31
- `\parbox`, 114
- `\parindent`, 108
- `\parskip`, 108
- `\part`, 31, 32
- `\pause`, 83
- .pdf (plik), 13, 67, 73, 76, 80
  - PDF, 74, 75, 78
  - pdfL<sup>A</sup>T<sub>E</sub>X, 75, 76, 81
  - pdfscreen, 83
  - pdftex*, 67, 76, 77
  - pdfT<sub>E</sub>X, 75
- `\phantom`, 54
  - Pianowski Piotr, 122
- picture, 84, 85, 88, 89
- pierwiastek kwadratowy, 48
- pismo
  - ▷ drukarskie, 102
  - ▷ krój, 102
  - ▷ odmiana, 102
  - ▷ stopień, 102
- platex, 22, 28, 29
- plmath*, 29
- `plmindex`, 13, 70
- `\pmb`, 57
- `\pmod`, 49
- .png (plik), 67, 76, 80
  - podpis, 68
  - polecenie, 5
    - ▷ kruche, 43
- `\polecenie`, 98
  - polski, 29–31, 49, 59, 95
  - PostScript, 9, 42, 75, 85
  - ppower4, 81
  - preambuła, 6
  - prim, 48
- `\printindex`, 70
- `\prod`, 50
- proof, 57
- prosper, 81
- `\protect`, 43
- `\providecommand`, 100
- `\ProvidesPackage`, 101
  - przecinek, 24
- .ps (plik), 73
- `\ps`, 44
  - pspicture, 85
  - pstricks, 75, 85, 88
  - pudelko, 68, 113
- `\put`, 86–91
- pxfonts, 106
- `\q bezier`, 84, 86, 93
- qcourier, 106
- `\qedhere`, 57
- qpalatin, 106
- qpxmath, 106
- `\qqquad`, 52, 110
- qswiss, 106
- qtimes, 106
- qtxmath, 106
- `\quad`, 46, 52, 110
- quotation, 37
- quote, 37
  - Radhakrishnana C.V., 83
- `\raisebox`, 115
- `\ref`, 34, 42, 46, 68, 74
- `\renewcommand`, 100
- `\renewenvironment`, 100
- `\right`, 50, 51
- `\rightmark`, 72
- rotate, 75
- rotate, 68
- rotating, 68
- `\rule`, 115, 116
- Ryćko Marek, 20, 27
- `\savebox`, 91
- `\scriptscriptstyle`, 55
- `\scriptsize`, 103
- `\scriptstyle`, 55
- `\section`, 31, 43, 72, 83
- `\sectionmark`, 72
- `\selecthyphenation`, 30
- `\selectlanguage`, 30
- `\setlength`, 85, 108, 111
- `\settodepth`, 113
- `\settoheight`, 113
- `\settowidth`, 113
- sfheaders, 106
- showidx, 70
- sideways, 68
- sidewaysfigure, 68
- sidewaystable, 68
- `\signature`, 43, 44
- Simpson Tom, 120
- `\sloppy`, 19
- słowo, 71
- `\small`, 103
- `\smallskip`, 110
- split, 54
- spójka, 24
- `\sqrt`, 48
- `\stackrel`, 50
- stopień pisma, 103
- `\stretch`, 109, 110, 117
- Strzelczyk Piotr, 122
- .sty (plik), 12, 73, 101
- subarray, 50
- `\subparagraph`, 31
- `\subsection`, 31, 83
- `\subsectionmark`, 72
- `\substack`, 50
- `\subsubsection`, 31
- `\sum`, 50
- `\surd`, 48
- symbol
  - ▷ końca dowodu, 57
- syntonly, 12
- środowisko, *zob.* otoczenie
  - T1*, 27, 29
  - T2A*, 28
  - T2B*, 28
  - T2C*, 28
- tabbing, 28
- `\tabcolsep`, 117
- table, 40–42
- `\tableofcontents`, 32, 42

- tabular, 38, 39, 53, 113, 116, 117, 119–121
- tabular\*, 116–118
- tabularx, 118
- tabularx, 116, 118
- tap, 122
- \temporal, 83
- .tex (plik), 8, 12
- \TeX, 21
  - TeXnicCenter, 75
- \texorpdfstring, 79, 80
- \text, 55
- \textbf, 103, 105
- \textcelsius, 23
  - textcomp, 23
- \texteuro, 23
- \textheight, 112
- \textit, 103
- \textmd, 103
- \textnormal, 103
- \textrm, 55, 103
- \textsc, 103
- \textsf, 103
- \textsl, 103
- \textstyle, 55
- \texttt, 103
- \textup, 103
- \textwidth, 68, 112
  - Thành Hàn Thế, 75
  - thebibliography, 68
- \thicklines, 87, 90, 92
- \thinlines, 87, 90, 92
- \thispagestyle, 11
  - tilde, 23
- \tiny, 103
- \title, 33, 83
- \titlegraphic, 83
  - titlepage, 10
  - Tkadlec Josef, 58
- .toc (plik), 13, 32
- \today, 21
- \topmargin, 112
- \totalheight, 114, 115
  - tryb matematyczny, 45
  - twocolumn, 10, 18
  - twoside, 10
- txfonts, 106
- tylda (~), 31
- ułamek
  - ▷ piętrowy, 49
  - ▷ zwykły, 49
- umlaut, 25
- \uncover, 83
- \underbrace, 48
  - underfull hbox, 19
- \underline, 48
  - unicode, 80
  - Unicode, 8, 26, 28, 29, 77, 80
- \unitlength, 85–87
- \updownarrow, 50
- url, 23
- \url, 99
  - URL, 23
- \usebox, 91
- \usepackage, 6, 11, 12, 23, 26, 67, 77, 101, 122
  - utf8, 8, 26
- \vdots, 51
- \vec, 48
- \vector, 87
- \verb, 38, 43, 99
  - verbatim, 38, 72
  - verbatim, 38, 72
- \verbatiminput, 72
  - verse, 37
- \vspace, 110
- \vspace\*, 110
  - w-tył-ciach, 5
  - wektor, 48
- \widehat, 48
- \widetilde, 48
- \width, 114, 115
  - wielokropek, 24, 51
  - Williams Graham, 11
  - Woliński Marcin, 29, 30
  - wstawka, 40
  - www, 23
  - wzorce podziału, 25
- X2, 28
- XeTeX, 28
- xfig, 65
- xpdf, 74
- xy, 95
- \xymatrix, 95
- yap, 9
  - zalety L<sup>A</sup>T<sub>E</sub>Xa, 3
  - Zapf Hermann, 106
- znak
  - ▷ całki, 50
  - ▷ sumowania, 50
- żywa pagina, 11, 71