Zofia Walczak

LaTeX dla niecierpliwych

część I

WYDZIAŁ MATEMATYKI I INFORMATYKI
UNIWERSYTETU ŁÓDZKIEGO

Zofia Walczak

LaTeX dla niecierpliwych

— część pierwsza —

WYDAWNICTWO
UNIWERSYTETU
ŁÓDZKIEGO

Zofia Walczak

LaTeX dla niecierpliwych

— część pierwsza —

Wydanie II
poprawione i uzupełnione

WYDZIAŁ MATEMATYKI I INFORMATYKI
UNIWERSYTETU ŁÓDZKIEGO

ŁÓDŹ 2014

Zofia Walczak – Uniwersytet Łódzki, Wydział Matematyki i Informatyki
Katedra Metodyki Nauczania Matematyki, 90-238 Łódź, ul. Banacha 22

RECENZENT
Aleksy Tralle

PROJEKT TYPOGRAFICZNY, SKŁAD I ŁAMANIE
Zofia Walczak

PROJEKT OKŁADKI
Szymon M. Walczak

© Copyright by Uniwersytet Łódzki, Łódź 2014

Wydane przez Wydawnictwo Uniwersytetu Łódzkiego
Wydanie II (poprawione i uzupełnione). W.06501.14.0.S

ISBN (wersja drukowana)978-83-7969-066-4
ISBN (ebook) 978-83-7969-180-7

Wydawnictwo Uniwersytetu Łódzkiego
90-131 Łódź, ul. Lindleya 8
www.wydawnictwo.uni.lodz.pl
e-mail: ksiegarnia@uni.lodz.pl
tel. (42) 665 58 63, faks (42) 665 58 62

Spis treści

Zamiast wstępu **7**

1 Jak zacząć **11**

1.1. Instalacja systemu 11

1.2. Wybieramy edytor 12

1.3. Przebieg kompilacji 16

2 Podstawowe informacje **19**

2.1. Struktura dokumentu 19

2.2. Akcenty, znaki specjalne 24

2.3. Tryby pracy LaTeX-a 27

2.4. Dokument w języku polskim 27

2.4.1. Dokument wielojęzyczny – pakiet `babel` 29

2.5. Jednostki miary 30

2.6. Fonty 32

2.6.1. Składanie dokumentu innym fontem niż domyślny . . 34

2.6.2. Fonty TeXGyre 39

3 Projektowanie dokumentu **45**

3.1. Plan pracy 45

3.2. Części dokumentu 47

3.3. Składanie dużych dokumentów 53

3.4. Style strony 54
3.5. Numerowanie stron 55
3.6. Pakiet `fancyhdr` 56
3.7. Klasy `extsizes` 61
3.8. Pakiet `geometry` 61

4 Składanie dokumentu **71**
4.1. Formatowanie tekstu 71
4.2. Struktury definiowane przez użytkownika 94
4.3. Przypisy 99
4.4. Odsyłacze 102
4.5. Tworzenie list 105
4.6. Pudełka 113
4.7. Pakiet `listings` 118

5 Tabele **125**
5.1. Ustawienia tabulatora do pisania w kolumnach 125
5.2. Składanie tabel 132
5.3. Wstawianie rysunków i tabel 139
5.3.1. Parametry dla środowisk `table` i `figure` 140

6 Matematyka **145**
6.1. Wyrażenia i wzory matematyczne 145
6.2. Pakiet `amsmath` 157
6.3. Pakiet `amsthm` 173
6.4. Wybrane symbole matematyczne 177

7 Grafika **181**
7.1. Kolory 181
7.2. Pakiet graphicx 185
7.3. Wstawianie plików graficznych 188

8 Dodatek **191**

8.1. Typografia, typograf 191

8.2. Strona tekstu 192

8.3. Pismo 194

Spis tabel **201**

Skorowidz **203**

Bibliografia **210**

Zamiast wstępu

TEX powstał ponad 30 lat temu w Stanach Zjednoczonych Ameryki Północnej, jego autorem jest Donald E. Knuth, matematyk i informatyk, autor między innymi wielotomowego dzieła pt. *Sztuka programowania* (The Art of Computer Pogramming).

TEX tworzy system do elektronicznego składu tekstu razem z:

- edytorem,
- sterownikami umożliwiającymi wydrukowanie lub wyświetlenie złożonego dokumentu,
- oraz wieloma programami towarzyszącymi, między innymi:
  - MetaFont do generowania fontów
  - MetaPost do tworzenia grafiki wektorowej,
  - BibTeX do przetwarzania baz bibliograficznych,
  - MakeIndex do tworzenia i sortowania skorowidzów.

TEX jest językiem niskiego poziomu, posiada ok. 300 pierwotnych sekwencji sterujących, które nie są rozkładalne na prostsze funkcje. Rozumie około 900 sekwencji sterujących zbudowanych z sekwencji pierwotnych. Są one częścią wbudowanego słownika i możliwe jest budowanie z nich makr i wygodnego interfejsu użytkownika. Donald E. Knuth opracował pierwszy zestaw makr, który nazwał formatem *Plain*. W połowie lat 80-tych ubiegłego stulecia powstały formaty LaTeX2.09 (Leslie Lamport) i $\mathcal{AMS}$TEX (M.D.

Spivak). Kilka lat później powstaje $\mathcal{AMS}$LaTeX, łączący możliwości i udogodnienia obu formatów. Kolejna wersja, która ukazała się w 1994 roku, nazywa się LaTeX2e i jest dzisiaj standardowym formatem LaTeX-a.

W ciągu kilku lat od powstania system TeX zyskał użytkowników w wielu krajach na świecie. Ci z nich, którzy nie posługiwali się językiem angielskim chcieli mieć możliwość składania tekstów w swoich językach ojczystych. Dostosowywaniem TeX-a do wymagań różnojęzycznych użytkowników (początkowo tylko używających mniej lub bardziej rozszerzonego alfabetu łacińskiego) zajęli się sami jego użytkownicy zrzeszeni w lokalnych grupach użytkowników systemu. Pierwsza taka grupa o nazwie TUG (TeX Users Grup) powstała w USA, polscy użytkownicy utworzyli w 1992 roku GUST (Grupa Użytkowników Systemu TeX).

W ostatnich 30 latach nastąpił ogromny postęp technologiczny w informatyzacji. Zmienił się nie tylko sprzęt, ale także oprogramowanie, powstały nowe technologie informatyczne. Konieczne więc stało się przystosowanie TeX-a do zmieniających się warunków technologicznych.

Pod koniec ubiegłego stulecia powstał ConTeXt, nowy system tworzenia dokumentów oparty na systemie TeX i programie METAPOST. Jego twórcą jest Hans Hagen. Więcej informacji o tym systemie można znaleźć na stronie http://wiki.contextgarden.net.

XeTeX, autorstwa Jonatana Kew, to system składu dokumentów łączący TeX-a Donalda Knutha z Unicodem i nowymi technologiami generowania fontów. Umożliwia korzystanie z fontów komputerowych w formatach OpenType i AAT (Apple Advanced Typography). Jego początki sięgają roku 2004. Więcej informacji o tym systemie jak i przykłady zastosowania w praktyce czytelnik znajdzie na stronie projektu scripts.sil.org/xetex

Wszystkie dostępne dzisiaj dystrybucje TeX-a umożliwiają także składanie dokumentów we wspomnianych wyżej formatach.

Adresatami tej książki są studenci kierunków ścisłych przygotowujący swoje prace licencjackie lub magisterskie, a także pracownicy naukowi sta-

jący przed koniecznością dostarczenia złożonego w LaTeX-u artykułu do czasopisma naukowego.

W obecnym, drugim wydaniu rozdział 2 został poszerzony o punkt opisujący składanie dokumentów fontem innym niż domyślny. Przedstawiono także obszernie całą rodzinę fontów TeXGyre wraz z przykładami wykorzystania w składanym dokumencie.

W rozdziale 3 dodano punkt „Składanie dużych dokumentów" zaś w rozdziale 4 punkt 1 dodano informację o fantomach i sposobie składania notek na marginesie. Rozdział 4 został też uzupełniony o punkt zawierający krótką instrukcję używania pakietu `listings`, pozwalającego na umieszczanie wewnątrz dokumentu fragmentów kodu napisanego w jednym z wielu dostępnych języków programowania.

W rozdziale 5 rozszerzono informacje dotyczące wstawiania tabel i rysunków do pliku tex-owego. Zostały obszernie omówione parametry środowisk `table` i `figure` i przedstawione przykłady użycia niektórych z nich.

Rozdział 6 został poszerzony o zestawienie wybranych symboli matematycznych, przydatnych w składaniu prac naukowych. Całość książki zamyka rozdział „Dodatek", w którym autorka zamieściła garść informacji o typografii, rodzajach pisma i zasadach składania tekstów w języku polskim.

W obecnym wydaniu zostały również poprawione wszystkie dostrzeżone błędy.

1. Jak zacząć

1.1. Instalacja systemu

Pracę z LaTeX-em należy zacząć od zainstalowania w komputerze całego systemu TeX. TeX jest dostępny dla wszystkich najczęściej spotykanych systemów operacyjnych a wybór instalacji zależy od posiadanego systemu. Najbardziej znanymi dystrybucjami dla Microsoft Windows są MiKTeX i TeXLive.

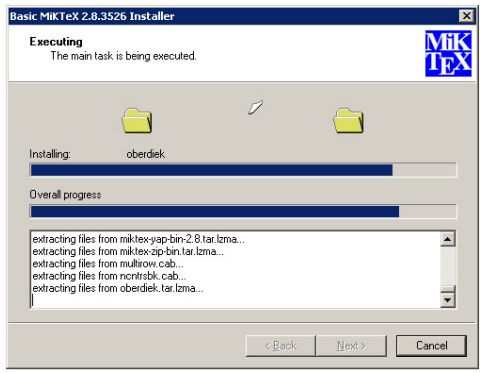
MiKTeX jest dystrybucją dla Microsoft Windows, dla systemów Linux jest testowana na razie wersja beta. Na MiKTeX-u oparta jest także dystrybucja TeX-a dla Microsoft Windows o nazwie proTeXt.

W przypadku TeXLive istnieją dystrybucje dla wszystkich używanych dzisiaj wersji Windows a także dla następujących systemów operacyjnych:

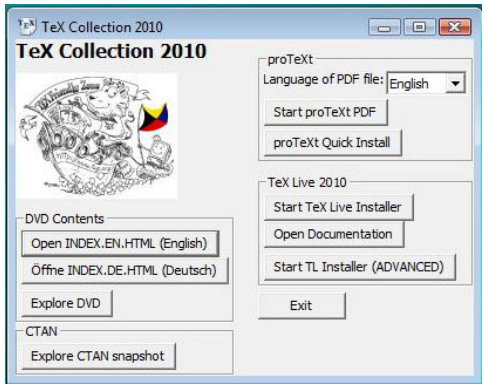
Arch Linux,	Debian,
Fedora,	Gentoo,
NetBSD,	OpenBSD,
Slackware,	Ubuntu.

Wszystkie dystrybucje zawierają graficzny instalator systemu lub dokładną informację o sposobie instalacji. Należy zwrócić szczególną uwagę na proponowany do zainstalowania zestaw pakietów. Najlepiej wybrać opcję „full", wówczas będziemy mogli korzystać z większości istniejących pakietów i programów wspomagających pracę TeX-a. Jest to ważne szczególnie przy instalowaniu MiKTeX-a, gdyż domyślnie instaluje on tylko pakiety i progra-

my podstawowe, pozostałe w razie potrzeby doinstalowuje z repozytorium poprzez internet.



Okno instalatora MiKTEX-a



Okno instalatora TEXLive

Użytkownicy komputerów z systemem MacOS X mogą też zaistalować MacTEX oparty na TEXLive.

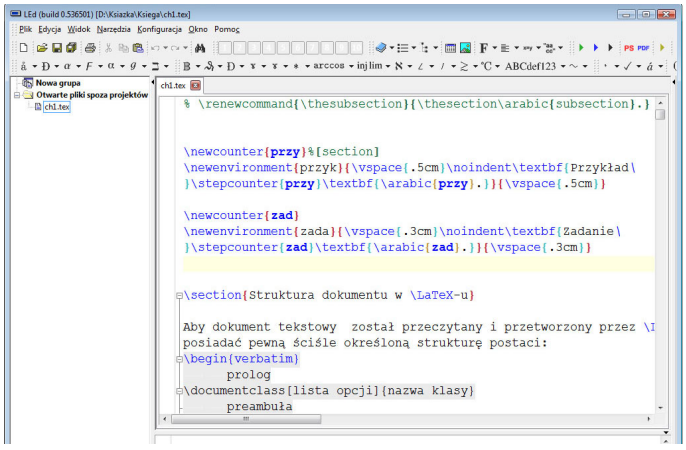
1.2. Wybieramy edytor

Tekst przeznaczony do złożenia przygotowujemy w dowolnym edytorze zgodnie z regułami opisanymi w makrach i zapisujemy go w kodzie ASCII jako *nazwa.tex*, bo tylko pliki z rozszerzeniem *.tex* są czytane a następnie kompilowane przez program. Nazwa którą nadamy swojemu dokumentowi powinna być możliwie krótka, nie powinna zawierać polskich liter (i żadnych innych znaków diakrytycznych) oraz znaków specjalnych TEX-a. Najnowsze dystry-

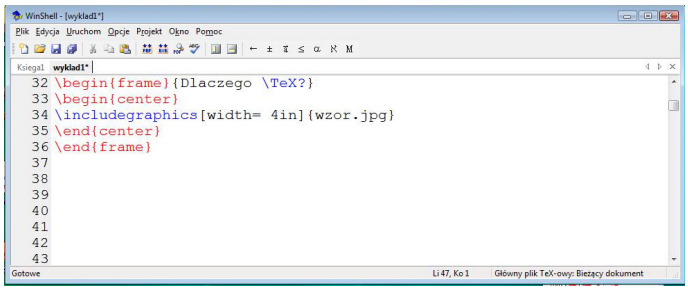
bucje TEXLive pozwalają na kompilowanie plików o długich nazwach, nawet ze spacjami.

Do składania tekstu wygodnie jest używać edytorów specjalnie przystosowanych do pracy z TEX-em, zapisują one pliki w odpowiednim formacie, a większość z nich udostępnia także szeroki wachlarz podpowiedzi składni środowisk latexowych oraz oznaczeń matematycznych. Takie edytory istnieją dla wszystkich systemów operacyjnych, można korzystać z edytorów darmowych, na licencji share-ware lub komercyjnych.

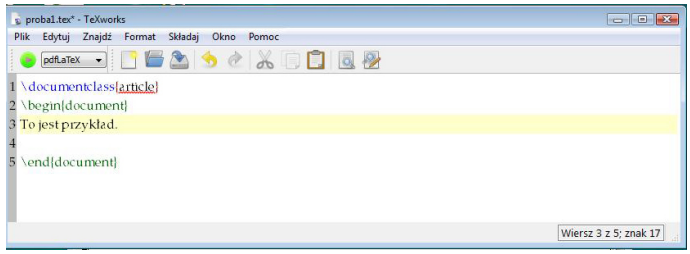
Aby pokazać czytelnikowi różnorodność dostępnych edytorów, umieściliśmy poniżej przykładowe ekrany darmowych edytorów uruchomionych w systemie MS Windows. Wszystkie one są również dostępne w systemach linuxowych. Każdy z pokazanych edytorów umożliwia sprawdzenie pisowni pod warunkiem doinstalowania słownika języka polskiego. Najprościej jest zainstalować słownik z pakietu Open Office.



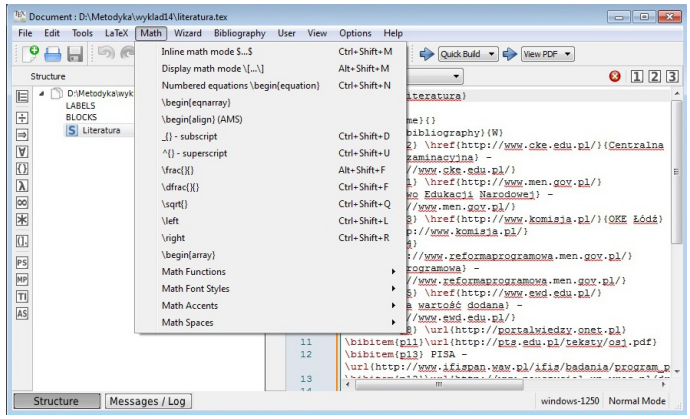
LeD



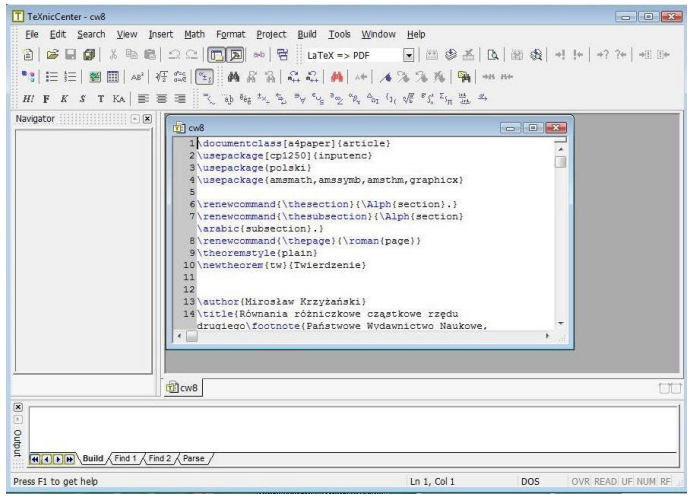
WinShell



TEXworks

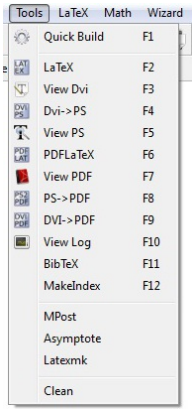


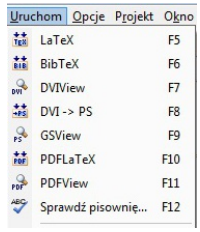
TEXmaker



TEXnicCenter

Każdy edytor dedykowany TEX-owi dostarcza też narzędzi do szybkiego uruchamiania kompilatora.





W zależności od tego w jakim formacie chcemy otrzymać plik końcowy uruchamiamy LaTeX-a lub PDFLaTeX-a.

Edytor TEXworks jest powiązany z przeglądarką plików pdf o nazwie PSV, różną od `Acrobat Readera`, która ma dwie bardzo wygodne własności:

- po kompilacji do pdf-a zawartość ekranu odświeża się automatycznie,
- umożliwia poruszanie się pomiędzy źródłem a dokumentem końcowym.

Jest to szczególnie wygodne w przypadku tworzenia dokumentów składających się z wielu rozdziałów, których treść znajduje się w osobnych plikach. Podobną właściwość ma także TeXmaker.

Po utworzeniu w wybranym edytorze pliku dokumentu dobrze jest zapisać go w osobnym katalogu. Podczas kompilacji, oprócz pliku końcowego tworzonych jest kilka plików pomocniczych które, w celu uniknięcia bałaganu, powinny być zapisane w tym samym katalogu co plik wyjściowy. Zaleca się, aby nazwa takiego katalogu nie zawierała znaków diakrytycznych ani spacji, gdyż może się zdarzyć, szczególnie w starszych wersjach systemu, że nagle program przestanie działać. Jeśli poprzednia kompilacja przebiegała pomyślnie i nie dokonaliśmy żadnych istotnych zmian to w sytuacji gdy program „odmawia współpracy” należy sprawdzić, czy katalog lub plik nie zawierają znaków diakrytycznych a następnie usunąć plik z rozszerzeniem .aux. Po wykonaniu tych zabiegów kompilacja powinna zakończyć się pomyślnie.

1.3. Przebieg kompilacji

Użytkownik LaTeX-a ma do wyboru dwa kompilatory: *latex* i *pdflatex*. Od wybranego kompilatora zależy format pliku końcowego.

latex interpretuje dostarczone informacje i tworzy plik *nazwa.dvi* (device independent) czyli niezależny od urządzenia, który możemy obejrzeć i wydrukować korzystając z programu `dviout` lub jego „starszego brata" `windvi`. Program `dviout` jest w każdej instalacji systemu TeX, jego poprzednika znajdziemy już tylko w starszych wersjach. Plik *nazwa.dvi* możemy też zamienić na plik postscriptowy za pomocą programu `dvips` (autor Thomas Rokicki), obejrzeć i wydrukować używając darmowej przeglądarki takich plików `GSview` (instalując tę przeglądarkę należy pamiętać, że do prawidłowego działania potrzebuje ona jeszcze programu `GhostScript`). Jeśli używany edytor nie udostępnia skrótu do tego programu, wówczas w linii poleceń należy napisać `dvips nazwa.dvi`.

pdflatex interpretuje dostarczone informacje i tworzy plik w formacie *pdf*, który możemy oglądać i wydrukować używając dowolnej przeglądarki plików pdf, na przykład programu `Adobe Acrobat Reader`.

W czasie jednokrotnej kompilacji *latex-em* (lub *pdflatex-em*) tworzone są między innymi pliki *nazwa.dvi* (lub *nazwa.pdf*) oraz pliki *nazwa.aux* i plik *nazwa.log*, oraz w nowych wersjach TeXLive plik *nazwa.synctex.gz*.

Plik *nazwa.log* zawiera informacje o przebiegu kompilacji, o wczytanych pakietach, przepełnionych lub niedopełnionych wierszach,

```
{c:/texlive/2010/texmf-dist/fonts/enc/dvips/cm-super/
cm-super-t1.enc}
ch2.tex(859):Overfull \hbox (2.42162pt too wide)
in paragraph at lines 859--873
Underfull \hbox(badness 10000)in paragraph at lines 10--15
```

a także o napotkanych błędach. Poniżej informacja o błędzie w wywołaniu środowiska `subequations` oraz o błędzie w nazwie instrukcji `quad`.

```
! Missing \endcsname inserted.
<to be read again>
                    \def
l.75 \end{subequations}}
```

```
! Undefined control sequence.
<argument>  f(x)&=2x+7 \guad
              x=4\\ f(x)&=x^2-2x+3 \\ f(x)&=x^3-2x^2
l.74 \end{align}
```

I jeszcze przykład informacji o konieczności zastosowania środowiska matematycznego

```
! Missing $ inserted.
<inserted text>
                $
l.83 \end{przyk}
```

Większość edytorów wskazuje miejsce w którym wystąpił błąd, po przerwaniu kompilacji umieszczając kursor w odpowiedniej linii tekstu i podając jej numer.

Jeśli w dokumencie znajdują się tabele i rysunki, to po pierwszym przebiegu utworzone zostaną pliki *nazwa.lot* (spis tabel) i *nazwa.lof* (spis rysunków), a po drugim przebiegu będą wypełnione informacjami o nich.

Uwaga: Aby w naszym dokumencie utworzyły się wszystkie spisy i odsyłacze, trzeba go skompilować co najmniej dwa razy.

2. Podstawowe informacje

2.1. Struktura dokumentu

Dokument, który będziemy składać za pomocą dowolnie wybranego edytora i kompilować *latex-em* lub *pdflatex-em*, musi posiadać ściśle określoną strukturę, na którą składają się:

```
prolog
\documentclass[lista opcji]{nazwa klasy}
preambuła
\begin{document}
Treść dokumentu
\end{document}
```

W `prologu` mogą się znaleźć na przykład polecenia wywołujące programy zewnętrzne wykonywane jeszcze przed uruchomieniem kompilatora TEX-a, ale początkujący użytkownik LATEX-a pozostawia prolog pusty.

Nazwa klasy dokumentu jest obowiązkowa. Informację o klasach dokumentu czytelnik znajdzie w dalszej części tego rozdziału.

Zauważmy, że argument `lista opcji` ujęty jest w nawiasy kwadratowe, a nazwa klasy umieszczona jest w nawiasach klamrowych. Regułą jest, że wszystkie wymagane parametry i opcje umieszczane są w nawiasach klamrowych, natomiast te nieobowiązkowe w nawiasach kwadratowych. Jeśli dana instrukcja lub polecenie może posiadać dodatkowe parametry, ale uruchamiamy je z parametrami domyślnymi, nie umieszczamy w ogóle nawiasów kwadratowych, na przykład:

```
\documentclass{article}
\usepackage[sumlimits,intlimits]{amsmath}
```

lub

```
\documentclass[a4paper,twoside]{article}
\usepackage{amsmath}
```

Jak widać klasa `article` może być wczytana z wieloma parametrami, ale możemy jej również użyć z parametrami domyślnymi. To samo dotyczy pakietu `amsmath`.

Niektóre deklaracje dopuszczają umieszczenie pustych nawiasów klamrowych jako wybór domyślnej akcji.

```
\newenvironment{przyklad}{\par\textbf{Przykład.␣}}{}
```

Zdefiniowaliśmy środowisko, które po wywołaniu `\begin{przyklad}` rozpocznie się pogrubionym wyrazem **Przykład.**, po kropce zostanie dodana spacja, a zakończenie środowiska poleceniem `\end{przyklad}` nie spowoduje żadnych dodatkowych działań, bo w jego definicji ostatnia para nawiasów klamrowych była pusta. Inny przykład:

```
\newenvironment{przyklad}{\par\textbf{Przykład.␣}}{\par}
```

Teraz po zamknięciu środowiska zostanie także zakończony bieżący akapit. Para nawiasów klamrowych, choć w niektórych przypadkach może być pusta, jest wymagana we wszystkich poleceniach i instrukcjach w których jest zdefiniowana. I tak na przykład w instrukcji `\bibitem{}` wprowadzającej każdą kolejną pozycję spisu literatury, pusta para nawiasów klamrowych oznacza, że nie zdefiniowano klucza do cytowania tej pozycji. Brak tych nawiasów spowoduje pojawienie się informacji o błędzie lub jako klucz zostanie przyjęta pierwsza litera treści pozycji bibliograficznej (i będzie ona pominięta w wydruku).

Preambuła dokumentu

Podczas uruchamiania LaTeX-a wczytywany jest podstawowy zestaw pakietów pozwalający na składanie tekstów, także matematycznych. Domyślne parametry graficzne strony oraz zestaw pakietów zależą od wybranej klasy dokumentu. Dodatkowe pakiety rozszerzające możliwości LaTeX-a wczytuje się poleceniem `\usepackage`. Wszystkie pakiety wczytywane bez parametrów bądź z parametrami domyślnymi, można wczytywać jednym poleceniem `\usepackage`, w nawiasach klamrowych umieszczając po kolei ich nazwy rozdzielone przecinkiem.

```
\usepackage[T1]{fontenc}
\usepackage{amssymb,amsmath,amsthm}
```

Pakiet to zbiór makrodefinicji ułatwiających tworzenie dokumentu.
Może go napisać każdy użytkownik LaTeX-a i wielu to czyni, a niektórzy je publikują umieszczając w internecie. Nie wszystkie istniejące pakiety mogą być wykorzystywane przez początkującego użytkownika systemu. Te o szerokich zastosowaniach są dostarczane z kolejnymi implementacjami TeX-a, inne trzeba instalować dodatkowo. W dystrybucji TeXLive dodatkowe pakiety najlepiej instalować w katalogu `texmf-local`, który nie jest usuwany podczas odinstalowywania systemu.

W preambule umieszcza się także wszystkie definicje i deklaracje użytkownika, które mają obowiązywać w całym dokumencie i mają wpływ na jego końcowy wygląd. Mogą tam się więc znaleźć takie deklaracje jak `\newcommand`, `\renewcommand`, `\newenvironment` itd. Tam też możemy umieścić tytuł dokumentu oraz nazwisko i adres autora.

- Polecenie `\usepackage` może być umieszczone tylko w preambule dokumentu.
- Nie wstawia się odstępu pomiędzy znakiem ukośnika wstecznego \ a nazwą polecenia!

- Nie wstawia się odstępów pomiędzy `\begin` i nazwę środowiska. Niektóre nowe pakiety dopuszczają taki zapis, ale generalna zasada brzmi: nie ma odstępów w tych miejscach.

Pakiety które posiadają dodatkowe opcje lub argumenty muszą być wczytywane osobno, tak jak na przykład `inputenc`, który jako opcję ma rodzaj kodowania danego dokumentu (zależy od systemu operacyjnego: w systemie Windows to cp1250, w Linux – utf8). Niektóre nowsze edytory pozwalają też na zapisanie dokumentu w kodowaniu innym niż posiadanego systemu operacyjnego zawsze jednak na początku dokumentu musi znaleźć się informacja o rodzaju kodowania.

Kolejność wczytywania pakietów nie jest obojętna. Część pakietów nie tylko wprowadza nowe polecenia, ale często zmienia definicje poleceń już istniejących (lub wczytanych razem z innym pakietem). Zdarza się też, że dwa pakiety „kłócą się" ze sobą i po wczytaniu obu, w procesie kompilacji otrzymujemy komunikaty o wystąpieniu dziwnych błędów. Takie dwa pakiety to na przykład `asmthm` i `theorem`.

Uwaga: Należy przestrzegać zasady, że preambuła powinna zawierać tylko te pakiety, które są konieczne do złożenia bieżącego dokumentu.

Klasa dokumentu

Każdy dokument LaTeX-owy rozpoczyna się od zadeklarowania klasy poleceniem

```
\documentclass[lista opcji]{nazwa klasy}
```

Klasa dokumentu to plik `nazwa klasy.cls`, zawierający definicje i deklaracje różnych parametrów dotyczących strony, m. in. szerokości i wysokości tekstu, rodzaju i rozmiaru czcionek w tytulariach itp. Można używać standardowych klas LaTeX-a: `article, book, report i letter` lub napisać swoją własną. Większość czasopism naukowych przygotowało własne

klasy dokumentów i udostępnia je autorom artykułów na swoich stronach internetowych. Dla polskich użytkowników powstały klasy: `mwart, mwbk i mwrep`, które są odpowiednikami amerykańskich `article, book, report` i uwzględniają polskie zwyczaje w piśmiennictwie naukowym.

Argument `lista opcji` jest nieobowiązkowy, ale może zawierać informacje o wielkości czcionki, jaką ma być składany cały dokument, rozmiarze papieru, sposobie drukowania (np. dokument jednostronny, dwustronny, ze stroną tytułową lub bez) itd. Lista opcji może być różna dla różnych klas, jeżeli będzie pusta to przyjęte zostaną wartości i ustawienia domyślne dla danej klasy.

10pt, 11pt, 12pt	– rozmiar czcionki
a4paper, a5paper, **letter**	– rozmiar papieru (wartość domyślna zależy od użytej klasy)
onecolumn, twocolumn	– jedna lub dwie kolumny tekstu na stronie
oneside, twoside	– druk po jednej lub po dwóch stronach
openright, openany	– tylko w klasie `book` – określa, czy nowy rozdział ma zawsze zaczynać się na stronie nieparzystej (openright) czy na kolejnej (openany)
notitlepage, titlepage	– zmiana domyślnych ustawień dotyczących strony tytułowej
leqno	– numery wzorów po lewej stronie tekstu
fleqn	– samodzielne wzory są wyrównywane do lewej strony tekstu (zamiast centralnie), z odpowiednim wcięciem kontrolowanym przez instrukcję `\mathindent`
draft, **final**	– wydruk roboczy lub finalny

Tabela 2.1: Standardowe opcje klasy

W tabeli 2.1 wartości domyślne w klasie `article` zostały wyróżnione.

Treść dokumentu

Treść dokumentu umieszczamy pomiędzy poleceniami `\begin{document}` i `\end{document}`. Tekst musi być napisany zgodnie z zasadami obowiązującymi w LaTeX-u. Zasady te przedstawimy w kolejnych rozdziałach.

2.2. Akcenty, znaki specjalne

Przygotowując dokument w LaTeX-u należy pamiętać, że pewne znaki zostały zarezerwowane do specjalnych celów. Są to:

```
#,  $,  %,  &,  \,  ^,  _,  {,  }.
```

Ukośnik wsteczny \ (zwany czasem w-tył-ciach, ang. backslash) rozpoczyna każdą instrukcję lub polecenie TeX-a, znak procentu (`%`) oznacza początek komentarza i działa do końca linii, znak dolara (`$`) otwiera i zamyka środowisko matematyczne, znaki ^ i _ oznaczają odpowiednio indeks górny i dolny, zaś nawiasy `{,}` określają zakres działania danego polecenia. Znak `&` używany jest do wskazania przejścia do następnej komórki w wierszach tabeli. Używa się go także do wskazania miejsca wyrównania wzorów w środowisku wieloliniowym. Znaki ^, _ i `&` mogą być użyte tylko w środowisku matematycznym. Znak # używany jest w definicji nowego polecenia lub środowiska do oznaczenia miejsca wstawienia parametru.

Każdy ze znaków specjalnych może być użyty w tekście pod warunkiem, że poprzedzimy go znakiem ukośnika wstecznego \ . Wyjątek stanowi sam znak \ – wywołujemy go poleceniem `$\backslash$`. Znaki dolara na początku i końcu polecenia informują, że jest ono zdefiniowane w trybie matematycznym.

Na klawiaturze komputera nie znajdziemy niektórych przydatnych znaków. Dla nich zostały zdefiniowane specjalne instrukcje (patrz tabela 2.2).

Czasami w tekście lub w spisie literatury musimy umieścić pojedyncze wyrazy pochodzące z innych języków niż polski i zawierające litery z różnymi

$ = \$	& = \&	% = \%	# = \#
_ = _	{ = \{	} = \}	
§ = \S	† = \dag	‡ = \ddag	
¶ = \P	© = \copyright	£ = \pounds	

Tabela 2.2: Znaki specjalne

akcentami. W tabeli 2.3 zebraliśmy instrukcje, przydatne do umieszczania różnych akcentów nad lub pod literami. Zastosowano je do liter e i o, ale można ich używać z dowolnymi literami w zależności od potrzeb.

Jeśli musimy użyć języka i alfabetu rosyjskiego, powinniśmy zmienić „na chwilę" kodowanie naszego dokumentu, wybrać właściwą rodzinę fontów (cyrylicę) i zapisać żądaną treść tak jak pokazano w przykładzie 2.4. Więcej o „chwilowej" zmianie rodziny fontów piszemy w rozdziale 2.6.1..

\'e	é	\`e	è	\^o	ô	\~o	õ
\=o	ō	\.e	ė	\"o	ö	\ss	ß
\u{o}	ŏ	\v{e}	ě	\H{o}	ő	\c{o}	ǫ
\oe	œ	\OE	Œ	\ae	æ	\AE	Æ
\aa	å	\AA	Å	\k{e}	ę	\ss	ß
\o	ø	\O	Ø	\l␣	ł	\L␣	Ł
\i	ı	\j	ȷ	!`	¡	?`	¿

Tabela 2.3: Akcenty

¶ Przykład 2.1.

Język hiszpański:

¡Hola! ¿Qué haces? `!`Hola! ?`Qu\'e haces?`

Los niños duermen. `Los ni\~nos duermen.`

□

¶ Przykład 2.2.

Język francuski:

Parlez vous français?	`Parlez vous fran\ais?`
No, je suis desolé!	`No, je suis desol\'e!`
D'où viens-tu?	`D'o\`u viens-tu?`
Je viens de Pologne.	`Je viens de Pologne.`

□

¶ Przykład 2.3.

Język fiński:

Hyvää huomenta!	`Hyv\"a\"a huomenta!`
Hyvää päivää!	`Hyv\"a\"a p\"aiv\"a\"a!`
Näkemiin!	`N\"akemiin!`

□

¶ Przykład 2.4.

W preambule dokumentu wczytujemy pakiet z fontami rosyjskimi `OT2` poleceniem `\usepackage[OT2,T1]{fontenc}` a następnie wpisujemy żądany tekst przełączając kodowanie na cyrylicę:

```
{\fontencoding{OT2}\fontfamily{wncyr}\selectfont
```

Здесь находится кириллический текст. А вот здесь э оборотное. Спасибо.

□

Oto jak powstał tekst po rosyjsku:

```
{\fontencoding{OT2}\fontfamily{wncyr}\selectfont
Zdes\cyrsftsn\ nakhodit\null sya kirillicheski\U{i}
tekst. A~vot zdes\cyrsftsn\ \cyrerev\ oborotnoe. Spasibo.}
```

Do składania całych dokumentów w językach obcych wykorzystujemy specjalne pakiety, które kontrolują wszystkie tytularia i przenoszenie wyrazów właściwe dla danego języka. Z instrukcji zamieszczonych w tabeli 2.3

korzystamy tylko do zapisania pojedynczych obcych wyrazów. Informacje jak składać dokumenty w języku polskim czytelnik znajdzie w rozdziale 2.4.

2.3. Tryby pracy LaTeX-a

Podczas pracy LaTeX znajduje się w jednym z trzech trybów: akapitowym, matematycznym i LR (left-to-right) czyli wierszowym.

W trybie akapitowym LaTeX znajduje się podczas składania tekstu. Kompilator najpierw „czyta" cały akapit a następnie dzieli go na wiersze, potem to samo dzieje się z następnymi akapitami aż w pamięci będzie już wystarczająca ilość tekstu do zapełnienia strony. Wtedy następuje łamanie strony. Dzięki takiej organizacji łamania udaje się uniknąć niechcianych odstępów np. między wyrazami.

Jeżeli w trakcie czytania treści dokumentu LaTeX natrafi na tekst matematyczny (zawarty np. pomiędzy znakami dolara), wtedy automatycznie przełącza się na tryb matematyczny.

Na tryb LR przełącza się zawsze wtedy, gdy napotka jedno z poleceń tworzących pudełka takie jak `\parbox{}{}` lub `\mbox{}`. W trybie LR tekst nie jest formatowany i sami musimy zadbać o łamanie wierszy.

2.4. Dokument w języku polskim

W dokumencie składanym w języku polskim wygodnie jest polskie litery wprowdzać bezpośrednio z klawiatury, po naciśnięciu prawego klawisza `Alt` i klawisza litery. Musimy zatem w preambule poinformować kompilator LaTeX-a o rodzaju używanego systemu operacyjnego, a tym samym kodowania. Najczęściej używanymi teraz systemami kodowania są: utf8 i cp1250 (MS Windows). O wybranym systemie kodowania informujemy kompilator poprzez wczytanie pakietu `inputenc` z właściwym dla danego kodowania argumentem. Preambuła dokumentu musi więc mieć jedną z dwu postaci:

```
\documentclass{mwart}
\usepackage[cp1250]{inputenc}
\usepackage{polski}
\begin{document}
```

```
\documentclass{mwart}
\usepackage[utf8]{inputenc}
\usepackage{polski}
\begin{document}
```

W systemie Linux dopuszczalne jest tylko druga postać, podczas gdy w systemie MS Windows możemy użyć obu, jeśli nasz edytor na to pozwala.

Wraz z pakietem `inputenc` wczytywany jest plik z tablicą konwersji znaków właściwą dla strony kodowej używanego systemu operacyjnego. Plik ten zawiera informacje, jaki kod binarny odpowiada konkretnemu znakowi, czyli jaki znak pojawi się w dokumencie kiedy na przykład napiszemy go z prawym klawiszem `Alt`.

Do niedawna kody binarne znaków były kodowane na bajtach maksymalnie ośmiobitowych, co dawało możliwość opisania maksymalnie 256 różnych znaków. Ponieważ TEX szybko stał się popularny w wielu krajach świata, powstała potrzeba utworzenia różnych stron kodowych przydatnych do wprowadzania znaków z różnych języków. Problem wielości stron kodowych został rozwiązany po wprowadzeniu kodowania Unicode. Najnowsze implementacje TEXLive i MiKTEX umożliwiają używanie kodowania `utf8` również w systemie MS Windows. W takim przypadku należy odpowiednio zmienić rodzaj kodowania w edytorze.

Pakiet `polski` zamienia na polskie wszystkie napisy, które są automatycznie generowane przez LATEX-a, np. angielski Abstract zamienia na Streszczenie, Chapter na Rozdział itd. Zmieniana też jest definicja makra `\today`, które wstawia datę sformatowaną według reguł języka polskiego i z polskimi nazwami miesięcy. Wprowadza również kilka zmian w składzie matematyki, na przykład definiuje nowe makra dla funkcji trygonometrycznych tg –

`\tg` oraz przedefiniowuje makra funkcji cyklometrycznych arc tg – `\arctan` a także po wczytaniu pakietu z opcją `plmath` wprowadza polskie znaki *mniejsze-większe* (⩽). Opcja `nomathsymbols` blokuje spolonizowanie przez pakiet standardowych symboli LaTeX-owych.

2.4.1. Dokument wielojęzyczny – pakiet `babel`

Do składania dokumentu w języku polskim możemy także użyć pakietu `babel` z opcją `polish`. Pakiet ten jest wygodny szczególnie wtedy, gdy w dokumencie będziemy umieszczać teksty także w innych językach, na przykład jeden rozdział po polsku a drugi po angielsku lub niemiecku. Jeśli korzystamy z pakietu `babel`, polskie litery musimy wprowadzać w sposób pokazany poniżej. Zdanie – Zażółć gęślą jaźń – zawiera wszystkie polskie litery ze znakami diakrytycznymi.

```
\documentclass{article}
\usepackage[polish]{babel}
\begin{document}
Za"r"o"l"c  g"e"sl"a  ja"z"n
\end{document}
```

W powyższym przykładzie wszystkie litery są poprzedzone znakiem cudzysłowu, a litera 'ż' powstaje z poprzedzonej cudzysłowem litery 'r'.

Jeżeli cały dokument ma być napisany np. po niemiecku, wystarczy w opcji klasy umieścić „german" oraz wczytać pakiet `babel`. Preambuła tego dokumentu wygląda tak:

```
\documentclass[german]{article}
\usepackage{babel}
\usepackage[T1]{fontenc}
```

lub tak:

```
\documentclass{article}
\usepackage[german]{babel}
\usepackage[T1]{fontenc}
```

Jeżeli dokument ma być wielojęzyczny, to preambuła musi wyglądać tak:

```
\documentclass{report}
\usepackage[cp1250]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[polish,german,french]{babel}
\usepackage{polski,amsmath}
\begin{document}
```

Język zmieniamy za każdym razem używając polecenia
`\selectlanguage{język}`

```
\begin{document}
\selectlanguage{german}
Tekst po niemiecku
\selectlanguage{french}
Tekst po francusku
\selectlanguage{polish}
Tekst po polsku
\selectlanguage{german}
I znowu tekst po niemiecku
```

2.5. Jednostki miary

TEX wie co oznaczają tradycyjne jednostki miary używane przez drukarzy takie jak „punkt" czy „pica", a także jednostki metryczne takie jak cal czy centymetr. Wynik pracy TEX-a oparty jest na systemie metrycznym. W tabeli 2.4 wymieniono większość używanych w TEX-u jednostek.

Zawsze gdy w LATEX-u określamy samodzielnie jakąś odległość musimy pamiętać o konieczności podania jednostki. Na przykład by otrzymać odległość zerową należy napisać `0pt` lub `0cm`.

Do zmiany dowolnej odległości służy polecenie `\setlength`. Domyślne wielkości, które możemy w ten sposób zmienić to między innymi:

cm	centymetr,
mm	milimetr,
in	cal (1 in = 2,54 cm),
pt	punkt (1 in = 72,27 pt),
bp	duży punkt (1 in = 72 bp),
em	szerokość litery M w bieżącym foncie,
ex	wysokość litery x w bieżącym foncie.

Tabela 2.4: Stałe jednostki miary rozumiane przez TEX-a

`\parindent`	–	wielkość wcięcia akapitowego,
`\parskip`	–	wielkość odstępu pomiędzy akapitami,
`\textwidth`	–	szerokość tekstu konkretnego akapitu,
`\textheight`	–	wysokość tekstu,
`\itemsep`	–	odstęp pomiędzy punktami listy.

¶ Przykład 2.5.

TEX powstał ponad 30 lat temu w Stanach Zjednoczonych.Komputery końca lat siedemdziesiątych ubiegłego wieku dzieli od używanych dzisiaj ogromna przepaść gdyż przez te lata nastąpił ogromny postęp technologiczny tak w odniesieniu do sprzętu jak i oprogramowania.

W ciągu kilku lat od powstania system rozprzestrzenił się na inne kraje i od razu pojawił się problem języków różnych od angielskiego. Funkcjonalność i możliwości TEX-a okazały się niewystarczające. □

W przykładzie 2.5 zwiększyliśmy odstęp pomiędzy akapitami. Oba akapity zostały zgrupowane w nawiasach klamrowych i wewnątrz nich użyliśmy polecenia

```
{\setlength{\parskip}{6ex plus 0.5ex minus 0.3ex}
treść akapitu 1

treść akapitu 2 }
```

Odległość pomiędzy akapitami ustaliliśmy na 6 ex, ale mogła ona być przez kompilator zwiększona o 0,5 ex lub zredukowana do 2,7 ex w trakcie rozmieszczania tekstu na stronie.

Szerokość tekstu dowolnego akapitu możemy zmienić umieszczając przed jego rozpoczęciem polecenie

```
\setlength{\textwidth}{10cm}
```

Aby po zakończeniu akapitu powrócić do poprzedniej szerokości tekstu, cały akapit należy zgrupować używając nawiasów klamrowych, jak w przykładzie 2.5.

Szerokość tekstu w całym dokumencie ustalamy w preambule

```
\textwidth=14cm
```

Niektóre polecenia określające odległości „oczekują" deklaracji rozciągliwości, czyli podania wartości o którą dana wielkość może być powiększona lub zmniejszona. Do nich należy, między innymi, wspomniany już `\parskip`, a także `\itemsep`. Specjalną, rozciągliwą instrukcją do stosowania wewnątrz dokumentu jest `\fill`. Jej naturalna długość jest równa zeru, ale może być dowolnie rozciągnięta.

2.6. Fonty

W miarę rozwoju technologicznego komputerów i ich oprogramowania użytkownicy TEX-a dostawali do używania z systemem coraz obszerniejsze zestawy fontów zawierających różne kroje liter.

W tym miejscu należy wyjaśnić, co rozumiemy przez słowo font. Font to jest plik, w którym zapisano kształty i inne informacje o literach danego

kroju. W czasach przed erą komputerową, każda litera lub znak drukarski musiała mieć swój wzorzec, przeniesiony z projektu typografa na metalowy prostopadłościan, zwany czcionką. Z tych czcionek składano wyrazy, z nich wiersze, akapity, strony i odbijano je (drukowano) coraz efektywniejszymi metodami, na papierze.

W każdej drukarni znajdowały się zestawy czcionek różnych krojów, a w każdym takim zestawie można było znaleźć między innymi wszystkie używane w danym języku litery, znaki interpunkcyjne, nawiasy itp. w różnych przydatnych rozmiarach (wielkościach). Taki zestaw czcionek danego kroju nazywany był przez drukarzy *pismem* i zawierał także czcionki odstępów poziomych i pionowych.

Teraz wszystkie te informacje umieszczane są w plikach komputerowych nazywanych fontami. Pojedynczy font to najczęściej zestaw czcionek danego kroju i jego odmiany. Możemy przyjąć, że font (czcionka) ma trzy wyróżniające go parametry:

krój (ang. typeface)	–	odpowiada za wygląd czcionki,
odmiana	–	określa grubość linii w literach oraz (lub) stopień pochylenia liter,
stopień	–	określa wielkość liter.

Rodzina fontów to zbiór wszystkich odmian danego fontu. Odmiany to na przykład: light, regular, italic, medium italic, medium bold, bold italic, medium italic, light italic. Nie każda rodzina fontów składa się ze wszystkich wymienionych odmian.

Wszystkie kroje pisma w których litery zbudowane są z kresek jednakowej szerokości nazywamy krojami *jednoelementowymi* (inaczej linearnymi). Kroje pisma w których litery są zbudowane z kresek różnej szerokości nazywamy krojami *dwuelementowymi*. Kroje można także podzielić na *szeryfowe* np. W K i *bezszeryfowe* np. W K lub na kroje *stałe* `monolit` i *proporcjonalne* monolit.

Pliki zawierające fonty mogą występować w wielu różnych formatach. Obecnie trzy najpopularniejsze formaty to Type 1 (T1) (fonty postscriptowe), TrueType (TT) i OpenType (OTF).

TEX Knutha „rozumie" tylko fonty kodowane w specjalnym formacie. Do pracy potrzebuje informacji metrycznych o rozmiarach znaków, ligaturach i wielkości kernów (podcięć) dla par znaków. Wszystkie te informacje zawarte są w plikach TFM (TEX Font Metric). Aby przetworzyć dokument TEX-em (LATEX-em) potrzebne więc są pliki TFM. Dopiero sterowniki zamieniające tekst na postać widoczną na ekranie lub przygotowaną do wydruku potrzebują „prawdziwych" fontów. Więcej na temat fontów w TEX-u czytelnik znajdzie w artykule im poświęconym na stronie www.gust.org.pl/doc/fonts/abc.

2.6.1. Składanie dokumentu innym fontem niż domyślny

W standardowej instalacji systemu TEX znajdziemy, oprócz domyślnych, bogaty zestaw krojów. Aby z nich skorzystać musimy włączyć odpowiedni system kodowania. W wielu przypadkach wystarczy załadować pakiet `polski`, choć bogatszy podzbiór znaków dla niektórych fontów zapewni nam użycie układu `QX`. Poniżej podamy przykłady użycia fontów z rodzin `iwona`, `antykwa toruńska` i `antykwa półtawskiego`. Omówimy też pokrótce fonty z rodziny `tex-gyre`.

Pismo `iwona`

Aby złożyć dokument krojem `iwona` wystarczy w preambule załadować go poleceniem

```
\usepackage{polski}
\usepackage{iwona}
```

Dostęp do większego zbioru znaków tej rodziny zapewnimy sobie wczytując dodatkowo

```
\usepackage[QX]{fontenc}
```

Poniżej fragment tekstu złożony krojem `iwona`.

> Należy zauważyć, że środowisko `array` może być stosowane tylko w trybie matematycznym. *Wszystkie te* środowiska tworzą w dokumencie ministronę. Środowisko TABULAR tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach natomiast `tabular*` dysponuje **argumentem** umożliwiającym ustalenie jej szerokości.
>
> Teraz wszystkie te informacje umieszczane są w plikach komputerowych nazywanych fontami. Pojedynczy font to najczęściej zestaw czcionek danego kroju i jego odmiany.

Jak łatwo zauważyć czcionka ta jest dwuelementową czcionką bezszeryfową. W powyższym fragmencie przedstawiliśmy też odmianę pogrubioną, pochyłą, kursywę i imitującą pismo maszynowe.

Aby użyć pisma `iwona` we wzorach matematycznych należy zadeklarować przy wywołaniu pakietu opcję `math`.

```
\usepackage[math]{iwona}
```

Kurier

`Kurier` jest dwuelementową czcionką bezszeryfową. Powstał jako pismo przeznaczone dla gazet i innej prasy do składania techniką linotypową. Aby tekst złożyć fontem `kurier` wystarczy w preambule dokumentu napisać:

```
\usepackage{polski}
\usepackage{kurier}
\usepackage[QX]{fontenc}
```

Poniżej fragment tekstu złożony krojem `Kurier`.

Należy zauważyć, że środowisko `array` może być stosowane tylko w trybie matematycznym. *Wszystkie te* środowiska tworzą w dokumencie ministronę. Środowisko TABULAR tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach natomiast `tabular*` dysponuje **argumentem** umożliwiającym ustalenie jej szerokości.

Teraz wszystkie te informacje umieszczane są w plikach komputerowych nazywanych fontami. Pojedynczy font to najczęściej zestaw czcionek danego kroju i jego odmiany.

Pismo `Antykwa Toruńska`

Aby złożyć dokument **`Antykwą Toruńską`** trzeba w preambule załadować krój poleceniem

```
\usepackage{polski}
\usepackage{antt}
```

Poniżej fragment tekstu złożony krojem `Antykwa Toruńska`.

Należy zauważyć, że środowisko `array` może być stosowane tylko w trybie matematycznym. *Wszystkie te* środowiska tworzą w dokumencie ministronę. Środowisko TABULAR tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach natomiast `tabular*` dysponuje **argumentem** umożliwiającym ustalenie jej szerokości.

Teraz wszystkie te informacje umieszczane są w plikach komputerowych nazywanych fontami. Pojedynczy font to najczęściej zestaw czcionek danego kroju i jego odmiany.

Pismo Antykwa Półtawskiego

Aby złożyć dokument **Antykwą Półtawskiego** należy w preambule załadować krój poleceniem

```
\usepackage{polski}
\usepackage{antpolt}
```

Poniżej fragment tekstu złożony krojem Antykwa Półtawskiego.

> Należy zauważyć, że środowisko array może być stosowane tylko w trybie matematycznym. *Wszystkie te* środowiska tworzą w dokumencie ministronę. Środowisko TABULAR tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach natomiast tabular* dysponuje **argumentem** umożliwiającym ustalenie jej szerokości.
>
> Teraz wszystkie te informacje umieszczane są w plikach komputerowych nazywanych fontami. Pojedynczy font to najczęściej zestaw czcionek danego kroju i jego odmiany.

Pismo Antykwa Półtawskiego ma tylko odmianę prostą, półgrubą, pochyłą i pochyłą półgrubą, Antykwa toruńska ma odmian więcej (zainteresowanych odsyłamy do dokumentacji dostępnej w każdej instalacji systemu). Z tym pismem można też używać fontów matematycznych specjalnie zaprojektowanych dla niego. Przy wywoływaniu pakietu należy zadeklarować opcję math.

```
\usepackage[math]{anttor}
```

Pismo Cyklop

Cyklop jest czcionką dwuelementową powstałą w latach dwudziestych XX wieku, w pracowni projektowej warszawskiej „Odlewni Czcionek J. Idźkowski i S-ka". W postaci czcionek ołowianych, produkowany był w odmianie kursywnej w rozmiarach 8-48 pt. Używany był w tytulariach gazetowych

i drukach akcydensowych. Elektroniczną postać fontów wygenerował Janusz M. Nowacki, uzupełniwszy je o odmianę prostą. Aby złożyć dokument krojem `Cyklop` należy w preambule załadować go poleceniem

```
\usepackage{polski}
\usepackage{cyklop}
```

Poniżej fragment tekstu złożony krojem `Cyklop`.

Należy zauważyć, że środowisko `array` **może być stosowane tylko w trybie matematycznym. *Wszystkie te* środowiska tworzą w dokumencie ministronę. Środowisko TABULAR tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach natomiast** `tabular*` **dysponuje argumentem umożliwiającym ustalenie jej szerokości.**

Łatwo można zauważyć, że `Cyklop` nie posiada odmiany naśladującej maszynę do pisania, zamiast została użyta odmiana z fontu domyślnego.

Aby fragment tekstu został złożony wybranym przez nas krojem, innym od używanego w dokumencie, należy go umieścić wewnątrz grupy.

```
{\fontfamily{nazwa}\selectfont ... tekst ...}
```

gdzie `nazwa` to

- w przypadku `iwony` – `iwona`,
- w przypadku `Kuriera` – `kurier`,
- w przypadku `Antykwy Toruńskiej` – `antt`,
- w przypadku `Antykwy Półtawskiego` – `antp`,
- w przypadku `Cyklopa` – `cyklop`.

Dotyczy to wszystkich opisywanych rodzin fontów.

2.6.2. Fonty TEXGyre

Fonty TEXGyre to zbiór następujących krojów: tgtermes, tgheros, tgschola, tgchorus, tgadventor, tgbonum, tgcursor, tgpagella.

Deklaracja `\usepackage{tgtermes}` w preambule dokumentu ustala krój TEXGyre Termes jako domyślny font szeryfowy dla całego dokumentu, czyli kroju skojarzonego z poleceniem `\rmfamily`. W dokumentach składanych po polsku konieczne jest wtedy uaktywnienie kodowania dodając opcję `[T1]` lub `[QX]` pakietu `polski`

```
\usepackage{tgtermes}
\usepackage[T1]{polski}
lub
\usepackage[QX]{polski}
```

Deklaracja `\usepackage{tgheros}` w preambule dokumentu ustala krój TEXGyre Heros jako domyślny font bezszeryfowy, czyli skojarzony z poleceniem `\sffamily`. W dokumentach składanych po polsku konieczne jest wtedy uaktywnienie kodowania `T1` lub `QX` na przykład w ten sposób

```
\usepackage{tgheros}
\usepackage[T1]{polski}
lub
\usepackage[QX]{polski}
```

Krój `\usepackage{tgschola}` – szeryfowy, skojarzony z poleceniem `\rmfamily` – ustala się wczytując

```
\usepackage{tgschola}
\usepackage[T1]{polski}
lub
\usepackage[QX]{polski}
```

Krój TEXGyre Chorus – szeryfowy, skojarzony z poleceniem `\rmfamily` – ustala się wczytując

```
\usepackage{tgchorus}
\usepackage[T1]{polski}
```

Krój TEXGyre Adventor – bezszeryfowy, skojarzony z poleceniem `\sffamily` – ustala się wczytując

```
\usepackage{tgadventor}
\usepackage[T1]{polski}
lub
\usepackage[QX]{polski}
```

Krój TEXGyre Bonum – szeryfowy, skojarzony z poleceniem `\rmfamily` – ustala się wczytując

```
\usepackage{tgbonum}
\usepackage[T1]{polski}
lub
\usepackage[QX]{polski}
```

Krój TEXGyre Cursor – maszynowy, skojarzony z poleceniem `\ttfamily` – ustala się wczytując

```
\usepackage{tgcursor}
\usepackage[T1]{polski}
lub
\usepackage[QX]{polski}
```

Krój TEXGyre Pagella – szeryfowy, skojarzony z poleceniem `\rmfamily` – ustala się wczytując

```
\usepackage{tgpagella}
\usepackage[T1]{polski}
lub
\usepackage[QX]{polski}
```

W dokumentach w których użyjemy jednego z opisywanych krojów szeryfowych zastąpi on domyślną antykwę i cały tekst będzie złożony tym krojem.

Jeśli chcemy mieć dostęp do jednego z krojów bezszeryfowych (na przykład fragment tekstu ma być tak właśnie napisany) wówczas powinniśmy dodatkowo wczytać odpowiedni pakiet w preambule, a fragment poprzedzić instrukcją `\sffamily`. Możemy też użyć sposobu opisanego poniżej.

Fragment tekstu składany wybranym fontem TEXGyre, innym od używanego w dokumencie, należy umieścić wewnątrz grupy.

```
{\fontfamily{nazwa}\selectfont ... tekst ...}
```

gdzie nazwa rodziny to w przypadku

- TEXGyre Pagella – `\fontfamily{qpl}`,
- TEXGyre Cursor – `\fontfamily{qcr}`,
- TEXGyre Bonum – `\fontfamily{qbk}`,
- TEXGyre Adventor – `\fontfamily{qag}`,
- TEXGyre Chorus – `\fontfamily{qzc}`,
- TEXGyre Schola – `\fontfamily{qcs}`,
- TEXGyre Heros – `\fontfamily{qhv}`,
- TEXGyre Heros Condensed – `\fontfamily{qhvc}}`,
- TEXGyre Termes – `\fontfamily{qtm}`.

Przykłady

We wszystkich przykładach użyjemy czcionki prostej, pogrubionej i kursywy.

1. Czcionka domyślna TEXGyre Adwentor – bezszeryfowa

   Należy zauważyć, że środowisko array może być stosowane tylko w trybie matematycznym. *Wszystkie środowiska tworzą* w dokumencie ministronę. Środowisko **tabular** tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach natomiast **tabular*** dysponuje **argumentem** umożliwiającym ustalenie jej szerokości. Parametr pozycja określa pionowe położenie tabeli.

2. Czcionka domyślna TEXGyre Termes – szeryfowa

   Należy zauważyć, że środowisko array może być stosowane tylko w trybie matematycznym. *Wszystkie te środowiska tworzą* w dokumencie ministronę. Środowisko **tabular** tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach natomiast **tabular*** dysponuje **argumentem** umożliwiającym ustalenie jej szerokości. Parametr pozycja określa pionowe położenie tabeli.

3. Czcionka domyślna TEXGyre Pagella – szeryfowa

   Środowisko array może być stosowane tylko w trybie matematycznym. *Wszystkie te środowiska tworzą* w dokumencie ministronę. Środowisko **tabular** tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach natomiast **tabular*** dysponuje **argumentem** umożliwiającym ustalenie jej szerokości. Parametr pozycja określa pionowe położenie tabeli.

4. Czcionka domyślna TEXGyre Cursor

   Należy zauważyć, że środowisko array może być stosowane tylko w trybie matematycznym. *Wszystkie te środowiska tworzą* w dokumencie ministronę. Środowisko **tabular** tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach, zaś **tabular*** dysponuje **argumentem** pozwalającym na ustalenie jej szerokości. Parametr pozycja określa pionowe położenie tabeli.

5. Czcionka domyślna TEXGyre Heros – bezszeryfowa

   Należy zauważyć, że środowisko array może być stosowane tylko w trybie matematycznym. *Wszystkie te środowiska tworzą* w dokumencie ministronę. Środowisko **tabular** tworzy tabelę o szerokości

zależnej od treści umieszczanych w komórkach natomiast **tabular*** dysponuje **argumentem** umożliwiającym ustalenie jej szerokości. Parametr pozycja określa pionowe położenie tabeli.

6. Czcionka domyślna TEXGyre Heros Condensed

   Należy zauważyć, że środowisko array może być stosowane tylko w trybie matematycznym. *Wszystkie te środowiska tworzą* w dokumencie ministronę. Środowisko **tabular** tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach natomiast **tabular*** dysponuje **argumentem** umożliwiającym ustalenie jej szerokości. Parametr pozycja określa pionowe położenie tabeli.

7. Czcionka domyślna TEXGyre Chorus

   Należy zauważyć, że środowisko array może być stosowane tylko w trybie matematycznym. Wszystkie te środowiska tworzą w dokumencie ministronę. Środowisko tabular tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach natomiast tabular dysponuje argumentem umożliwiającym ustalenie jej szerokości. Parametr pozycja określa pionowe położenie tabeli.*

8. Czcionka domyślna TEXGyre Schola

   Należy zauważyć, że środowisko array może być stosowane tylko w trybie matematycznym. *Wszystkie te środowiska tworzą* w dokumencie ministronę. Środowisko **tabular** tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach natomiast **tabular*** dysponuje **argumentem** umożliwiającym ustalenie jej szerokości. Parametr pozycja określa pionowe położenie tabeli.

9. Czcionka domyślna TEXGyre Bonum

   Należy zauważyć, że środowisko array może być stosowane tylko w trybie matematycznym. *Wszystkie te środowiska tworzą* w dokumencie ministronę. Środowisko **tabular** tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach

natomiast **tabular*** dysponuje **argumentem** umożliwiającym ustalenie jej szerokości. Parametr pozycja określa pionowe położenie tabeli.

Łatwo można zauważyć, że w TEXGyre Heros Condensed nie ma odmiany pogrubionej, tu zastosowana została czcionka niezwężona. TEXGyre Cursor również nie posiada odmiany pogrubionej, a TEXGyre Chorus jest zbliżony do kursywy i nie posiada ani oddzielnej kursywy ani odmiany pogrubionej.

3. Projektowanie dokumentu

3.1. Plan pracy

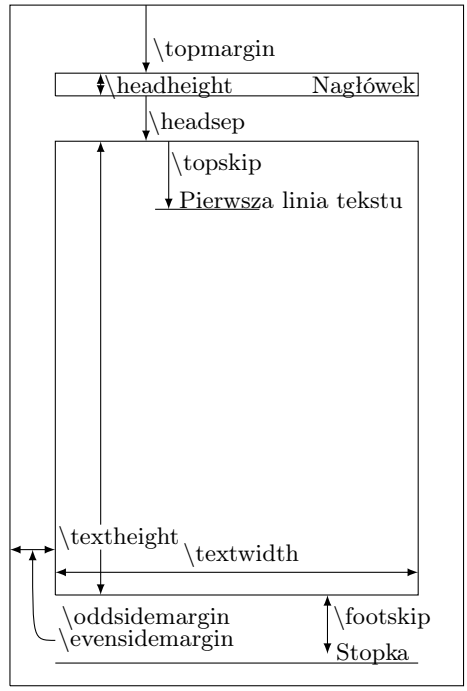
Przed przystąpieniem do pracy nad dokumentem należy zaplanować jego wygląd to znaczy:

1. Ustalić jaka będzie jego objętość (liczba stron).
2. Na jakie części będzie podzielony (rozdziały, punkty itd.).
3. Czy będzie drukowany jednostronnie czy dwustronnie, czy tekst będzie składany w jednej czy w dwóch kolumnach.
4. Ustalić rozmiar strony.
5. Ustalić rozmiar marginesów, wielkość i rodzaj czcionki oraz interlinii.
6. Zdecydować o wyglądzie stopki i nagłówka strony oraz sposobie numerowania stron.
7. Czy w dokumencie będziemy zamieszczać ilustracje i tabele.
8. Czy spis treści umieścimy na początku czy na końcu dokumentu.

Objętość dokumentu implikuje wybór klasy LaTeX-a której użyjemy do jego składania. Podstawowe klasy dokumentów to `article` – do składania krótkich artykułów naukowych, `report` – do dłuższych rozpraw naukowych oraz `book` – do składania książek. W każdej z tych klas zdefiniowane są podstawowe cechy strony (opisane w punktach 3–7) i ustawione jako domyślne.

Różnice pomiędzy wspomnianymi klasami zostały pokrótce opisane na stronie 48 i następnych.

Domyślne ustawienia strony, na przykład rozmiary marginesów, wielkość interlinii itd. możemy zmieniać używając poleceń zdefiniowanych w pakiecie `geometry`, a wygląd stopki i nagłówka na stronie używając pakietu `fancyhdr`. Oba te pakiety są dostarczane z każdą dystrybucją TEX-a, omówimy je w dalszych rozdziałach.



Rysunek 3.1: Parametry układu graficznego strony

Parametry strony opisane na rysunku 3.1, można zmienić także bez korzystania z pakietu `geometry`. Należy wtedy skorzystać z instrukcji `\setlength{nazwa}{wielkość}`.

1. `\oddsidemargin` – definiuje lewy margines, a w przypadku klasy `book` lub kiedy dokument jest składany dwustronnie, lewy margines stron nieparzystych. Jego wielkość zmieniamy instrukcją `\setlength`
2. `\evensidemargin` – definiuje lewy margines na stronach parzystych w przypadku klasy `book` lub kiedy tworzony dokument ma być składany dwustronnie,
3. `\topmargin` – odległość od nagłówka,
4. `\headheight` – ustawia wysokość nagłowka,
5. `\headsep` – odległość między nagłówkiem a tekstem dokumentu,
6. `\topskip` – odległość pierwszej linii tekstu od górnego marginesu,
7. `\textheight` i `\textwidth` – wysokość i szerokość tekstu,
8. `\footskip` – odległość końca tekstu od stopki,
9. `\paperwidth`, `\paperheight` – szerokość i wysokość papieru.

3.2. Części dokumentu

Użytkownik formatu LaTeX może korzystać z bogatego zestawu makr przydatnych w formatowaniu wizualnym, opisującym wygląd dokumentu i logicznym, dotyczącym jego logicznej struktury. Formatowanie wizualne polega na ustaleniu ogólnego wyglądu dokumentu, a więc np. wielkości i rodzaju liter, układu akapitów na stronie, wielkości marginesów itp. Formatowaniem wizualnym strony będziemy zajmować się w następnym rozdziale, teraz natomiast omówimy formatowanie logiczne. Operuje ono takimi pojęciami, jak tytuł rozdziału, tytuł punktu, tabela, tytuł tabeli, odsyłacz, przypis, notka na marginesie, spisy treści, tabel, rysunków, literatury itp. i ma zdecydowaną przewagę nad wizualnym, zwiększa wygodę pracy i jej wydajność.

Artykuł przygotowywany do druku w czasopiśmie naukowym składa się z tytułu, nazwiska autora, jego adresu, daty utworzenia (opcjonalnie), krót-

kiego streszczenia, treści dokumentu oraz spisu literatury. Niekiedy na dole pierwszej strony pojawiają się informacje o grantach lub podziękowania osobom lub instytucjom. Treść artykułu bywa podzielona na rozdziały, podrozdziały itp. Taki artykuł składamy najczęściej w klasie `article` lub równorzędnej `mwart` chyba, że wydawnictwo udostępniło autorom swoją klasę dokumentów.

W LaTeX-u tytuł, nazwisko autora, jego adres, datę utworzenia (opcjonalnie) i ewentualne podziękowania umieszcza się w preambule dokumentu wykorzystując następujące polecenia:

```
\documentclass{article}
\title{Tytuł dokumentu}
\author{Nazwisko autora \thanks{Podziękowania}\\ Jego adres}
\date{Data utworzenia}
\begin{document}
\maketitle
```

Aby wszystkie te polecenia zostały wykonane, należy umieścić instrukcję `\maketitle` bezpośrednio po `\begin{document}`. Polecenie `\thanks` umieszcza się w tekście poleceń `\author`, `\title` lub `\date`. Tekst podziękowań jest drukowany na dole pierwszej strony jako przypis, a w miejscu wstawienia polecenia `\thanks` pojawia się numer odsyłacza.

W klasie `article`, domyślnie, tytuł i nazwisko autora oraz streszczenie znajdują się na jednej stronie. Aby tytuł i nazwisko autora znalazły się na osobnej stronie, należy klasę dokumentu wczytać z parametrem `titlepage`.

```
\documentclass[titlepage]{article}
```

Streszczenie

Tekst streszczenia umieszczony w środowisku `abstract` i zostanie odpowiednio sformatowany i poprzedzony nazwą „Streszczenie”.

```
\begin{abstract}
treść streszczenia
\end{abstract}
```

W klasie `article` (`mwart`) streszczenie pojawia się bezpośrednio po nagłówku z tytułem i nazwiskiem autora, na pierwszej stronie dokumentu, w klasie `report` (`mwrep`) streszczenie umieszczane jest na osobnej stronie. Jeżeli w klasie `article` wybrana jest opcja `titlepage`, wtedy tytuł i nazwisko autora oraz streszczenie drukowane są na oddzielnych stronach. W klasie `book` streszczenie nie występuje.

Każdy dokument powinien być podzielony na logiczne części. Ułatwiają one czytanie i zrozumienie treści oraz poruszanie się po dokumencie. W klasie `article` (i w `mwart`) mamy następujące instrukcje hierarchizujące tekst:

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

W klasie `report` i `book` dodatkowo można użyć instrukcji `\part{...}` i `\chapter{...}`.

We wszystkich przypadkach LaTeX dobiera odpowiednie odstępy między rozdziałami (częściami), krój i wielkość pisma w śródtytułach oraz dba o ich numerację.

Spis treści

Spis treści generowany jest instrukcją `\tableofcontents` którą wstawiamy tam, gdzie ma się on pojawić. Zawiera tabelaryczny spis wszystkich numerowanych części dokumentu oraz numery stron, na których się one zaczynają. W krótkich dokumentach spis treści umieszcza się zazwyczaj na końcu lub nie generuje się go wcale. Rozdział bez numeru, na przykład „Wstęp”, wygenerowany poleceniem

```
\section*{Wstęp} lub \chapter*{Wstęp}
```

dodajemy do spisu instrukcją

```
\addcontentsline{toc}{section}{Wstęp}  lub
\addcontentsline{toc}{chapter}{Wstęp}
```

Aby cały spis został utworzony, należy dwukrotnie uruchomić kompilator.

Spisy tabel i rysunków

Jeśli w dokumencie występują tabele i rysunki, to ich spisy są generowane instrukcjami `\listoftables` i `\listoffigures` odpowiednio. Aby spisy tabel i rysunków pojawiły się w spisie treści, należy odpowiednie instrukcje `\addcontentsline{toc}{}{}` umieścić bezpośrednio po instrukcji generującą odpowiedni spis. W tym przypadku dokument także musi być przetworzony dwukrotnie.

Spis literatury

Każda praca naukowa powinna zawierać spis literatury. W zależności od klasy dokumentu, spis literatury umieszczany jest na osobnych stronach lub bezpośrednio po zakończeniu artykułu. W klasie `article` spis literatury umieszczany jest bezpośrednio po tekście artykułu i zatytułowany jest `Literatura`, w klasie `report` na kolejnej stronie, a w klasie `book` na następnej stronie nieparzystej. W obu tych klasach ma tytuł `Bibliografia`.

Spis literatury umieszczamy w środowisku `thebibliography`, a kolejne jego pozycje rozpoczynamy instrukcją `\bibitem`:

```
\begin{thebibliography}{przykładowa etykieta}
\bibitem[etykieta]{klucz} autor, tytuł itd.
.........
\end{thebibliography}
```

Bez opcjonalnego argumentu `etykieta`, jako etykiety poszczególnych pozycji w spisie literatury LaTeX umieszcza kolejne numery ujęte w nawiasy

kwadratowe. Jeżeli chcemy, by kolejne pozycje spisu były oznaczane etykietami literowymi, wówczas każdą taką etykietę umieszczamy w miejscu argumentu `etykieta`.

Spis będzie poprawnie sformatowany, gdy jako argument `przykładowa etykieta` umieścimy najdłuższą kombinację liter występujących jako etykiety kolejnych pozycji literatury.

Jeżeli kolejne pozycje są numerowane, to jako argument `przykładowa etykieta` wstawiamy cyfrę 9, gdy pozycji literatury jest mniej niż 10, liczbę 99, gdy pozycji literatury jest mniej niż 100, i tak dalej.

Obowiązkowy argument `klucz` może składać się z dowolnej kombinacji liter, cyfr i znaków, z wyjątkiem znaku przecinka. Chociaż w dokumencie można umieścić dowolną ilość spisów literatury, zwykle tworzy się tylko jeden.

Odwołania do spisu literatury

Możliwość przytoczenia numeru pozycji bibliograficznej znajdującej się w spisie literatury zapewnia nam obecność w instrukcji `\bibitem` obowiązkowego argumentu `klucz`. Aby w tekście dokumentu odwołać się do konkretnej pozycji, argument `klucz` umieszczamy w instrukcji `\cite{klucz}`.

Jeśli w spisie literatury umieściliśmy pozycję w sposób jak poniżej:

```
\begin{thebibliography}{9}
\bibitem{aa1} S.A. Bleiler and A.J. Casson,
```

to zacytować ją możemy pisząc `\cite{aa1}`.

Rozdział o nazwie „Dodatek”

W niektórych dokumentach musimy umieścić rozdział, który jest słabo związany z poprzednimi, ale jest niezbędny, gdyż zawiera bardzo ważne informacje dodatkowe. Należy wtedy skorzystać z polecenia `\appendix` lub ze środowiska `appendix`. Jeśli pojawia się jedno z tych poleceń, liczniki części

(section) w klasie `article`, a rozdziałów (chapter) w klasach `book` i `report` zostają wyzerowane i zastąpione dużymi literami alfabetu.

Tworzenie skorowidza

Skorowidz w dokumencie generowany jest programem `MakeIndex`. Aby go utworzyć należy po kolei:

- W preambule dokumentu umieścić następujące dwie linie

```
\usepackage{makeindex}
\makeindex
```

- Każda nazwa mająca znaleźć się w skorowidzu powinna być umieszczona jako argument polecenia `\index{treść}`.

- W miejscu gdzie ma się znaleźć skorowidz (zwykle na końcu dokumentu) umieszczamy instrukcję `\printindex`.

Krok 1 Dokument zapisany jako nazwa.tex musi teraz być **dwukrotnie** (aby utworzyły się wszystkie odnośniki i spisy) skompilowany. W czasie kompilacji LaTeX-em (lub pdfLaTeX-em) generowany jest między innymi plik nazwa.idx.

Krok 2 Uruchamiamy w linii poleceń program `MakeIndex` pisząc

```
makeindex nazwa.idx
```

którym generujemy pliki nazwa.ind i nazwa.ilg. Pierwszy z nich zawiera sformatowany tekst skorowidza, drugi jest odpowiednikiem pliku log.

Krok 3 Ponownie co najmniej raz kompilujemy plik nazwa.tex LaTeX-em (lub pdfLaTeX-em).

Poniżej kilka istotnych informacji o których należy pamiętać.

- Nie wolno zostawiać dodatkowych (niepotrzebnych) odstępów w treści która ma się znaleźć w skorowidzu.
  – Dobrze `\index{Słowo}`
  – Źle `\index{ Słowo}`
  To dla programu są dwa różne słowa.

- Jeżeli raz napiszemy słowo z dużej litery a drugi raz z małej, będą to dla programu `MakeIndex` dwa różne słowa.

Więcej informacji o składaniu skorowidzów czytelnik znajdzie na przykład w [3].

3.3. Składanie dużych dokumentów

Jeśli składany dokument ma więcej niż dwa rozdziały (lub więcej niż 10 stron), wygodnie jest podzielić go na osobne pliki, zawierające na przykład kolejne rozdziały (chapter) lub nawet punkty (section). Taki plik nie może zawierać preambuły ani poleceń `\begin{document}` i `\end{document}`. Cała preambuła wraz z wczytywanymi pakietami i definicjami mającymi obowiązywać w całym dokumencie powinna być zapisana w osobnym pliku głównym. Tam także powinien się znaleźć tytuł, autor, spis literatury, indeks oraz spisy treści, tabel, rysunków itp. Tak utworzony plik główny będziemy kompilować.

Pozostałe części dokumentu dołączamy do pliku głównego za pomocą instrukcji `\include{plik}` lub `\input{plik}`. Kolejność włączania zależy od kolejności rozdziałów. Podane wyżej instrukcje różnią się od siebie sposobem dołączania pliku. `\include{plik}` rozpoczyna skład dołączonego tekstu od nowej strony, zaś `\input{plik}` kontynuuje już rozpoczętą stronę. Obie te instrukcje powinny znajdować się w pliku głównym, choć można ich używać również wewnątrz innych plików.

Aby spośród plików zawierających rozdziały książki o nazwach plik1,

plik2, ... wybrać tylko kilka, na przykład plik2 i plik4, możemy użyć instrukcji `\includeonly{plik1,plik2,\dots}`. Spowoduje ona wybranie tylko dwóch konkretnych plików. Należy jednak pamiętać, że musi ona być umieszczona w preambule dokumentu, czyli w pliku głównym. Nazwy plików należy oddzielać przecinkiem, nie wolno zostawiać odstępów między nazwami.

Argument `plik` obu poleceń `\input` i `\include` może zawierać pełną ścieżkę dostępu do pliku. Należy pamiętać, że bez względu na używany system operacyjny, katalogi oddzielamy znakiem /.

```
\chapter{Jak zacząć}
\input d:/Dane/Początek.tex
```

3.4. Style strony

Styl strony to inaczej jej wygląd. Możemy go zadeklarować umieszczając w preambule polecenie `\pagestyle{nazwa stylu}`. Standardowo dostępne są następujące style:

- plain – nagłówek (head) strony jest pusty, stopka (foot) zawiera wycentrowany numer strony. Ten styl przyjmowany jest jako domyślny, jeżeli żaden inny nie jest zdefiniowany w preambule.
- empty – nagłówek i stopka są puste, nie drukują się także numery stron.
- headings – nagłówek zawiera numer strony oraz nazwę i tytuł rozdziału (punktu, zależnie od klasy dokumentu). Na stronie z tytułem nagłówek jest pusty. Stopka jest zawsze pusta.
- myheadings – numer strony tak samo jak w stylu headings znajduje się w nagłówku, a tekst trzeba podać samodzielnie stosując polecenia `\markright{tekst}` (w dokumencie drukowanym jednostronie)

i `\markboth{lewy tekst}{prawy tekst}` (w dokumencie drukowanym dwustronnie). Stopka, tak jak w stylu headings jest pusta.

Jeżeli któraś strona w dokumencie ma nie mieć wydrukowanego numeru, wtedy gdziekolwiek w tekście na tej stronie umieszczamy polecenie `\thispagestyle{empty}`. Następna strona będzie miała numer kolejny (poprzednia zostanie policzona, lecz numer nie będzie drukowany).

3.5. Numerowanie stron

We wszystkich podstawowych klasach (`article, mwart, report, mwrep, book, mwbk`) strony są numerowane cyframi arabskimi. Polecenie `\pagenumbering{styl numerowania}` pozwala zmienić styl numerowania na jeden z następujących:

- arabic – liczby arabskie (domyślnie),
- roman – liczby rzymskie małe,
- Roman – liczby rzymskie duże,
- alph – małe litery alfabetu łacińskiego,
- Alph – duże litery alfabetu łacińskiego.

Jeśli decydujemy się numerować strony literami alfabetu łacińskiego (małymi lub dużymi) to musimy pamiętać, że jest ich tylko 26 i tylko tyle stron możemy ponumerować.

Numerowaniem steruje licznik o nazwie `page`. W każdym miejscu dokumentu możemy nadać mu inną wartość niż kolejna za pomocą instrukcji `\setcounter{page}{numer}`. Stronie na której pojawi się ta instrukcja, zostanie nadana liczba `numer`, a następne otrzymają numery kolejne. Bardzo rzadko zdarza się jednak, aby po stronie o numerze n następowała strona o numerze $n + 5$, natomiast w niektórych większych dokumentach,

np. w książkach, kilka pierwszych stron bywa numerowanych małymi liczbami rzymskimi. Wówczas właściwa treść książki powinna rozpocząć się od pierwszej strony, i tu znajduje zastosowanie opisana wyżej możliwość zmiany wartości licznika stron.

Nie zawsze musimy numerować strony od pierwszej. Aby rozpocząć numerowanie stron naszego dokumentu od strony drugiej wystarczy w preambule napisać: `\setcounter{page}{2}`.

3.6. Pakiet `fancyhdr`

Strona dokumentu LaTeX-owego jest złożona z tekstu zasadniczego oraz wielu elementów takich jak marginesy, stopki czy nagłówki (patrz rysunek 4.6.). Strona w LaTeX-u powstaje w inny sposób niż strony tworzone w popularnych edytorach dlatego zmodyfikowanie jej układu jest dość skomplikowane. Przystępując do zmiany domyślnych ustawień należy pamiętać, że

1. Lewy margines na stronie parzystej nazywa się `\evensidemargin` (a nie `\leftmargin`) a na stronie nieparzystej `\oddsidemargin`. W dokumentach drukowanych jednostronnie określa się tylko margines `\oddsidemargin`, dla stron nieparzystych i parzystych.
2. Większość parametrów nie może być zmieniana wewnątrz dokumentu. Niektóre zmiany są wykonywane dopiero po zakończeniu strony. Jeśli chcielibyśmy zmienić wysokość `\height` tylko jednej strony musimy użyć instrukcji `\enlargethispage`.

Makra dostarczone wraz z pakietem `fancyhdr` pozwalają na łatwe przedefiniowanie nagłówków i stopek w naszym dokumencie. Możemy

- zdefiniować trzyczęściowe nagłówki i stopki,
- wstawiać różnego rodzaju linie oddzielające nagłówki i stopki od tekstu głównego,

- nagłówki i stopki szersze niż tekst główny,
- wieloliniowe nagłówki i stopki,
- inne nagłówki i stopki na stronach parzystych i nieparzystych,
- inne nagłówki i stopki na stronie z tytułem rozdziału i na stronach zawierających rysunki lub tabele,
- kontrolować rodzaj i rozmiar fontu w nagłówkach i stopkach.

Rysunek poniżej pokazuje wszystkie elementy strony które możemy modyfikować.

Jak to zrobić?

Najpierw należy poleceniem `\usepackage{}` wczytać pakiet **`fancyhdr`**, następnie umieścić deklarację stylu stron poleceniem `\pagestyle{fancy}`. Po wczytaniu pakietu będziemy mogli definiować poszczególne elementy nagłówków i stopek korzystając z następujących instrukcji:

- `\lhead` — lewy nagłówek,
- `\chead` — środkowy nagłówek,
- `\rhead` — prawy nagłówek,
- `\lfoot` — lewa stopka,
- `\cfoot` — środkowa stopka,
- `\rfoot` — prawa stopka.

Te instrukcje działają w dokumencie składanym jednostronnie. W składzie dwustronnym poszczególne części nagłówka (stopki) na stronach parzystych i nieparzystych różnią się, należy więc rozróżnić je przy projektowaniu nagłówków i stopek.

Do projektowania nagłówków i stopek używamy wtedy poleceń

```
\fancyhead[miejsce]{treść}
\fancyfoot[miejsce]{treść}
```

gdzie parametr `miejsce` może przyjmować następujące wartości:

E	Strona parzysta (even)
O	Strona nieparzysta (odd)
L	Lewa strona
C	Środkowa strona
R	Prawa strona
H	Nagłówek
F	Stopka

Poniżej fragment kodu przykładu 3.1 pokazujący jak został zdefiniowany nagłówek i stopka:

```
\fancyhead{} % wszystkie nagłówki puste
\fancyhead[RO,LE]{ Absolwenci WMiI 2012}
\fancyfoot{} % wszystkie stopki puste
\fancyfoot[LE,RO]{\thepage}
\fancyfoot[LO,CE]{Od: Dziekan}
\fancyfoot[CO,RE]{Do: Rektor UŁ}
\renewcommand{\headrulewidth}{0.4pt}
\renewcommand{\footrulewidth}{0.4pt}
```

¶ Przykład 3.1.
Strony nieparzyste

Absolwenci WMiI 2012

Lista

Od: Dziekan Do: Rektor UŁ 3

□

¶ Przykład 3.2.
Strony parzyste

Absolwenci WMiI 2012

Lista

4 Od: Dziekan Do: Rektor UŁ

□

Za linie oddzielające tekst nagłówka od treści dokumentu odpowiada polecenie `\headrule` i `\footrule`. Domyślnie są to linie ciągłe o ustalonej grubości, ale możemy je przedefiniować tak, aby otrzymać np. linię złożoną z kropek. Polecenie `\headrule` powinno wyglądać następująco:

```
\renewcommand{%
\headrule}{\vbox to 0pt{\hbox to \headwidth{\dotfill}\vss}}
```

Poleceniem `\renewcommand` możemy też zmienić szerokość tych linii

```
\renewcommand{\headrulewidth}{0.4pt}
\renewcommand{\footrulewidth}{0.6pt}
```

Pakiet `fancyhdr` pozwala też na zmianę szerokości nagłówka i stopki. Domyślnie oba te elementy mają szerokość tekstu. Możemy je zwęzić lub poszerzyć używając poleceń:

```
\fancyheadoffset[miejsce]{długość}
\fancyfootoffset[miejsce]{długość}
\fancyhfoffset[miejsce]{długość}
```

Są one podobne do poleceń `\fancyfoot` i `\fancyhf` z tą jednak różnicą, że nie akceptują opcji C. Pozostałe akceptowalne miejsca to E, O, L i R. Jeśli parametr „długość" będzie dodatni, to otrzymamy nagłówek (stopkę) szerszy, jeśli ujemny to węższy.

Jeżeli wczytamy w preambule dokumentu pakiet `lastpage` będziemy mogli zdefiniować numery stron postaci „n z m". Należy wtedy przedefiniować część stopki (nagłówka) w której ten numer ma się znajdować. Poniżej przykład dla numeru w stopce, w środkowej jej części

```
\cfoot{\thepage\ z \pageref{LastPage}}
```

Tak zdefiniowana numeracja stron pojawia się w nagłówkach lub w stopkach ale nie w odsyłaczach, w spisach treści i indeksach.

W dokumentacjach technicznych pojawia się numerowanie stron w obrębie rozdziału czyli postaci „2-10", gdzie pierwszy numer oznacza numer rozdziału a drugi jest numerem strony tego rozdziału. Aby uzyskać takie numerowanie w dokumencie należy użyć pakietu `chappg`. Pakiet ten resetuje również licznik stron po zakończeniu rozdziału i w nowym rozdziale rozpoczyna numerowanie stron od pierwszej.

Aby styl numerowania „2-10" pojawił się również w spisie treści, odsyłaczach czy w indeksie, musimy przedefiniować odpowiednio `\thepage`.

Opis wielu innych poleceń pakietu `fancyhdr` można znaleźć w dokumentacji dostarczanej wraz z dystrybucją TEXLive lub w katalogach The Comprehensive TEXArchive Network (ctan.org).

3.7. Klasy extsizes

Standardowe klasy LaTeX-a umożliwiają składanie tekstu czcionką 10, 11 i 12 punktową, które to wielkości są najczęściej używane w publikacjach. Może się jednak zdarzyć, że do złożenia naszej publikacji będziemy potrzebować np. czcionki 14-to punktowej. Wówczas możemy skorzystać z którejś z klas extsizes: extarticle, extreport, extbook, extletter czy extproc. Klasy te definiują nowe rozmiary czcionek: 8 pt, 9 pt, 10 pt, 11 pt, 12 pt, 14 pt, 17 pt i 20 pt, lecz nie należy ich używać do składania dokumentu, gdy wystarczą nam standardowe rozmiary.

Klasy extsizes zostały zaadaptowane z klas LaTeX-owych przez Wolfganga Maya, a zmodyfikował je James Kilfiger.

Aby użyć jednej z klas extsizes z konkretną wielkością czcionki należy dokument rozpocząć

```
\documentclass[14pt]{extreport}
```

lub

```
\documentclass[9pt]{extarticle}
```

Wszystkie inne parametry dokumentu są ustawiane automatycznie. Przy składaniu dokumentu dużymi rozmiarami czcionek (18 lub 20 punktów) tekst może być źle łamany na wiersze. Wówczas można w preambule dokumentu umieścić instrukcję \sloppy, choć lepiej jest bezpośrednio w dokumencie podpowiedzieć LaTeX-owi, gdzie może podzielić wyraz umieszczając w tym miejscu instrukcję \-.

3.8. Pakiet geometry

Pakiet geometry zapewnia elastyczny i łatwy w użyciu interfejs umożliwiający zmianę wymiarów strony dokumentu dostarczając użytkownikowi zestaw intuicyjnych parametrów. Wczytuje się go jak zwykle poleceniem

`\usepackage{}`. Pakiet posiada bogaty zbiór dodatkowych parametrów, które omówimy w dalszej części tego dokumentu. Niektóre z nich są wczytywane wraz z pakietem z przypisanymi im wartościami domyślnymi. Jeżeli chcemy zmienić domyślne wartości parametrów odpowiadających za np. szerokość marginesów, możemy je umieścić jako parametry polecenia `\usepackage` lub wymienić je jako argument polecenia `\geometry{}`. Tabela 3.2 pokazuje jakie wielkości opisują stronę.

```
\usepackage[layout=a5paper,landscape]{geometry}
\geometry{showframe, hoffset=2cm}
```

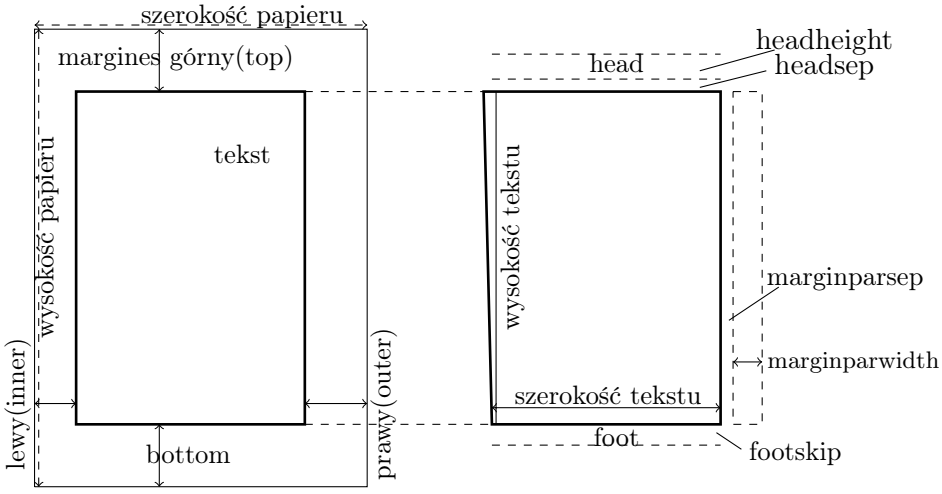
Parametry strony w pakiecie geometry

Strona składa się z części zadrukowanej (total body) i marginesów. Na część zadrukowaną składają się: obszar tekstu (text area) z opcjonalnymi stopką, nagłówkiem i notkami na marginesie (marginpar). Marginesy są cztery: lewy, prawy, górny i dolny. Dla dokumentów dwustronnych poziome marginesy powinny nazywać się wewnętrzny i zewnętrzny. W tabel 3.1 wyjaśniamy znaczenie niektórych określeń używanych w tym dokumencie.

rozmiar papieru (paper)	–	część zadrukowana (total body) plus marginesy
część zadrukowana (total body)	–	obszar tekstu (body) i opcjonalne stopka (foot), nagłówek (head) i marginpar.
marginesy	–	górny, dolny, lewy (wewnętrzny) i prawy (zewnętrzny).

Tabela 3.1: Geometria strony

Jeśli użyjemy opcji `includehead` i `includefoot`, wówczas treść dokumentu zostanie rozmieszczona w prostokącie o bokach „wysokość tekstu" i „szerokość tekstu".



Rysunek 3.2: Organizacja strony

Strona tekstu składa się z trzech części (długości) w każdą stronę: jedno 'total body' i dwa marginesy. Wystarczy zdefiniować dwie wielkości, trzecia zostanie wyliczona przez mechanizm auto-uzupełniania.

Polecenia pakietu `geometry`

W pakiecie zdefiniowane są następujące polecenia:

- `\geometry{parametry}`
- `\newgeometry{parametry}` i `\restoregeometry`
- `\savegeometry{nazwa}` i `\loadgeometry{nazwa}`

Zadeklarowanie w preambule `\geometry{parametry}` spowoduje zmianę układu strony zgodnie z wartościami parametrów przywołanych w argumencie `parametry`. Jeśli polecenie `\geometry{parametry}` wpiszemy w preambule kilkakrotnie z różnymi parametrami, to wszystkie te parametry zostaną zastosowane łącznie. Można w ten sposób zmieniać układ strony dokumentu w trakcie jego powstawania.

Deklaracja `\newgeometry{<parametry>}` pozwala na zmianę parametrów strony wewnątrz dokumentu, czyli pozwala wewnątrz dokumentu wstawić na przykład krótszą i węższą stronę niż pozostałe. Aby powrócić do ustawień sprzed zastosowania `\newgeometry` wystarczy napisać `\restoregeometry`.

Jeśli tworzony dokument ma zawierać strony o różnych szerokościach, wygodnie jest użyć poleceń `\savegeometry{}` i `\loadgeometry{}`. Pierwsze z nich powoduje zapamiętanie nowych parametrów strony a drugie stosuje je tam, gdzie się pojawi.

Listę parametrów poleceń `\geometry{}`, `\newgeometry{}` i opcji `\usepackage{geometry}` wypisujemy w postaci „nazwa=wartość" oddzielonych przecinkami. Lista ta musi czynić zadość następującym warunkom:

- parametry można wypisać w kilku liniach, o ile nie oddzielimy ich linią pustą,
- dodatkowe odstępy pomiędzy kolejnymi wartościami są ignorowane,
- działanie poszczególnych parametrów nie zależy od miejsca położenia na liście.

Przykłady poniżej pokazują kilka równoważnych list parametrów.

¶ Przykład 3.3.

```
\usepackage[a5paper, hmargin = {3 cm,.6 in},
height = 10in]{geometry}
```

□

¶ Przykład 3.4.

```
\usepackage[a5paper,height=10in,hmargin={3cm,0.6in}]%
{geometry}.
```

□

¶ Przykład 3.5.

```
\usepackage{geometry}
\geometry{a5paper,height=10in,hmargin={3cm,0.6in}}
```

oraz

```
\usepackage[a5paper]{geometry}
\geometry{height=10in,hmargin={3cm,0.6in}}
```

□

Typy parametrów

W pakiecie geometry dostępne są cztery typy parametrów.

1. Przyjmujące wartości 'prawda (true)', 'fałsz (false)' (boolean type) np. `twoside=true`. Jeśli nie ma podanej wartości, domyślną jest 'prawda'. Nazwa (rozmiar) papieru jest wyjątkiem i powinien być podawany bez żadnej wartości.

2. Przyjmujące tylko jedną wartość np. `left=1.5cm,footskip=1cm`.

3. Przyjmujace dwie wartości <nazwa>={<wartość1>,<wartość2>} np. `hmargin={1cm,2cm}`

4. Przyjmujące trzy wartości <nazwa>={<wart1>,<wart2>,<wart3>}. W każdym przypadku <wartość> musi oznaczać jakiś wymiar lub zero. Jako zero można zostawić puste miejsce lub wpisać znak '∗' np.: `hdivide={2cm,*,2cm},␣vdivide={1cm,14cm,␣}`.

Rozmiar tekstu (body size)

W następnej tabeli opisujemy parametry najczęściej używane do określenia wymiarów tekstu na stronie.

`vscale` – stosunek wysokości części zadrukowanej do długości papieru.

`scale` – stosunek szerokości części zadrukowanej (total body) do papieru.

`hscale` – stosunek szerokości części zadrukowanej (total body) do szerokości papieru `\paperwidth`. Ustalenie wartości `hscale=0.9` jest równoważne deklaracji `width=0.9\paperwidth`. Domyślna wartość tego parametru to 0.7.

`width` – szerokość części zadrukowanej (total body).

`height` – wysokość części zadrukowanej (total body).

`total` – szerokość i wysokość części zadrukowanej (total body).

`textwidth` – ustala szerokość tekstu.

`textheight` – ustala wysokość tekstu.

`text|body` – ustala szerokość i wysokość tekstu. `body={<szerokość>, <wysokość>}` lub `text=<wartość>`.

`lines` – pozwala użytkownikowi ustalić wysokość tekstu (`\textheight`) poprzez podanie ilości linii tekstu na stronie `lines=<liczba>`.

`includehead` – wysokość nagłówka wraz z odstępem od tekstu wliczana jest do wysokości części zadrukowanej (total body)

Marginesy

Dla przejrzystości dokumentu istotne są również wielkości marginesów górnego i dolnego oraz lewego i prawego. Parametry wymienione w następnej tabeli pozwalają kontrolować ich rozmiary.

`left|lmargin|inner` – margines lewy (druk jednostronny) lub wewnętrzny (druk dwustronny). Jest to odległość od lewej krawędzi papieru do lewego brzegu total body (określenia wyjaśnione w tabeli 3.1).

`right|rmargin|outer`	–	prawy lub zewnętrzny (druk dwustronny) margines, `\right=<szerokość>`.
`top|tmargin`	–	margines górny `\top=<szerokość>`. Ta opcja nie ma nic wspólnego z poleceniem `\topmargin`.
`bottom|bmargin`	–	margines dolny na stronie.
`hmargin`	–	lewy i prawy margines ustalany jednocześnie poleceniami `hmargin={<lewy>,<prawy>}` lub `hmargin=<wielkość>`.
`vmargin`	–	górny i dolny margines ustalany jednocześnie poleceniami `vmargin={<góra>,<dół>}` lub `vmargin=<wielkość>`.
`hmarginratio`	–	stosunek wielkości lewego i prawego marginesu. Stosunek ten powinien być zapisany liczbami całkowitymi <100, oddzielonymi dwukropkiem np. 2:3. Domyślnie jest 1:1 dla druku jednostronnego i 2:3 dla druku dwustronnego.
`vmarginratio`	–	stosunek wielkości marginesów dolnego i górnego. Domyślnie 2:3.
`centering`	–	automatycznie centruje tekst na stronie i jest równoważny z `marginratio=1:1`. Domyślna wartość jest 'false'.
`bindingoffset`	–	rezerwuje na oprawę część obszaru z lewej strony w druku jednostronnym, a od strony wewnętrznej w druku dwustronnym. `bindingoffset=<wielkość>`.

Wszystkie te opcje zmieniają bądź modyfikują wartości domyślne parametrów strony wczytywane razem z klasą dokumentu.

Inne parametry opcjonalne

`mag` – ustala wartość powiększenia (`\mag`) i automatycznie modyfikuje wielkości `\hoffset` i `\voffset`. Podaje się go w postaci `mag=<liczba>`. Wartość 'liczba' powinna być liczbą całkowitą. Liczba 1000 oznacza rozmiar normalny (dla danego papieru). Dla `a4paper` `mag=1414` będzie oznaczało powiększenie do rozmiaru `a3paper`, który jest $1.414(=\sqrt{2})$ razy większy niż `a4paper`. W tym przypadku powiększone zostaną także wszystkie wielkości podane jako 'true', na przykład `1.5truein`.

`truedimen` – zmienia wszystkie wewnętrzne wielkości na takie, które nie będą podlegały zwiększeniu gdy powiększymy dokument poleceniem `\mag`. Zazwyczaj ten parametr jest używany z poleceniem `\mag`.

`showframe` – rysuje ramki wokół wszystkich zdefiniowanych obszarów strony takich jak tekst, nagłówek, stopka itd.

`showcrop` – drukuje znaki we wszystkich rogach zdefiniowanego przez użytkownika układu strony.

Opis pozostałych parametrów dostępnych w pakiecie `geometry` czytelnik znajdzie w dokumentacji pakietu.

Ustawienia domyślne

Domyślne ustawienia parametrów strony są następujące:

1. Marginesy górny i dolny są w stosunku 2/3.
2. Marginesy poziome zależą składu dokumentu.

$$\text{lewy(wewnętrzny)} : \text{prawy(zewnętrzny)} = \begin{cases} 1:1 & \text{jednostronnie} \\ 2:3 & \text{dwustronnie} \end{cases}$$

Dla dokumentu jednostronnego pakiet `geometry` ma następujące domyślne parametry strony:

1. `scale`=0.7 ('body' to 0.7 × paper),
2. `marginratio`= $\{1:1, 2:3\}$ (1:1 – marginesy poziome, 2:3 – marginesy pionowe),
3. `ignoreall` (nagłówek, stopka i notki na marginesie nie są brane pod uwagę, gdy wyliczane są wymiary 'body'.

Dla dokumentu dwustronnego z parametrem `twoside`, domyślne ustawienia są takie same jak dla jednostronnego z wyjątkiem marginesów poziomych, które są ustalone w stosunku 2:3.

4. Składanie dokumentu

4.1. Formatowanie tekstu

Dokument TEX-owy (i LATEX-owy) jest przygotowywany i zapisywany w czystym kodzie ASCII, a jakiekolwiek formatowanie tekstu, np. wszelką zmianę grubości, wielkości lub kształtu liter w tekście musimy specjalnie zaznaczyć.

Możemy przyjąć, że font (czcionka) ma trzy wyróżniające go parametry:

- Krój (ang. typeface) - odpowiada za wygląd czcionki,
- Odmiana - określa grubość linii w literach oraz (lub) stopień pochylenia liter,
- Stopień - określa wielkość liter.

Kroje pism łączone są w rodziny. Kroje należące do danej rodziny muszą posiadać pewne cechy wspólne, mogą natomiast różnić się pochyleniem, grubością czy odstępem pomiędzy znakami. Rodzina fontów zatem to zbiór wszystkich odmian danego kroju. Odmiany to na przykład: `light, regular, italic, medium italic, medium bold, bold italic` czy `light italic`. Nie każda rodzina fontów składa się z wszystkich wymienionych odmian.

Podstawowa odmiana kroju określana jest najczęściej jako regular lub normal (też antykwa), jego cieńsza wersja jako `light`, a grubsza jako `bold`.

W prawie każdym kroju istnieje odmiana pochyła zwana kursywą (ang. *italic*), wzorowana na kursywie renesansowej. Projektant kroju może też zaprojektować inną odmianę pochyłą (ang. *slanted*), polegającą na pochyleniu

kresek znaków prostych. W foncie może też znaleźć się odmiana szeroka, wąska czy też imitująca `pismo maszynowe` czyli z równymi odstępami pomiędzy literami.

Domyślnie LaTeX składa dokument czcionką Computer Modern. Aby do całego dokumentu zastosować inny krój pisma, należy wczytać go w preambule polecniem `\usepackage{nazwa kroju}`. Niektóre dostępne kroje to:

- lmodern – rodzina Latin Modern,
- iwona – rodzina czcionek bezszeryfowych,
- antpolt – rodzina czcionek Antykwa Półtawskiego – czcionki szeryfowe,
- anttor – rodzina czcionek Antykwa Toruńska – czcionki szeryfowe,

Nie wszystkie wymienione kroje mają fonty matematyczne. W tej chwili na pewno Latin Modern zawiera takie czcionki. Więcej informacji o rodzajach fontów czytelnik znajdzie na www.gust.org.pl

Wyróżnienia w tekście

Zmiana odmiany pisma w tekstowym trybie pracy LaTeX-a (czyli w trybie akapitowym) możliwa jest dzięki instrukcjom przedstawionym w tabeli poniżej.

`\textit{}`	— tekst napisany kursywą
`\textbf{}`	— tekst napisany czcionką pogrubioną
`\texttt{}`	— tekst napisany czcionką imitującą pismo maszynowe
`\textsl{}`	— tekst napisany czcionką pochyłą.

Tabela 4.1: Instrukcje zmieniające odmiany czcionek

Należy pamiętać, by cały wyróżniony tekst był umieszczony w nawiasach klamrowych ograniczających działanie instrukcji.

```
\textit{Ten tekst napisany jest kursywą}
\textbf{Ten tekst napisany jest czcionką pogrubioną}
\textsl{Ten tekst napisany jest czcionką pochyłą}
\texttt{Ten tekst napisany jest czcionką imitującą
pismo maszynowe}
```

¶ Przykład 4.1.

Ten tekst napisany jest kursywą

Ten tekst napisany jest czcionką pogrubioną

Ten tekst napisany jest czcionką pochyłą

`Ten tekst napisany jest czcionką imitującą pismo maszynowe`.

□

W formacie `plain` odmianę pisma zmienia się instrukcjami `\it`, `\bf`, `\tt` i `\sl`, ale działają one nieco inaczej niż te zdefiniowane w LaTeX-u. Jeżeli do jakiegoś wyrazu zastosujemy instrukcję `\bf` i nie zamkniemy jej wraz z wyrazem w nawiasach klamrowych, to reszta naszego dokumentu będzie złożona czcionką pogrubioną.

```
{\bf Ten tekst ma być częściowo} złożony odmianą pogrubioną.
```

Ten tekst ma być częściowo złożony odmianą pogrubioną.

Jeżeli teraz instrukcji `\bf` użyjemy tak samo jak `\textbf` otrzymamy

Ten tekst ma być częściowo złożony odmianą pogrubioną.

Aby wyróżnić tekst przez pochylenie możemy też użyć jednej z następujących deklaracji `\em` lub równoważnej `\emph`. Obie te deklaracje różni tylko sposób, w jaki powinny być wywoływane. Pierwsza działa dopóki nie zostanie użyta inna deklaracja zmiany fontu, więc cały tekst razem z nią należy zamknąć w grupę za pomocą nawiasów klamrowych. W drugiej deklaracji wyróżniany tekst jest jej argumentem, więc w nawiasach klamrowych umieszczamy tylko sam tekst. To, jaką odmianą fontu jest składany tekst gdy używamy opisanych deklaracji, zależy od użytej klasy dokumentu. W klasie

`article` (i `mwart`) domyślnie stosowana jest kursywa. W przykładzie poniżej można zobaczyć efekt użycia obu tych deklaracji w klasie `article`.

¶ Przykład 4.2.

Pierwsza deklaracja *pochylająca litery tekstu.*

```
Pierwsza deklaracja {\em pochylającej litery tekstu}
```

Druga deklaracja *pochylającej litery tekstu.*

```
Druga  deklaracja \emph{ pochylającej litery tekstu}.
```

□

Jeden lub kilku wyrazów możemy także wyróżnić poprzez podkreślenie. Najprościej w takim przypadku zastosować polecenie `\underline{wyraz}`.

¶ Przykład 4.3.

Jeden lub kilka wyrazów możemy wyróżnić poprzez podkreślenie. Wyróżnienie całego akapitu przez podkreślenie nie jest uważane za poprawne. Należy pamiętać, że w tekście wyróżnionym podkreśleniem LaTeX nie przenosi wyrazów (traktuje go jako niełamliwe pudełko), mogą więc powstawać wiersze nadmiarowe lub zwiększone odstępy międzywyrazowe tak jak w tym przykładzie.

□

Zmiana rozmiaru czcionki

Standardową czcionką dla dokumentu LaTeX-owego jest 10-cio punktowa prosta czcionka rodziny Computer Modern. Jeśli w preambule dokumentu wczytamy inną rodzinę (patrz rozdział 2.6.), wówczas nasz dokument zostanie złożony czcionkami tej rodziny. W preambule dokumentu, jako jedną z opcji klasy możemy również ustalić wielkość czcionki. Domyślnie dostępne są wielkości 10, 11 i 12 punktów (1 cal = 72,27 pt, 1 cm = ok. 28,45 pt).

W LaTeX-u znajdziemy też deklaracje różnych wielkości czcionek, od bardzo małej (`\tiny`) do bardzo dużej (`\Huge`). Przykładowe wielkości umieszczone są w tabeli 4.2. Używając tych deklaracji w dokumencie należy pamiętać o grupowaniu powiększanego tekstu w nawiasach klamrowych.

¶ Przykład 4.4.

Na początek zastosujemy powiększenie w sposób właściwy, napisaliśmy `{\huge powiększenie}`. A teraz inne powiększenie i brak grupowania, zobaczymy co się stanie.

□

W drugim przypadku powiększony został cały tekst a nie tylko jeden wyraz następujący po instrukcji zwiększającej czcionkę. Aby zakończyć działanie instrukcji powiększającej i powrócić do wielkości domyślnej dla naszego dokumentu można też wywołać instrukcję `\normalsize`.

Stopień pisma tekstu głównego	10pt	11pt	12pt	
`\tiny`	5pt	6pt	6pt	najmniejsza
`\scriptsize`	7pt	8pt	8pt	bardzo mała
`\footnotesize`	8pt	9pt	10pt	nieco większa
`\small`	9pt	10pt	11pt	mała
`\normalsize`	10pt	11pt	12pt	rozmiar normalny
`\large`	12pt	12pt	14pt	duża
`\Large`	14pt	14pt	17pt	większa
`\LARGE`	17pt	17pt	20pt	jeszcze większa
`\huge`	20pt	20pt	25pt	jeszcze większa
`\Huge`	25pt	25pt	25pt	największa

Tabela 4.2: Predefiniowane wielkości czcionek

Do składania większych partii tekstu czcionką większą od podstawowej, wygodnie jest skorzystać z pary instrukcji `\fontsize{}{}` i `\selectfont`.

Użyjemy ich w następnym przykładzie.

¶ Przykład 4.5.

Tekst tej książki jest składany czcionką 11-punktową. Te wyrazy są powiększone do 14 punktów, te dwa do 18 punktów a ten jest napisany czcionką 24-punktową.

□

```
11-punktową. {\fontsize{14}{14}\selectfont Te wyrazy}
 do 14~punktów, {\fontsize{18}{18}\selectfont te dwa}
a~{\fontsize{24}{24}\selectfont ten} jest napisany
czcionką 24-punktową.
```

Argument umieszczony w pierwszej klamrze instrukcji `\fontsize` to rozmiar czcionki (w punktach), a argument w drugiej parze kontroluje odległość pomiędzy wierszami (jeśli taka potrzeba zaistnieje).

Instrukcja `\selectfont` powoduje rozpoczęcie składania tekstu wybranym fontem. Cały powiększany tekst, razem z omówionymi instrukcjami, należy zgrupować w nawiasach klamrowych. Takie zgrupowanie zagwarantuje nam zastosowanie instrukcji tylko do konkretnego tekstu. Aby zakończyć działanie omawianych instrukcji można, zamiast grupować je wraz z tekstem, umieścić na końcu instrukcję `\normalsize`, która informuje kompilator o powrocie do wielkości domyślnej fontu.

Opisanych wyżej instrukcji nie stosujemy do powiększania czcionek w całym dokumencie. Wygodne do tego celu jest użycie klas `extsizes` (patrz rozdział 3.7.) lub zastosowanie powiększenia dostępnego po wczytaniu pakietu `geometry` (patrz rozdział 3.8.).

Odstępy poziome i pionowe

TEX składa wiersze i akapity inaczej niż standardowe edytory, a więc rozpoczynając dokument należy pamiętać, że:

1. Bez względu na to ile umieścimy odstępów (spacji) pomiędzy dwoma wyrazami, w dokumencie pojawi się tylko jeden, czyli jeden odstęp poziomy znaczy tyle samo co wiele, tylko pierwszy jest brany pod uwagę.
2. Odstępy na początku linii są ignorowane.
3. Odstęp kończący polecenie lub instrukcję jest usuwany.
4. Przejście do nowej linii (CR) jest traktowane jak pojedynczy odstęp.
5. Kończymy akapit pozostawiając pustą linię, lub umieszczając na końcu akapitu instrukcję `\par`.
6. Dwie, trzy lub więcej wolnych linii w tekście znaczy tyle samo co jedna.

Widzimy, że powyższe własności TEX-a (LATEX-a) wykluczają możliwość uzyskania w dokumencie odstępu innego niż domyślny międzywyrazowy czy międzywierszowy. Dlatego też zostały zdefiniowane osobne polecenia umożliwiające wstawienie dowolnego odstępu poziomego lub pionowego.
Odstęp poziomy dowolnej długości uzyskamy wstawiając pomiędzy dwa wyrazy polecenie `\hspace{odległość}` lub `\hspace*{odległość}`. Odległość musi być podana razem z jednostką np. `\hspace{2cm}`. Polecenie bez gwiazdki wstawione na początku wiersza jest ignorowane, natomiast z gwiazdką jest wykonywane.

¶ Przykład 4.6.

To jest przykład odstępu poziomego o długości 1,5 cm.

```
To jest przykład odstępu poziomego\hspace{1.5cm}
 o długości 1,5 cm.
```

□

Dodając odstępy o ściśle ustalonej wielkości należy mieć świadomość, że nie ulegną one zmianie przy zmianie wielkości czcionki np. na mniejszą.
Poziomy odstęp, który będzie zmieniał się razem ze zmianą wielkości czcionki możemy uzyskać za pomocą instrukcji `\quad` i `\qquad`, lub przez umieszczenie pomiędzy wyrazami znaku ukośnika wstecznego (w-tył-ciach lub ang.

backslash) i pustej spacji, czyli \␣. Pierwsze z nich dodaje do pojedynczego odstęp równy szerokości litery m w bieżącym foncie, drugie dwa razy więcej a trzecie standardowy odstęp międzywyrazowy w bieżącym foncie.

Aby uzyskać odstęp poziomy dowolnej wielkości, zmieniający się wraz ze zmianą wielkości czcionki należy zamiast „nierozciągliwych" wielkości takich jak centymetr czy cal używać jednostki `em`. Wtedy na przykład instrukcja `\hspace{3em}` wykona poziomy odstęp o szerokości 3 liter m w bieżącym rozmiarze. Gdy zwiększymy (zmniejszymy) rozmiar czcionki, nasz odstęp też się zwiększy (zmniejszy).

¶ Przykład 4.7.

a b - normalny odstęp między wyrazami
a b - normalny odstęp między wyrazami i dodatkowy \␣
a b - odstęp większy `a\quad b`
a b - dodatkowy odstęp (mniejszy niż poprzedni) a␣\␣b
a b - podwójny dodatkowy odstęp `a\qquad b`

□

Dodatkowe, bardzo małe odstępy można też uzyskać umieszczając bezpośrednio po znaku sterującym dwukropek, przecinek, średnik lub wykrzyknik `\,`, `\;`, `\:`, `\!`. Wszystkie one działają w środowisku matematycznym i są bardzo pomocne przy pisaniu matematycznych wzorów (`\,` można używać również w trybie tekstowym). Poniższa tabelka podaje wielkości tych odstępów.

`\,` mały odstęp = 3/18 odstępu `\quad`,
`\:` średni odstęp = 4/18 odstępu `\quad`,
`\;` duży odstęp = 5/18 odstępu `\quad`,
`\!` odstęp ujemny = −3/18 odstępu `\quad`.

Polecenie `\hfill` będące skróceniem polecenia `\hspace{fill}`, powoduje wypełnienie spacjami przestrzeni między wyrazami i „rozepchnięcie"

ich do lewego i prawego marginesu, jak w następującym przykładzie.

¶ Przykład 4.8.

```
\noindent Strona lewa \hfill strona prawa
```

Strona lewa strona prawa

□

Pustą przestrzeń między wyrazami można, jak w następnym przykładzie, wypełnić kropkami lub linią ciągłą stosując instrukcje `\dotfill` lub `\hrulefill`.

¶ Przykład 4.9.

abecadło ..... z pieca spadło
o ziemię _____ się hukło
rozsypało się ..... po _____ kątach
strasznie _____ się ..... potłukło.

□

W tej samej linii możemy umieścić dowolną kombinację poleceń `\hfill`, `\dotfill`, `\hrulefill`. Jeżeli któraś z nich wystąpi więcej niż jeden raz w tym samym miejscu, to odpowiednie wypełnienie będzie wykonane tylko raz.

¶ Przykład 4.10.

```
Harcerze \hfill\hrulefill\hfill  wesoło śpiewli.
```

Harcerze _____ wesoło śpiewli.

□

Powyżej odległość między wyrazami *Harcerze* i *wesoło śpiewali* została podzielona na trzy części i środkowa została wypełniona kreską.

Odstęp pionowy umieszczony pomiędzy akapitami jest wykonywany tylko wtedy, gdy poprzedzający go akapit został zakończony pustą linią lub instrukcją `\par`. W przeciwnym wypadku polecenie odstępu zostanie wykonane dopiero po zakończeniu kolejnego akapitu lub bieżącego wiersza.

Odległość pomiędzy akapitami ustalana jest automatycznie wraz z wyborem wielkości fontu. Jest to wielkość „rozciągliwa" wyrażana w jednostkach `ex` (1`ex` jest równy wysokości małej litery x w bieżącym foncie). Jest ona ustalana poleceniem `\parskip` i możemy ją zmienić poleceniem `\setlength{\parskip}{odległość}`, gdzie jako `odległość` należy podać również w jakich granicach może być ona rozciągnięta, np. warunek

```
\setlength{\parskip 1.5ex plus 0.5ex minus 0.5ex}
```

dopuszcza zwiększenie lub zmniejszenie odległości między akapitami o 0.5ex.

Jeśli w pewnym miejscu dokumentu umieścimy polecenie zmieniające odstęp pomiędzy akapitami, to będzie ono obowiązywało do końca dokumentu. Aby zmiana dotyczyła całego dokumentu należy je umieścić w preambule, jeśli tylko wybranych akapitów, należy je umieścić w nawiasach klamrowych (zgrupować) razem z poleceniem zmieniającym odstęp.

Powiększenie odstępu między konkretnymi dwoma akapitami uzyskamy instrukcją `\vspace{odległość}` (lub `\vspace*{odległość}`). Parametr `odległość` musi być liczbą (dodatnią lub ujemną) z mianem (np. 1cm, 3ex, itp). Ujemny parametr wstawia ujemny odstęp (czyli zmniejsza odstęp istniejący). Instrukcja z gwiazdką wymusza wstawienie odstępu tam, gdzie normalnie LaTeX nie dopuszcza takiej możliwości, a więc np. na początku strony lub akapitu.

Odległość pomiędzy wierszami dokumentu jest ustalana automatycznie wraz z deklaracją wielkości fontu. Dla każdego rozmiaru są określone wielkości `\baselineskip` oraz `\baselinestretch`, a prawdziwy odstęp pomiędzy wierszami jest iloczynem `\baselineskip` × `\baselinestretch`, który zapewnia właściwe odstępy dla wszystkich rozmiarów fontów.

Aby zmienić wielkość interlinii w całym dokumencie, możemy skorzystać z instrukcji `\linespread{liczba}`, gdzie parametr `liczba` może przyjmować wartości > 1 (1 jest wartością domyślną). Ta instrukcja nie zmieni wielkości interlinii w pojedynczym akapicie wewnątrz dokumentu.

Można także instrukcję \baselinestretch, która domyślnie ma wartość 1, przedefiniować następująco:

```
\renewcommand{\baselinestretch}{wartość}
```

gdzie wartość to ułamek dziesiętny. Wpisanie wartości 1.5 oznacza powiększenie odstępu pomiędzy wierszami o 50% dla wszystkich rozmiarów fontów. Jeśli umieścimy powyższą instrukcję w preambule to zmienimy odstępy pomiędzy wierszami w całym dokumencie.

Zmiana odstępów pomiędzy wierszami (wielkości interlinii) tylko w wybranym akapicie wymaga bardziej skomplikowanych zabiegów. Instrukcja \baselinestretch nie zadziała, aż nie zostanie dwukrotnie zmieniona wielkość czcionki. Aby więc otrzymać akapit z odstępami jak w przykładzie 4.11, należy napisać

```
{\footnotesize
\renewcommand{\baselinestretch}{2}\normalsize
Zawartość akapitu \par}
```

¶ Przykład 4.11.

W tym akapicie odległość pomiędzy wierszami jest dwa razy większa niż domyślna w całym dokumencie, a więc w instrukcji \baselinestretch argument wartość jest równy 2. Zmiana odstępów pomiędzy wierszami (wielkości interlinii) tylko w wybranym akapicie wymaga bardziej skomplikowanych zabiegów.

□

Instrukcja \vfill (skrócona forma \vspace{\fill}), wstawiona po zakończonym akapicie, spowoduje wypełnienie strony pustymi wierszami. Jeśli akapit ma być umieszczony u dołu strony (pozostała część strony pozostanie pusta), wtedy należy użyć formy z gwiazdką \vspace*{\fill},

ponieważ z definicji TEX (i LATEX) ignoruje wszelkie odstępy wstawiane na początku strony lub wiersza. Forma „z gwiazdką" likwiduje to ograniczenie. *Dodatkowe odstępy pionowe* można wstawiać używając gotowych poleceń `\bigskip`, `\medskip` i `\smallskip`. Wielkość każdego z tych odstępów zależy od wielkości użytego w dokumencie fontu – jeśli zmniejszymy lub powiększymy czcionkę dokumentu, odległości zadeklarowane tymi poleceniami zmienią się proporcjonalnie.

Ten sam efekt uzyskamy, jeśli w instrukcji `\vspace{odległość}` parametr `odległość` będzie wielokrotnością wielkości `ex`, czyli wysokości małej litery x w bieżącym foncie.

Uwaga: Aby polecenie wstawiające odstęp pionowy zostało wykonane tam gdzie chcemy, należy pamiętać o zakończeniu poprzedzającego je akapitu.

Środowiska `center`, `flushleft` i `flushright`

Tekst akapitu może być umieszczony na stronie w różny sposób. Zazwyczaj każdy wiersz akapitu jest wyrównany do obu marginesów, ale możemy też inaczej rozmieścić tekst. Aby cały akapit umieścić pośrodku strony należy skorzystać ze środowiska `center`.

```
\begin{center}
Akapit tekstu
\end{center}
```

Umieszczony w tym środowisku tekst zostanie automatycznie podzielony na wiersze krótsze niż szerokość tekstu i każdy z nich zostanie wyśrodkowany. Jeśli zależy nam na podzieleniu tekstu na linie o konkretnej długości lub zawartości, wówczas musimy każdą z nich złamać instrukcją `\\`.

```
\begin{center}
pierwsza linia tekstu \\
druga linia tekstu\\
....\\
```

```
n-ta  linia tekstu
\end{center}
```

¶ Przykład 4.12.

Jeżeli wyśrodkowujemy cały akapit tekstu i nie zależy nam na podzieleniu go na wiersze w szczególny sposób, wówczas możemy zaniechać łamania linii.

□

¶ Przykład 4.13.

Ten sam tekst
umieszczony w środowisku `center`, z wierszami
złamanymi przez autora,
będzie sformatowany inaczej.

□

Do wyśrodkowania tekstu jednego krótkiego wiersza można użyć polecenia `\centerline{tekst}`.

LaTeX domyślnie składa tekst wyrównując go do obu marginesów. Sposób wyrównywania możemy zmienić wykorzystując jedno ze środowisk opisanych poniżej.

Wyrównanie wierszy akapitu do jednego z marginesów uzyskamy korzystając ze środowiska `flushleft` (wyrównanie do lewego marginesu) lub `flushright` (wyrównanie do prawego marginesu). W obu tych środowiskach możemy samodzielnie podzielić wiersze lub zostawić tę pracę kompilatorowi.

Podobny efekt otrzymamy, gdy formatowany tekst umieścimy w środowisku `raggedleft` (odpowiednik `flushright`) lub `raggedright` (odpowiednik `flushleft`). Jeśli nie zastosujemy ręcznego łamania linii, to po umieszczeniu tekstu akapitu w środowisku `flushleft` otrzymamy skład tekstu w tzw. chorągiewkę.

Dla wyrównania tekstu do prawego lub lewego marginesu można użyć także jednego z poleceń

```
\flushleft{tekst}|,  \flushright{tekst}
\raggedleft{tekst}|,  \raggedright{tekst}.
```

Różnica pomiędzy środowiskami a poleceniami o tych samych nazwach polega na tym, że w przypadku tych ostatnich każda linia musi koniecznie być zakończona znakiem łamania (\\).

Uwaga: W żadnym z tych środowisk LaTeX nie przenosi wyrazów. Zamieszczone poniżej przykłady pokażą zastosowanie niektórych z omawianych środowisk i poleceń.

```
\begin{flushleft}
linia pierwsza \\
linia druga \\
linia trzecia \\
....\\
linia n
\end{flushright}
```

¶ Przykład 4.14.

Aktualny lider w rankingu ATP
ma w nim tak ogromną przewagę, że może przegrać
wszystkie pozostałe mecze tego
sezonu a i tak jej nie straci. Pozycję swoją zawdzięcza
sezonowi zeszłorocznemu, kiedy
to wygrał 41 spotkań z rzędu.

□

¶ Przykład 4.15.

Ten tekst jest argumentem polecenia `raggedright`.
Każda jego linia będzie ręcznie złamana i wyrównana
do lewego marginesu. Takie postępowanie jest wymuszone definicją
tego polecenia. Ostatnia linia także powinna być zakończona
znakiem `\\`.

□

Polecenie `\raggedright` możemy zastosować do dowolnego tekstu tak, by każdy wyraz zaczynał się w nowym wierszu, na przykład wyliczając dni tygodnia.

¶ Przykład 4.16.

Poniedziałek
Wtorek
Środa
Czwartek
Piątek
Sobota
Niedziela

□

I jeszcze tekst złożony automatycznie „w chorągiewkę" w środowisku `flushleft`.

¶ Przykład 4.17.

Z dawnych tradycji drukarstwa polskiego wiele dowodów kunsztu
literniczego znajdujemy w zachowanych starych drukach.
W muzeach i bibliotekach zachowane zostały wzorce rodzimej
twórczości. Próby stworzenia rodzimych warsztatów projektowania
liternictwa drukarskiego podejmowało dawniej rzemiosło
artystyczne.[1]

□

[1] Wyjątki z książki *Pismo i styl*, Tibor Szántó

Niekiedy zachodzi potrzeba wyróżnienia części tekstu w taki sposób, by jego odległość od lewego i prawego marginesu była taka sama. Do sformatowania tekstu możemy wtedy użyć jednego z dwóch środowisk `quote` lub `quotation` w następujący sposób:

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

Różnica między tymi dwiema formami polega na tym, że w środowisku `quote` akapity nie zaczynają się od wcięcia i są od siebie oddzielone dodatkowym odstępem, zaś w środowisku `quotation` tekst każdego akapitu rozpoczyna się wcięciem a odległość pomiędzy akapitami jest równa odległości miedzy wierszami. W obu przypadkach dodawany jest dodatkowy odstęp przed i za wyróżnionym tekstem.

Do składania wierszy możemy używać środowiska `verse`.

```
\begin{verse}
pierwsza linia pierwszej zwrotki\\
...
ostatnia linia pierwszej zwrotki
pierwsza linia drugiej zwrotki\\
...
ostatnia linia drugiej zwrotki
\end{verse}
```

A oto wiersz pt. Odrodzenie A. Puszkina w tłumaczeniu J. Tuwima (tekst umieszczony w środowisku `verse`):

> Artysta-wandal pędzlem sennym
> Powleka arcytwór czernidłem,
> Potem, bezmyślnie, skarb bezcenny
> Pokrywa własnym malowidłem.
>
> Z czasem się obce farby kruszą,
> Odpada obraz ów bezprawny

I znów stoimy przed geniuszem,
Przed jego arcydziełem dawnym.

Tak z duszy mojej umęczonej
Znikają błędy i wahania,
A wstają zjawy upragnione
Niewinnych, czystych dni zarania.

Tekst w kolumnach

LaTeX domyślnie składa tekst w jednej kolumnie. Jeżeli w deklaracji klasy dokumentu umieścimy opcję `twocolumn`, cały dokument zostanie złożony w dwóch kolumnach. Składanie tekstu w dwóch kolumnach jest dość trudne, szczególnie jeśli jest to tekst matematyczny. Z uwagi na długie wzory czy rysunki matematycy składają swoje prace w jednej kolumnie. Jeżeli jednak zechcemy część tekstu umieścić np. dwóch kolumnach, wygodnie jest skorzystać z możliwości pakietu `multicol` autorstwa Franka Mittelbacha. Znajdziemy tam środowisko `multicols`, które pozwala na składanie tekstu na przemian w jednej lub kilku kolumnach.

Pakiet `multicol` wczytujemy w preambule dokumentu a tekst przeznaczony do złożenia w kilku kolumnach umieszczamy w środowisku `multicols`, podając jako jego argument ilość kolumn.

```
\begin{multicols}{2}
... Wszystkie te misterne działania ...
\end{multicols}
```

Poniżej przykład, który zilustruje działanie pakietu `multicol`. Przy składaniu tekstu w dwóch i więcej kolumnach należy pamiętać, że długie wyrazy mogą nie mieścić się w pojedynczym wierszu.

¶ Przykład 4.18.

> ... Wszystkie te misterne działania służą im do zdobycia właściwego męża, w odpowiednim czasie. Japońskie małżeństwa, w większości przypadków są aranżowane i stanowią swoisty układ między dwoma rodzinami, które wyraziły zgodę na wzajemne powinowactwo. ...

□

¶ Przykład 4.19.
I jeszcze jeden przykład, tym razem tekst umieścimy w trzech kolumnach.

> Musiało minąć już ze cztery godziny, od czasu gdy kobieta wyszła do pracy. Akurat druga grupa odchodziła z koszami piasku w kierunku trójkołowego samochodu. Przysłuchiwał się uważnie i gdy stwierdził, że mężczyźni nie wracają, wstał cicho i włożył ubranie. Kobieta zabrała lampę, musiał więc zrobić wszystko po omacku. Kobieta była całkowicie pochłonięta pracą.
>
> Oczywiście nie uważa, że nie ma w tym żadnego niebezpieczeństwa. Nie wiadomo, czy na przykład za pół godziny nie zmienią swego zdania. Jest jeszcze sprawa tego urzędnika z prefektury.[2]

□

Składanie tekstu w dwóch lub więcej kolumnach może spowodować niechciane „dziury" pomiędzy wyrazami lub tak zwaną epidemię dywizów. Zawsze w takim przypadku należy zastanowić się, czy nie lepiej będzie złożyć nasz tekst w jednej kolumnie.

[2] W obu przykładach fragmenty książki Kobo Abe „Kobieta z wydm"

Łamanie wierszy

LaTeX łamie wiersze automatycznie, ale nie zawsze chcemy, by robił to za nas. Możemy go zmusić do złamania linii w konkretnym miejscu instrukcją `\linebreak[liczba]` lub nie pozwolić mu jej złamać instrukcją `\nolinebreak[liczba]`. Argument `liczba` jest opcjonalny i informuje jak ważne dla nas jest złamanie linii właśnie w tym miejscu. Może przyjmować wartości od 0 do 4 i im wyższa wartość tym polecenie jest silniejsze. Domyślnie argument przyjmuje wartość 4.

Inny sposób „ręcznego" złamania wiersza polega na umieszczeniu w miejscu złamania instrukcji `\\`. Dalszy ciąg tekstu pojawi się w kolejnym wierszu bez wcięcia akapitowego. Jeżeli dodamy opcjonalny argument `[odległość]` wówczas odległość pomiędzy tymi wierszami powiększy się o wartość argumentu `odległość`.

Instrukcja `\\` jest wykorzystywana głównie do wskazania końca wiersza w środowiskach takich jak `tabular`, `array` oraz w środowiskach do składania wzorów matematycznych w wielu liniach. Nie zaleca się stosowania jej w tekście ciągłym.

```
\noindent To jest ilustracja łamania wiersza\\
bez dodatkowego odstępu.
```

¶ Przykład 4.20.

To jest ilustracja łamania wiersza
bez zwiększania odstępu pomiędzy kolejnymi wierszami tekstu.
W tym przypadku zastosowana zostaje domyślna interlinia.

□

```
\noindent To jest ilustracja łamania wiersza\\[4mm]
z dodatkowym odstępem 4mm. Standardowa interlinia
 zostaje powiększona o 4mm.
```

¶ Przykład 4.21.

To jest ilustracja łamania wiersza

z dodatkowym odstępem 2mm.

□

Podobny efekt jak w przypadku \\ otrzymamy, gdy zastosujemy instrukcję `\newline`, z tym, że nie ma tu możliwości zwiększenia odstępu między wierszami.

Instrukcja `\linebreak` powoduje, że tekst w łamanej linii jest wyrównywany do lewego i prawego marginesu poprzez dodanie dodatkowych odstępów międzywyrazowych, podczas gdy instrukcje \\ i `\newline` powodują, że tekst jest uzupełniany do końca linii pustymi odstępami i odstępy między wyrazami pozostają niezmienione.

¶ Przykład 4.22.

Ten akapit został podzielony na wiersze instrukcją
`\newline`. Tekst został wyrównany do lewego marginesu.

□

¶ Przykład 4.23.

Akapit został podzielony na wiersze instrukcją `\linebreak`. Tekst w pierwszym wierszu został wyrównany do obu marginesów, ale pomiędzy wyrazy zostały dodane dodatkowe odstępy.

□

Uwaga. Zawsze, gdy ingerujemy w automatyczne łamanie wierszy musimy mieć świadomość, że może to spowodować złe sformatowanie całego dokumentu.

Łamanie stron

Kompilator LaTeX-a „czyta” cały akapit tekstu i dzieli go na wiersze, potem czyta następny akapit i dzieli go na wiersze i tak dalej, aż uzbiera akapitów z

naddatkiem na stronę. Wtedy łamie stronę we właściwym miejscu, a resztę materiału przenosi na następną. I znowu czyta kolejne akapity dzieląc je na wiersze, łamie stronę itd. Dzięki takiej organizacji łamania udaje się uniknąć zbyt dużych, niechcianych odstępów między wyrazami.

Możemy mu tę automatyczną pracę zakłócić i, podobnie jak w przypadku linii, możemy zażyczyć sobie złamania strony w konkretnym, wygodnym dla nas miejscu. Mamy dwie instrukcje pozwalające łamać stronę w dowolnym miejscu. `\pagebreak[liczba]` i `\newpage`. Argument `liczba` jest opcjonalny i informuje jak ważne jest złamanie strony właśnie w tym miejscu. Może przyjmować wartości od 0 do 4. Im wyższa wartość tym zalecenie jest silniejsze. Domyślnie jest przyjęta wartość 4. Instrukcje te w nowych implementacjach LaTeX-a nie różnią się. Obie powodują zakończenie strony w miejscu pojawienia się, wypełnienie jej pustymi odstępami i rozpoczęcie nowej.

Instrukcja `\nopagebreak` umieszczona między paragrafami nie pozwala LaTeX-owi na złamanie tam strony, a umieszczone wewnątrz paragrafu zabrania złamania strony na końcu bieżącej linii.

Formatowanie akapitu

Każdy akapit tekstu domyślnie rozpoczyna się wcięciem. Jego wielkość regulowana jest poleceniem `\parindent`. Jeśli przed rozpoczęciem nowego akapitu umieścimy deklarację `\setlength{\parindent}{2cm}` to następny i kolejne akapity będą się rozpoczynać dwucentymetrowym wcięciem. Jeśli tylko kilka akapitów naszego tekstu ma mieć wcięcie inne niż pozostałe należy je zgrupować poprzez umieszczenie w nawiasach klamrowych.

¶ Przykład 4.24.

To jest wcięcie akapitowe o długości 2 cm. Ten tekst służy do zilustrowania różnej długości wcięć akapitowych. □

```
{\setlength{\parindent}{2cm}
To będzie wcięcie akapitowe o długości 2 cm. Ten tekst
służy do zilustrowania różnej długości wcięć akapitowych.}
```

Instrukcja `\noindent` likwiduje wcięcie akapitowe, a `\indent` wstawia je na początku wiersza, który nie rozpoczyna akapitu.

Dzielenie wyrazów

LaTeX ma wbudowany algorytm dzielenia wyrazów dla języka angielskiego. Jeżeli używamy języka polskiego to razem z pakietem `polski` wczytywany jest też algorytm dzielenia wyrazów w języku polskim. W pojedynczych przypadkach może zaistnieć konieczność ręcznego podzielenia jakiegoś wyrazu. Można wtedy podzielić wyraz ręcznie wstawiając w miejscu podziału znak `\-`. Dzielenie wyrazów w taki sposób należy stosować bardzo ostrożnie gdyż dodanie nawet jednego znaku (słowa) przed miejscem podziału może zupełnie zmienić formatowanie. Możemy też „podpowiedzieć" LaTeX-owi gdzie może długi wyraz podzielić pisząc np.
`kon\-stan\-ty\-no\-po\-li\-tańczykiewiczówna`.

Wyrazy, które występują często w dokumencie i są błędnie przenoszone mogą być umieszczone w preambule jako argument polecenia `\hyphenation{lista}`(często może to dotyczyć nazwisk lub terminów naukowych). `Lista` to zbiór naszych długich słów lub nazwisk, oddzielonych od siebie spacjami, w których miejsca dopuszczalnego podziału zaznaczone są łącznikiem (dywizem).

¶ Przykład 4.25.

```
\hyphenation{kar-ko-ło-mny kon-stan-ty-no-po-li-tań-czyk
 pod-noś-nik ...}
```

□

Jeżeli z jakiegoś powodu chcemy aby w części tekstu LaTeX dzielił jak najmniej wyrazów, możemy umieścić nasz tekst w środowisku `sloppypar`.

```
\begin{sloppypar}
tekst paragrafu
\end{sloppypar}
```

W takim przypadku dodawane są dodatkowe odstępy między wyrazami i linia tekstu jest traktowana jako zbiór poszczególnych wyrazów a nie jako całość, a przenoszenie wyrazów zostanie wtedy zredukowane do minimum. Nie pojawiają się też informacje o przekroczeniu dozwolonej szerokości tekstu nawet, jeśli taka sytuacja ma miejsce.

Użycie środowiska `sloppypar` jest wskazane przy wąskich kolumnach tekstu. Podobnie działa polecenie `\sloppy`.

Fantomy

Fantomy są to niewidoczne odstępy zajmujące dokładnie tyle miejsca co argument umieszczony w nawiasach klamrowych. Wywołuje się je poleceniem `\phantom{argument}`. Pozwala ono zarezerwować miejsce na znaki które nie mają się pojawić w wydruku. Stosuje się je zwykle we wzorach matematycznych, aby uzyskać przejrzystość oznaczeń szczególnie gdy występują w nich wskaźniki górne i dolne, tak jak w przykładzie poniżej.

$$\Gamma_{ij}^{\ k} \quad \text{i} \quad \Gamma_{ij}^{k}$$

```
\Gamma_{ij}^{\phantom{ij}k}
\qquad \textrm{i} \qquad
\Gamma_{ij}^{k}
```

Notki na marginesie

Notkę na marginesie dokumentu generuje polecenie `\marginpar`. Ma ono jeden argument obowiązkowy którym jest tekst mający pojawić się na marginesie. Tekst rozpoczyna się na wysokości wiersza, w którym pojawiło się polecenie. Domyślnie tekst notki pojawia się na prawym marginesie dokumentu składanego jednostronnie a w przypadku dokumentu składanego dwustronnie na marginesie zewnętrznym. W przypadku tekstu sformatowanego

notka na marginesie

w dwóch kolumnach tekst notki pojawi się na marginesie bliższym kolumnie, w której użyliśmy polecenia `\marginpar`.

Tekst notki jest wstawiany w pudełko (domyślnie o szerokości 1.9 cm) i trzeba go samodzielnie podzielić na linijki.

Polecenie `\marginpar` można przedefiniować tak, by notki były pisane bardzo małą czcionką. Jeżeli w całym dokumencie mają być składane mała czcionką, polecenia zmieniające domyślną wielkość notki należy umieścić w preambule dokumentu.

```
\newcommand{\marginal}[1]{%
\leavevmode\marginpar{\tiny\raggedright#1\par}}
```

Polecenie `\marginpar{\rule[-10mm]{1mm}{15mm}}` umieszczone w pierwszej linii akapitu spowoduje zaznaczenie całego akapitu pionową linią umieszczoną na marginesie.

4.2. Struktury definiowane przez użytkownika

Teksty twierdzeń czy definicji są składane w szczególny sposób. Każde twierdzenie rozpoczyna się nazwą „Twierdzenie" i numerem, a jego treść jest składana pismem pochyłym (kursywą). Definicja rozpoczyna się nazwą „Definicje" i numerem, a jej treść jest składana najczęściej pismem prostym. LaTeX oferuje nam środowisko `\newtheorem`, które możemy zaopatrzyć w stosowną etykietę zgodnie z regułą

```
\newtheorem{nazwa}[licznik]{etykieta}
```

Otrzymamy środowisko rozpoczynające się od podanej etykiety (np. Twierdzenie) z numerem, takie jak w przykładach poniżej.

¶ Przykład 4.26.

Twierdzenie 1. *Treść tego twierdzenia jest złożona kursywą. Jest to domyślna czcionka dla środowiska* `\newtheorem`*. Etykieta wraz z numerem jest pogrubiona.*

□

¶ Przykład 4.27.

Definicja 1. *Treść tej definicji jest złożona kursywą. Jest to domyślna czcionka dla środowiska* `\newtheorem`*. Aby treść definicji złożyć pismem prostym należy w preambule dokumentu wczytać pakiet amsthm. Pakiet ten opisany jest w rozdziale 6.3.*

□

Twierdzenia, definicje i inne środowiska typu `\newtheorem` są numerowane oddzielnie, a numeracja kontynuuje się do końca dokumentu. Aby w każdym nowym rozdziale uzyskać numerowanie od początku, należy w definicji nowego twierdzenia

```
\newtheorem{nazwa}[licznik]{etykieta}[z licznikiem]
```

jako argument `[z licznikiem]` umieścić `section` lub `chapter`.

Jeżeli używamy klas zawierających wzorce składania dokumentów pochodzące z języka angielskiego (article, report i book), to po numerze rozdziału, punktu itd. nie będzie kropki (w dokumentach pisanych w języku polskim po numerze twierdzenia umieszczamy kropkę). Ta sama zasada dotyczy twierdzeń i definicji – brak kropki po numerze. Domyślne numerowanie rozdziałów musimy wtedy zmienić samodzielnie w sposób opisany na stronie 101 i następnych, a w przypadku twierdzeń i definicji najlepiej jest skorzystać z pakietu `amsthm` opisanego w rozdziale 6.3.

Jeżeli twierdzenia, lematy, definicje itd. mają być numerowane wspólnie, licznik jednego (licznik nazywa się tak samo jak środowisko) powtarzamy

w nawiasach kwadratowych w definicji następnego (przed etykietą środowiska):

```
\newtheorem{aks}{Aksjomat}
\newtheorem{uw}[aks]{Uwaga}
\newtheorem{lemm}[aks]{Lemat}
```

Definiowanie nowych poleceń i środowisk

Nowe polecenie definiuje się za pomocą instrukcji `\newcommand`: `\newcommand{\nowa_nazwa}{jak ma działać}`.

```
\newcommand{\rb}{\mathbb R}
\newcommand{\cp}{\mathcal P}
```

Aby otrzymać $\mathbb{R}$ i $\mathcal{P}$, wystarczy teraz w trybie matematycznym napisać `$\rb$` i `$\cp$`.

Tej samej instrukcji `\newcommand` można użyć do skrócenia nazwy jakiegoś polecenia, na przykład otwarcia i zamknięcia środowiska o długiej nazwie.
`\newcommand{\nowa_nazwa}{stara_nazwa}`.
Wpisując wzór wyśrodkowany

$$a^2 + b^2 = c^2 \tag{4.1}$$

wystarczy napisać

```
\newcommand{\be}{\begin{equation}}
\newcommand{\ee}{\end{equation}}
\be
a^2+b^2 = c^2
\ee
```

Aby zmienić działanie istniejącego polecenia możemy użyć instrukcji `\renewcommand{\nazwa_polecenia}{nowa_postać}`.

Jeśli nową definicję polecenia umieścimy w preambule, będzie działać w całym dokumencie, a jeżeli umieścimy ją wewnątrz innego środowiska, jej działanie skończy się wraz z zamknięciem środowiska zewnętrznego.

Nowe środowisko definiujemy za pomocą polecenia `\newenvironment`, a już istniejące zmieniamy poleceniem `\renewenvironment`.

Środowisko zdefiniowane przez nas (tak jak i nowe polecenie zdefiniowane przez `\newcommand`) może posiadać do 9 dodatkowych argumentów.

```
\renewenvironment{nazwa}[larg][arg_opcj]
{początek}{koniec}
\renewcommand{nazwa}[larg][arg_opcj]{początek}{koniec}
```

Środowiska bez argumentów

Wyobraźmy sobie, że w przygotowywanym przez nas dokumencie będziemy umieszczać wielokrotnie różne partie tekstu, które powinny być złożone kursywą i umieszczone w blokach węższych niż cały tekst. Możemy dla wygody, zdefiniować nowe środowisko (używając polecenia `\newenvironment`) które samo będzie formatować tekst zgodnie z naszymi oczekiwaniami.

```
\newenvironment{ntek}{\begin{quote}\itshape}
{\end{quote}}
```

W definicji użyliśmy opisanego wcześniej środowiska `quote`.

Przed przykładem ilustrującym zastosowanie nowego środowiska umieszczamy tekst, który służy do zaznaczenia długości wiersza tekstu głównego.

> *To jest tekst, który chcemy umieścić w nowym środowisku automatycznie składającym zawartość kursywą i zwężającym całość w stosunku do tekstu głównego. Nazwaliśmy je* `ntek`*, otwiera je polecenie* `\begin{ntek}` *i zamyka* `\end{ntek}`.

Po przykładzie również umieszczamy dodatkowy tekst, który służy do zaznaczenia długości wiersza tekstu głównego.

Polecenia z argumentem

Dla ułatwienia wprowadzania niektórych partii tekstu możemy zdefiniować polecenie z argumentem. Takie polecenie posiada tekst wspólny i argument, który możemy zmieniać. Definiuje się je w sposób pokazany poniżej.

```
\newcommand{\rec}[1]{%
\begin{center}\small
{\it Wpłynęło #1}
\end{center}}
```

Jeśli napiszemy: `\rec{25 listopada 2010 r}` to otrzymamy

Wpłynęło 25 listopada 2010 r.

I jeszcze jeden przykład. W pewnym dokumencie powtarza się często długi wyraz np. *półtłusty* odmieniany w różnych przypadkach i rodzajach. Aby skrócić czas wpisywania dość skomplikowanego tekstu możemy zdefiniować nowe polecenie z parametrem:
`\newcommand{\ptt}[1]{półtłust{#1}}` i używać na przykład tak
`\ptt ych`, `\ptt ymi` – półtłustych, półtłustymi.

Środowisko z argumentem

LaTeX dostarcza nam także mechanizmów do definiowania środowisk z argumentem.

```
\newenvironment{bal}[1]%
{\itshape {\dots #1}}{\dots}
\begin{bal}
Niech żyje bal, bo to życie to bal jest nad bale
\end{bal}
```

¶ Przykład 4.28.

. . . Niech żyje bal, bo to życie to bal jest nad bale . . . □

4.3. Przypisy

Przypisy w tekście wstawiamy poleceniem `\footnote{tekst}`. W miejscu jego umieszczenia pojawia się numer przypisu, a na dole strony jego treść opatrzona tym samym numerem. Numer przypisu powinien następować bezpośrednio po wyrazie którego dotyczy. więc nie dodaje się odstępu pomiędzy wyrazem i poleceniem `\footnote{tekst}`.

W klasie `article` wszystkie przypisy są numerowane w sposób ciągły, w klasach `book` i `report` rozpoczynają się od pierwszego wraz z rozpoczęciem nowego rozdziału. Jeśli przygotowujemy dokument w klasie `article` i chcemy, by przypisy w każdym rozdziale numerowane były od początku, należy rozpoczynając rozdział wyzerować licznik przypisów poleceniem `\setcounter{footnote}{0}`.

Przypis nie pojawi się, gdy umieścimy go w środowisku `\parbox`, tylko w miejscu wpisania polecenia zostanie wstawiony jego numer.

```
\parbox{7cm}{To jest przypis\footnote{przypis}
w~pudełku \texttt{parbox}}
```

To jest przypis[3] w pudełku `parbox`

W pudełku `minipage` przypis pojawi się u dołu ministrony, a nie na dole strony bieżącej.

¶ Przykład 4.29.

Chociaż w środowisku `minipage`[a] możemy użyć polecenia `\footnote{}`, przypis pojawi się u dołu ministrony a nie na dole bieżącej strony.

[a] Przypis pojawi się u dołu ministrony

□

W środowiskach matematycznych zazwyczaj polecenie `\footnote` działa poprawnie.

$$x + y = 12^4 \tag{4.2}$$

[4] Sprawdzenie działania polecenia tworzącego przypis

Przypisy niestandardowe

Wszędzie tam, gdzie nie możemy użyć normalnego trybu wstawiania przypisu lub chcemy wstawić przypis niestandardowy, powinniśmy skorzystać jednocześnie z dwóch poleceń:

```
\footnotemark[numer] i  \footnotetext[numer]{tekst}
```

Numer przypisu wstawiamy ręcznie za pomocą pierwszego z poleceń, a jego tekst wstawiamy jako argument drugiego polecenia bezpośrednio po zamknięciu środowiska , czyli już w trybie akapitowym. Należy potem pamiętać o zwiększeniu licznika `footnote` o 1, gdyż numer wstawiany ręcznie nie jest dodawany. Polecenie `\footnotemark[numer]` nie działa w pudełku `parbox`.

Przypisy są domyślnie numerowane kolejnymi liczbami arabskimi, a w środowisku `minipage` małymi literami alfabetu łacińskiego.

Styl numerowania przypisów możemy zmienić za pomocą `\renewcommand`

```
\renewcommand{\thefootnote}{\styl_numer{footnote}}
```

Dostępne są następujące style:

```
\arabic, \roman, \Roman, \alph i \Alph
```

oraz dziewięć symboli wywoływanych poleceniem `\fnsymbol`. Są to symbole (w kolejności występowania w zbiorze)
∗, †, ‡, §, ¶, ‖ , ∗∗ , ††, ‡‡.

Gdy numerujemy symbolicznie, po poleceniu `\footnote` umieszczamy w nawiasach kwadratowych numer symbolu.

Utworzymy teraz przypis[§] oznaczony znakiem paragrafu.

```
\footnote[4]{oznaczony znakiem paragrafu.}
```

Do domyślnego numerowania przypisów wracamy umieszczając wewnątrz dokumentu instrukcję

[§]oznaczony znakiem paragrafu (czwarty w kolejności).

```
\renewcommand{\thefootnote}{\arabic{footnote}}
```

i ewentualnie zmieniając numer kolejnych przypisów.

```
\setcounter{footnote}{liczba}
```

Zmiana sposobu numerowania rozdziałów i równań

Licznik rozdziałów (chapter) nazywa się `chapter`, za jego wygląd odpowiada instrukcja `\thechapter`, punktów (section) nazywa się `section`, za jego wygląd odpowiada instrukcja `\thesection` a licznik podpunktów (subsection) nazywa się `subsection` i za jego wygląd odpowiada instrukcja `\thesubsection`. Licznik równań nazywa się `equation`, a za jego wygląd odpowiada instrukcja `\theequation`.

Wygląd każdego z tych liczników zmieniamy poleceniem

```
\renewcommand{zmieniana_komenda}{nowa_jej_postać}
```

- Zmiana numeracji równań – numery podwójne, pierwszy to numer punktu (section), drugi kolejny numer równania.

  ```
  \renewcommand{\theequation}%
  {\thesection\arabic{equation}}
  ```

- Zmiana numeracji rozdziałów – dodanie kropki po numerze.

  ```
  \renewcommand{\thechapter}{\arabic{chapter}.}
  ```

- Zmiana numeracji punktów – dodanie kropki po numerze.

  ```
  \renewcommand{\thesection}{\thechapter%
  \arabic{section}.}
  ```

- Dodanie kropki po numerze podpunktu.

  ```
  \renewcommand{\thesubsection}{\thechapter%
  \thesection\arabic{subsection}.}
  ```

- Zmiana numeracji punktów – numerowanie liczbami rzymskimi (małymi).

```
\renewcommand{\thesection}%
{\roman{section}.}
```

Jeśli dokument nie jest podzielony na rozdziały a tylko na punkty i podpunkty, to w powyższych definicjach należy usunąć `\thechapter`.

4.4. Odsyłacze

LaTeX ma zdefiniowany mechanizm tworzenia odsyłaczy do wszystkich numerowanych środowisk oraz do stron. W każdej pracy naukowej lub technicznej znajdziemy odesłania do wzorów, twierdzeń, tabel, list czy literatury. Składając taką pracę LaTeX-em powinniśmy skorzystać z tego mechanizmu albowiem ręczne wpisywanie numerów odsyłaczy może prowadzić do wielu błędnych adresów i jest bardzo czasochłonne, szczególnie gdy uzupełniamy treść dokumentu w trakcie pisania.

Do wstawiania odsyłaczy do wzorów, twierdzeń, tabel i list służy następująca para instrukcji

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

gdzie `etykieta` to dowolny ciąg liter, cyfr i znaków interpunkcyjnych z wyjątkiem znaków specjalnych

```
#, $, %, &, \, ^, _, {, }.
```

¶ Przykład 4.30.
Poniżej przykład odesłania do zacytowanego twierdzenia

Twierdzenie 1. *Jeśli pierścień $\mathcal{P}$ nie ma dzielników zera i $\varphi \neq 0$, $\psi \neq 0$, to $st\varphi\psi = st\varphi + st\psi$.*

W twierdzeniu 1 podaliśmy własność pierścienia $\mathcal{P}$. □

W przykładzie zdefiniowaliśmy środowisko `tw` i wewnątrz niego umieściliśmy etykietę, którą później przywołaliśmy instrukcją `\ref`.

```
\newtheorem{tw}{Twierdzenie}
\begin{tw}\label{tw:1}
treść twierdzenia
\end{tw}
W~twierdzeniu \ref{tw:1} podaliśmy
```

Za pomocą tej samej pary instrukcji możemy odesłać czytelnika do numerowanego wzoru, tak jak w przykładzie następnym

¶ Przykład 4.31.

Najpierw umieścimy równość sprawdzającą umiejętność wpisywania wyrażeń i oznaczeń matematycznych.

$$\int \frac{dx}{(x-k)^2+b} = \int \frac{\sqrt{b}dt}{bt^2+b} = \frac{1}{\sqrt{b}} \operatorname{arc\,tg} t + C. \qquad (4.3)$$

W równaniu (4.3) umieściliśmy różne wyrażenia i znaki matematyczne. □

Odsyłacz w ostatnim przykładzie powstał następująco:

```
\begin{equation}\label{r1}
tu wzór
\end{equation}
W równaniu \eqref{r1} umieściliśmy
```

Instrukcja `\eqref{}` jest rozszerzeniem instrukcji `\ref{}` o dodatkowe nawiasy, w które jest ujęty numer wzoru. Mamy ją do dyspozycji po wczytaniu pakietu `amsmath`.

Te same instrukcje pozwalają także odsyłać czytelnika do konkretnych rozdziałów naszego dokumentu, na przykład do rozdziału 1 w którym wyjaśniamy jak rozpocząć pracę z LaTeX-em, natomiast odesłanie do strony w dokumencie, na przykład do strony 118 na której omawiamy zastosowanie środowiska `minipage`, realizuje para instrukcji `\label{etykieta}`

i `\pageref{etykieta}`. Etykietę umieszczamy gdziekolwiek wewnątrz tekstu na stronie, do numeru której zostaniemy odesłani przez instrukcję `\pageref`. Etykietę pozwalającą odesłać czytelnika do konkretnego rozdziału umieszczamy bezpośrednio po jego nazwie, np.:

```
\chapter{nazwa}\label{etykieta}
```

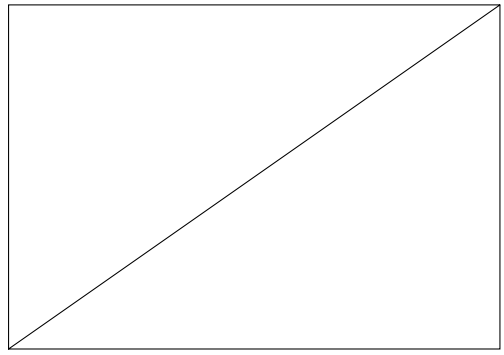
Wstawianie odsyłaczy w środowiskach takich jak rysunek czy tabela wymaga większej uwagi. Po pierwsze nasz rysunek musi być umieszczony wewnątrz środowiska **`figure`** (a tabela wewnątrz **`table`**). W każdym z tych środowisk zdefiniowane jest polecenie `\caption{}` odpowiedzialne za podpis pod lub nad rysunkiem (lub tabelą). Oba one mają też zdefiniowany licznik (o nazwie caption), którego wartość drukowana jest w podpisie (lub nagłówku) danego środowiska bezpośrednio po nazwie środowiska. Etykieta (`label`) odsyłacza musi być umieszczona wewnątrz argumentu polecenia `caption` (tak jak w przypadku rysunku 4.1), gdyż tylko wtedy numeracja odsyłaczy będzie zawsze prawidłowa (nawet przy składaniu dużych dokumentów, których części wczytywane są poleceniem `\input` lub `\include`). Jeśli w takim dokumencie umieścimy etykietę odsyłacza bezpośrednio po otwarciu środowiska **`figure`** (tak jak ma to miejsce na przykład w przypadku środowiska **`equation`**) wówczas numeracja rysunków w kolejnych częściach dokumentu może nie być poprawna.

¶ Przykład 4.32.

Umieścimy teraz rysunek prostokąta w środowisku **`figure`**, zaopatrzymy go w podpis i zacytujemy w tekście powyżej rysunku, umieszczając tam instrukcję `\ref{x}`. Miejsce umieszczenia odsyłacza jest dowolne, należy jednak pamiętać o konieczności dwukrotnego skompilowania dokumentu. □

```
\begin{figure}[h]
tu wstawiony jest rysunek
```

```
\caption{To jest prostokąt z jedną przekątną\label{x}}
\end{figure}
```



Rysunek 4.1: To jest prostokąt z jedną przekątną

Podpis pod rysunkiem zaopatrzyliśmy w etykietę odsyłacza i przywołaliśmy ją instrukcją `\ref{x}`.

4.5. Tworzenie list

Środowiska tworzące listy dostępne w LaTeX-u to tworzące listy wypunktowane **itemize**, numerowane **enumerate** oraz dające możliwość wstawienia etykiety tekstowej **description**. Jeżeli żadne z tych środowisk nie spełnia naszych oczekiwań, możemy zdefiniować własne wpisując odpowiednie parametry jako argumenty środowiska **list**. Poniżej przykłady środowisk **itemize**, **enumerate** i **description**

```
\begin{itemize}
\item tekst
.........
\end{itemize}
```

- tekst

- tekst

```
\begin{enumerate}
\item tekst
\item tekst
.........
\end{enumerate}
```

1. tekst
2. tekst
3. tekst

```
\begin{description}
\item[Etykieta] tekst
\item[Etykieta] tekst
.........
\end{description}
```

Etykieta tekst

Etykieta tekst

Etykieta tekst

Środowisko `itemize` i `enumerate`

W środowisku (`itemize`) kolejne punkty zaczynają się etykietą, która zależy od użytej klasy. I tak w klasie `article` jest to gruba czarna kropka, w klasie `mwart` – myślnik. Grubą kropkę uzyskuje się w TEX-u poleceniem `$\bullet$` (użytym w trybie matematycznym). Poniżej przykład listy wypunktowanej.

¶ Przykład 4.33.

- Treść pierwszego punktu listy.
- Treść drugiego punktu listy.

□

```
\begin{itemize}
\item Treść pierwszego punktu listy.
\item  Treść drugiego punktu listy.
\end{itemize}
```

W środowisku numerującym (`enumerate`) tekst kolejnych punktów poprzedzony jest etykietą kolejnego numeru.

```
\begin{enumerate}
\item To jest punkt pierwszy.
\item To jest punkt drugi.
\item To jest punkt trzeci.
\end{enumerate}
```

1. To jest punkt pierwszy.
2. To jest punkt drugi.
3. To jest punkt trzeci.

Środowisko `list`

Użytkownik LaTeX-a może zbudować własne środowisku typu lista, definiując odpowiednio do swoich potrzeb poszczególne argumenty środowiska `list`. Środowisko to wywołuje się tak samo jak `enumerate` czy `itemize` ale, w odróżnieniu od nich, pozwala ono na sterowanie szerokością wierszy listy, odstępem pomiędzy kolejnymi punktami itp. Jest to możliwe dzięki parametrom, których wartości możemy zadeklarować bezpośrednio po wywołaniu środowiska.

```
\begin{list}{etykieta}{lista_deklaracji}
\item tekst
\item tekst
\end{list}
```

Parametr `etykieta` ustala domyślną etykietę rozpoczynającą kolejne punkty listy. Wszystkie będą mieć taką samą etykietę, a jej zmiana w dowolnym punkcie listy jest możliwa poprzez umieszczenie innego znaku po poleceniu `\item` jak poniżej:

```
\begin{list}{\#}{}
\item Znak ustalony w~definicji listy.
\item[\&] A ten punkt ma zmieniony znak rozpoczynający go.
\item I znowu ustalony w~definicji znak.
\end{list}
```

Taką zmianę zastosowaliśmy w przykładzie 4.34.

¶ Przykład 4.34.
Zdefiniujemy listę z ustaloną etykietą dla wszystkich punktów a następnie w jednym punkcie zamienimy ją na inną.

To jest punkt rozpoczynający się znakiem ustalonym w definicji listy.

To też jest punkt rozpoczynający się znakiem ustalonym w definicji listy.

& A ten punkt ma zmieniony znak rozpoczynający go.

I znowu ustalony w definicji znak.

□

W definicji powyższej listy druga para nawiasów klamrowych pozostała pusta co oznacza, że użyte zostały domyślne ustawienia odstępów poziomych i pionowych listy. Zajmiemy się teraz parametrem `lista_deklaracji` czyli spisem geometrycznych parametrów formatujących naszą listę, takich jak szerokość etykiety, odstęp pomiędzy kolejnymi punktami listy itd.

`\topsep` dodatkowy odstęp pionowy przed i po liście.

`\partopsep` dodatkowy odstęp od góry jeśli środowisko poprzedza pusta linia.

`\itemsep` dodatkowy odstęp pomiędzy punktami listy (dodany do domyślnego odstępu między wierszami).

`\parsep` dodatkowy odstęp pomiędzy akapitami wewnątrz punktu listy.

`\leftmargin` poziomy odstęp pomiędzy lewym marginesem a listą (musi być nieujemny).

`\rightmargin` poziomy odstęp pomiędzy prawym marginesem a listą (musi być nieujemny).

`\listparindent` zwiększenie (zmniejszenie) wcięcia akapitu po etykiecie.

`\itemindent` wcięcie pierwszej linii tekstu kolejnego punktu listy (może być ujemne).

`\labelsep` poziomy odstęp pomiędzy końcem pudełka zawierającego etykietę a tekstem.

`\labelwidth` domyślnie szerokość pudełka zarezerwowanego na etykietę; jeżeli etykieta jest większa niż zarezerwowane miejsce – zajmuje część miejsca w pierwszym wierszu punktu listy,

`\makelabel{label}` tworzy drukowaną etykietę

Przykłady zastosowania środowiska `list`

Przedstawimy teraz kilka przykładów list, w których zostaną kolejno zdefiniowane nowe wartości parametrów geometrycznych. Będą one dobierane tak, by od razu można było zobaczyć co zostało zmienione. Na początek standardowa lista wypunktowana umieszczona w środowisku `itemize`. Każdy przykład będzie zawarty pomiędzy liniami demonstrującymi aktualną szerokość tekstu.

Szerokość tekstu

- To jest pierwszy punkt listy w środowisku `itemize` o domyślnych parametrach geometrycznych.
- A to drugi punkt tej samej listy aby zilustrować domyślne odstępy pomiędzy kolejnymi jej punktami.

Szerokość tekstu

  - Parametr `\leftmargin` ma teraz wartość 2 cm, reszta ustawień domyślna.

Szerokość tekstu

- Parametr `\rightmargin` ma teraz wartość 3 cm,
  reszta ustawień domyślna.

Szerokość tekstu

- Odstęp od poprzedzającego i następnego akapitu czyli `\topsep` został powiększony o 1.5 cm, reszta ustawień domyślna.

Szerokość tekstu

⇔ Jako pierwszy parametr listy poleceniem `\makelabel{}` został wpisany znak, który ma być etykietą każdego punktu, reszta ustawień domyślna.

⇔ Drugi punkt listy.

Szerokość tekstu

⇔ W tej samej liście dodatkowo zmienimy parametr `\labelsep` i nadamy mu wartość 1 cm, reszta ustawień domyślna.

⇔ Drugi punkt listy.

Szerokość tekstu

Teraz parametrowi `\labelsep` nadaliśmy wartość ujemną (-0.5 cm), parametrowi `\labelwidth` wartość 1 cm, pierwszy parametr listy pozostawiliśmy taki jak poprzednio, a reszta ustawień jest domyślna.

Drugi punkt listy. Okazuje się, że w tym przypadku etykieta została wydrukowana na tekście.

Szerokość tekstu

⇔ Jeszcze ustalmy parametr `\labelwidth` równy 0 cm, pierwszy parametr listy pozostawiamy taki jak poprzednio, reszta ustawień domyślna.

⇔ Drugi punkt listy.

Szerokość tekstu

∗ I na koniec zmienimy parametr `\itemindent` i nadamy mu wartość 2 cm, zmienimy również pierwszy parametr listy na inny znak, reszta ustawień zostaje domyślna.

∗ Drugi punkt listy.

Parametry środowiska `list` pozwalają na skonstruowanie listy odpowiadającej bardzo różnym potrzebom użytkownika. Zajmowaliśmy się dotychczas tylko listami wypunktowanymi, ale za pomocą środowiska `list` można też utworzyć listę numerowaną. Wystarczy poleceniem `\newcounter` utworzyć licznik, a następnie wstawić go w miejsce deklaracji etykiety. Poniżej najprostsza taka lista:

¶ Przykład 4.35.

I. Ta lista jest numerowana liczbami rzymskimi.

II. Utworzyliśmy nowy numer o nazwie *numer*.

□

```
\newcounter{numer}
\begin{list}{\Roman{numer}. }{\usecounter{numer}}
\item Ta lista jest numerowana liczbami rzymskimi.
\item Utworzyliśmy nowy numer o nazwie \textit{numer}.
\end{list}
```

W tej samej liście zmniejszymy teraz odstęp pomiędzy kolejnymi punktami zmieniając wartość polecenia `\itemsep`.

¶ Przykład 4.36.

I. Ta lista jest numerowana liczbami rzymskimi.
II. Utworzyliśmy nowy numer o nazwie *numer*.

□

```
\newcounter{numer}
\begin{list}{\Roman{numer}. }{\usecounter{numer}%
\setlength{\itemsep}{-6pt}}
\item Ta lista jest numerowana liczbami rzymskimi.
\item Utworzyliśmy nowy numer o nazwie \textit{numer}.
\end{list}
```

Jeszcze dwa przykłady list z innymi etykietami numerowanymi i zmniejszonym odstępem pomiędzy punktami.

A–I To jest punkt pierwszy.
A–II To jest punkt drugi.
A–III To jest punkt trzeci.
A–IV I jeszcze punkt czwarty.

Taka lista nie wygląda zbyt ładnie, spróbujemy więc wyrównać etykiety poszczególnych punktów.

A–I To jest punkt pierwszy.
A–II To jest punkt drugi.
A–III To jest punkt trzeci.
A–IV I jeszcze punkt czwarty.

Teraz jest lepiej. Oto co i jak zmieniliśmy:

```
\begin{list}{{A--\Roman{numer}}}
{\usecounter{numer}\setlength{\itemsep}{-4pt}%
\setlength{\labelwidth}{2cm}%
\setlength{\leftmargin}{2cm}%
\setlength{\labelsep}{0.5cm}}
```

Poniżej zmieściliśmy jeszcze jeden przykład numerowanych etykiet w środowisku `list`

Rysunek 1: *Przykładowy spis rysunków.*

Rysunek 2: *Pokazuje inny rodzaj etykiety.*

Rysunek 3: *Treść kolejnych punktów listy jest napisana pismem pochyłym.*

oraz wartości parametrów takiej listy:

```
\begin{list}{\bfseries\upshape Rysunek \arabic{fig}:}%
{\usecounter{fig}%
\setlength{\labelwidth}{2.15cm}%
\setlength{\leftmargin}{2.6cm}%
\setlength{\labelsep}{0.5cm}%
\setlength{\rightmargin}{1cm}%
\setlength{\parsep}{0.5ex plus0.2ex minus0.1ex}%
\setlength{\itemsep}{0ex plus0.2ex minus0.1ex}%
\slshape}%
```

4.6. Pudełka

TEX składa tekst z pudełek, w każdym pudełku „siedzi" litera lub znak. Możemy więc traktować jako pudełko całe wyrazy lub akapity. LATEX oferuje użytkownikom pudełko typu LR – „left-right" w którym zawartość jest umieszczana horyzontalnie od lewej do prawej oraz pudełka akapitu. Dodatkowo jest jeszcze pudełko liniowe to znaczy zawierające linię. W rzeczywistości jest to czarny prostokąt generowany poleceniem

```
\rule[lift]{width}{height}.
```

Parametr `lift` służy do umieszczenia linii (wyżej lub niżej) względem linii bazowej, a parametry `width` i `height` to wymiary naszego prostokąta. Podamy teraz kilka przykładów linii różnej grubości w których parametr`lift` będzie przyjmował różne wartości.

¶ Przykład 4.37.

Najpierw krótka linia na poziomie tekstu ▬, teraz ta sama linia wyżej ▬ oraz niżej ▬.

□

```
Najpierw krótka linia na poziomie tekstu \rule{10mm}{1mm},
teraz ta sama linia wyżej \rule[2ex]{10mm}{1mm} oraz niżej
\rule[-2ex]{10mm}{1mm}.
```

Do umieszczenia krótkiego tekstu w pudełku typu LR możemy użyć jednego z poleceń `\mbox`, `\makebox`, `\fbox` i `\framebox`. `\mbox{tekst}` i `\fbox{tekst}` nie posiadają żadnych argumentów, natomiast pozostałe mają argumenty `[szerokość][pozycja]{tekst}`

Polecenie	Wynik
`\mbox{Treść}`	To jest Treść
`\makebox[2cm]{Treść}`	To jest Treść
`\fbox{Treść}`	To jest Treść
`\framebox[4cm][l]{Treść}`	To jest Treść
`\framebox[4cm][r]{Treść}`	To jest Treść
`\framebox[4cm][s]{Treść}`	To jest Treść w pudełku
`\framebox[2mm]{środek}`	środek
`\makebox[0pt][l]{/}S`	To jest /S

Parametr szerokość musi zawierać oprócz liczby także jednostkę, natomiast jako parametr `pozycja` mogą być podane następujące litery:

l – wyrównywanie tekstu do lewego marginesu,

r – wyrównywanie tekstu do prawego marginesu,

s – rozstrzelenie tekstu po całym pudełku.

TEX umieszcza litery, wyrazy i akapity w pudełkach możemy więc „poruszać" tymi pudełkami. I tak do podniesienia lub opuszczenia wyrazu w stosunku do linii bazowej możemy wykorzystać polecenie `\raisebox` w następujący sposób:

```
Linia bazowa \raisebox{1ex}{wyżej} a teraz
 \raisebox{-1ex}{niżej}i z powrotem.
```

Linia podstawowa $^{\text{wyżej}}$ i teraz $_{\text{niżej}}$ i znowu na linii podstawowej.

Do umieszczania całych akapitów w pudełku, którego położeniem w dokumencie można sterować, możemy zastosować polecenie `\parbox` lub użyć środowiska `minipage`. Zdefiniowane są następująco:

```
\parbox[pozycja]{szerokość}{tekst}

\begin{minipage}[pozycja]{szerokość}
tekst
\end{minipage}
```

gdzie parametr `pozycja` może przybierać wartości:
b - równanie do dolnego brzegu pudełka,
t - równanie do górnego brzegu pudełka,
a parametr `szerokość` musi być liczbą z mianem, na przykład 10 cm, 6 in.
Następne przykłady pokażą zastosowanie polecenia `\parbox`.

```
\parbox{3cm}{To jest przykład pudełka o szerokości 3cm
wyrównanego w pionie do \dots}
\hfill BIEŻĄCA LINIA \hfill
\parbox{5cm}{To jest przykład pudełka o szerokości 5 cm
wyrównanego w pionie do \dots}
```

¶ Przykład 4.38.

To jest przykład pudełka o szerokości 3cm wyrównanego w pionie do środka linii bazowej (domyślny parametr `c`)

BIEŻĄCA LINIA

To jest przykład pudełka o szerokości 5cm wyrównanego w pionie do środka linii bazowej (domyślny parametr `c`)

□

I jeszcze jeden przykład ilustrujący użycie polecenia `\parbox`.

```
\parbox[t]{3cm}{To jest przykład pudełka o szerokości
3 cm wyrównanego w pionie tak \dots}
\hfill BIEŻĄCA LINIA \hfill
\parbox[b]{5cm}{To jest przykład pudełka o szerokości
5cm wyrównanego w pionie tak \dots}
```

¶ Przykład 4.39.

To jest przykład pudełka o szerokości 5cm wyrównanego w pionie tak, by ostatnia linia pudełka leżała na linii bazowej (parametr `b`)

To jest przykład pudełka o szerokości 3cm wyrównanego w pionie tak, by pierwsza linia pudełka leżała na linii bazowej (parametr `t`)

BIEŻĄCA LINIA

□

Poniżej czytelnik znajdzie jeszcze kilka przykładów ilustrujących użycie środowiska `minipage`.

¶ Przykład 4.40.

To jest przykład pudełka o szerokości 3 cm wyrównanego w pionie do środka linii bazowej

BIEŻĄCA LINIA

To jest przykład pudełka (ministrony) o szerokości 5 cm wyrównanego w pionie ...

□

```
\begin{minipage}{3cm}
To jest przykład pudełka o szerokości 3 cm
\end{minipage}
\hfill BIEŻĄCA LINIA \hfill
\begin{minipage}{5cm}
To jest przykład pudełka o szerokości 5 cm
\end{minipage}
```

¶ Przykład 4.41.

Pudełko o szerokości 3 cm wyrównane w pionie tak, by pierwsza linia leżała na linii bazowej (parametr `t`)

BIEŻĄCA LINIA

Pudełko o szerokości 5 cm wyrównane w pionie tak, by ostatnia linia pudełka leżała na linii ... bazowej (parametr `b`)

□

```
\begin{minipage}{3cm}
Pudełko   o~szerokości 3 cm wyrównane ...
\end{minipage}
\hfill BIEŻĄCA LINIA \hfill
\begin{minipage}{5cm}
Pudełko  o~szerokości 5 cm wyrównane ...
\end{minipage}
```

W przykładzie 4.40 parametr `pozycja` jest domyślny czyli `c`, w przykładzie 4.41 jako parametr `pozycja` wstawiliśmy najpierw `t` a potem `b`. Pudełka `minipage` i `parbox` można też ze sobą łaczyć.

Należy jeszcze wspomnieć o różnicach dzielących `minipage` i `parbox`. Środowisko `minipage` zachowuje się tak jak cała strona dokumentu, można w nim umieszczać wszystkie te polecenia i instrukcje, które umieszcza się na stronie. Wyjątek stanowi przypis wywołany poleceniem `\footnote`. Jego treść znajdzie się u dołu ministrony, a nie na dole bieżącej strony tekstu (patrz przykład 4.29).

`\parbox` jest poleceniem, którego argumentem jest tekst umieszczany w pudełku i w związku z tym nie wszystkie instrukcje i środowiska mogą być tam używane.

4.7. Pakiet `listings`

Pakiet `listings` pozwala na umieszczenie wewnątrz dokumentu składanego LaTeX-em kodu programu napisanego w jednym z języków programowania. Możliwe jest wstawienie małego fragmentu kodu wewnątrz bieżącego akapitu, całego segmentu kodu jako oddzielny akapit a także można wstawić kod umieszczony w pliku zewnętrznym. Pakiet wczytujemy w preambule znanym już poleceniem `\usepackage{listings}`.

Identyfikacja języka następuje poprzez umieszczenie przed wstawieniem kodu informacji `\lstset{language=[dialekt]nazwa jezyka}`.

Fragment kodu wewnątrz akapitu wstawia instrukcja `\lstinline!fragment kodu!`. Znaki ! mogą zostać zastąpione dowolnymi innymi znakami, pod warunkiem, że nie występują one w kodzie, zawsze jednak należy użyć dwóch jednakowych znaków na początku i na końcu kodu.

¶ Przykład 4.42.

Teraz instrukcją `\lstinline!fragment kodu!` wstawimy wewnątrz tekstu maleńki fragment kodu w języku `Pascal`: **for** i:= **maxint to** 0 **do**.

□

Duże fragmenty kodu umieszczamy w środowisku `lstlisting`.

Zawsze musimy określić język, którego kod będziemy wstawiać do dokumentu. Jeśli w dalszej części dokumentu będziemy chcieli wstawić kod innego języka, wtedy przed wywołaniem środowiska `lstlisting` należy podać jego nazwę poleceniem `\lstset`.

¶ Przykład 4.43.

```
program srednia2;
var a, suma: Real;
i: Integer;
begin
 suma := 0.0;
 for i :=1 to 20 do
 begin
 Read(a);
 suma := suma + a;
 end;
 Writeln((suma / 20.0):0:3);
end.
```

□

Kod (lub jego część) można wczytać także bezpośrednio z pliku stosując polecenie `\lstinputlisting{scieżka do pliku}`. Jeśli dodamy opcjonalny argument `lastline=<liczba>` będziemy mogli wstawić tylko kilka linii kodu.
`\lstinputlisting[lastline=4]{próby.tex}` wstawi pierwsze cztery linie pliku tex-owego:

¶ Przykład 4.44.

```
\documentclass[12pt]{book1}
\usepackage{geometry}
\geometry{paperheight=240mm, paperwidth=168mm, %
textheight=185mm, textwidth=120mm}
\usepackage[cp1250]{inputenc}
\usepackage{polski}
\usepackage{amssymb, amsmath}
```

□

Lista dostępnych języków programowania jest bardzo długa, można się z nią zapoznać zaglądając do dokumentacji pakietu [4], w najnowszej wersji pakietu dodane zostały języki `Lua` i nowy `Fortran` (2003, 2008). W tabeli poniżej umieściliśmy nazwy niektórych z dostępnych języków.

C++ (ANSI, GNU, ISO, Visual)	Delphi	Gnuplot
Lua (5.0, 5.1, 5.2)	Lisp (empty, Auto)	HTML
Mathematica (1.0, 3.0,5.2)	Java (empty, AspectJ)	Logo
Pascal (Borland6, Standard, XSC)	Perl	PHP
XML	Python	Ruby

Tabela 4.3: Wybrane języki programowania dostępne w pakiecie `listings`

Pakiet `listings` posiada wiele opcji, które wprowadzamy instrukcją `\lstset`. Poniżej w tabeli zebraliśmy najczęściej używane.

`showspace=true`	wszystkie spacje są zaznczane
`showtabs=true`	zaznaczane są tabulatory
`extendedchars=true`	pozwala umieszczać polskie litery wewnątrz kodu
`numbers=left`	umieszcza numery linii po lewej stronie kodu
`numberstyle=\tiny`	określa wielkość fontu numerującego linie
`stepnumber=2`	określa częstotliwość numerowania linii – tu co druga
`numbersep=5pt`	ustawia odległość numeru od początku linii
`firstline=4`	wstawia do dokumentu kod od 4 linii
`lastline=8`	wstawia do dokumentu tylko 8 pierwszych linii kodu

Przykłady

1. XML

```
<?xml version="1.0" encoding="iso-8859-2" ?>
- <zwierzaki>
  <pies>Brytan</pies>
  <kot>Feliks</kot>
  <królik>Kicek</królik>
  </zwierzaki>
```

```
 \lstset{language=XML}
\begin{lstlisting}
<?xml version="1.0" encoding="iso-8859-2" ?>
- <zwierzaki>
  <pies>Brytan</pies>
```

```
  <kot>Feliks</kot>
  <królik>Kicek</królik>
  </zwierzaki>
\end{lstlisting}
```

2. Ruby

```
# Ruby knows what you
# mean, even if you
# want to do math on
# an entire Array
cities  = %w[ London
              Oslo
              Paris
              Amsterdam
              Berlin ]
visited = %w[Berlin Oslo]

puts "I␣still␣need␣" +
     "to␣visit␣the␣" +
     "following␣cities:",
     cities - visited
```

3. Python

```
class koszyk
    def _init_(self):
        self.koszyk=[]
    def dodaj (self,obiekt):
        self.koszyk.append(obiekt)
    def rozmiar(self):
        return len(self.koszyk)
```

```
s=koszyk()
s.dodaj("pierwszy wpis")
s.dodaj("drugi wpis")
print s.rozmiar()
del s
```

Opcja showspaces=true

```
# Ruby knows what you
# mean, even if you
# want to do math on
# an entire Array
cities  = %w [London
              Oslo
              Paris
              Amsterdam
              Berlin ]
visited = %w[Berlin Oslo]
```

Opcje showtabs=true i showspaces=false

```
cities  = %w [London
        ,       Oslo
        ,       Paris
        ,       Amsterdam
        ,       Berlin ]
visited = %w[Berlin Oslo]
```

5. Tabele

5.1. Ustawienia tabulatora do pisania w kolumnach

Do umieszczania danych w prostych tabelach możemy wykorzystać środowisko `tabbing`. Pozwala ono ustawiać tabulator w z góry zaplanowanych miejscach na stronie. Szerokość kolumn tabeli ustawiamy w pierwszej linii, przechodząc do następnej kolumny za pomocą znaku \=. W kolejnych liniach przechodzimy do następnej kolumny znakiem \>. Każdą linię kończymy znakiem \\.

Możemy również podać wielkość odstępu między wierszami tabeli, umieszczając po znaku złamania wiersza w nawiasach kwadratowych stosowną liczbę. Zazwyczaj podaje się ułamek lub wielokrotność wysokości litery x bieżącego fontu. W pierwszym przykładzie odstęp między pierwszym wierszem i następnymi będzie równy 0.8 ex, czyli mniej niż wysokość litery x. W następnym będzie równy 3 ex, czyli trzykrotności wysokości litery x.

```
\begin{tabbing}
Rodzaj obuwia\qquad\=Jakość\quad\=Kolor\qquad\=Cena\\[0.8ex]
Czółenka \> średnia \> brązowy \> niska \\
Sandały \> dobra \> biały \> wysoka
\end{tabbing}
```

¶ Przykład 5.1.

Rodzaj obuwia	Jakość	Kolor	Cena
Czółenka	średnia	brązowy	niska
Sandały	dobra	biały	wysoka

□

W następnej tabeli odstęp między pierwszym wierszem i następnymi będzie równy 3ex.

¶ Przykład 5.2.

Rodzaj obuwia	Jakość	Kolor	Cena
Czółenka	średnia	brązowy	niska
Sandały	dobra	biały	wysoka
Pantofle	kiepska	granatowy	niska

□

```
\begin{tabbing}
Rodzaj obuwia\qquad\=Jakość\quad\=Kolor\quad\=Cena\\[3ex]
Czółenka \> średnia \> brązowy \> niska \\
Sandały \> dobra \> biały \> wysoka \\
Pantofle \> kiepska \> granatowy \> niska
\end{tabbing}
```

W przykładzie 5.2 wyraz określający kolor obuwia był zbyt długi i nie zmieścił się w zarezerwowanym dla niego miejscu. W środowisku `tabbing` TEX rezerwuje w następnych liniach tyle miejsca ile zajmuje wyraz w pierwszej linii, a w tym przykładzie, inaczej niż poprzednio, po wyrazie Kolor wstawiliśmy pojedynczy odstęp `\quad`.

Jeżeli nie możemy z góry przewidzieć długości wyrazów, które będziemy wpisywać w kolejnych wierszach tabeli, należy posłużyć się tzw. linią wzor-

ca. Zaraz po deklaracji otwierającej środowisko umieszczamy linię, w której podajemy szerokości kolumn. Możemy to zrobić w dwojaki sposób: za pomocą polecenia `\hspace` lub napisać najdłuższy wyraz, który może wystąpić w naszej tabeli. Przykłady poniżej pokazują oba te przypadki. Forma polecenia `\hspace*{}` na początku linii jest konieczna, by pierwsza kolumna nie została zredukowana do szerokości 0 cm.

¶ Przykład 5.3.

	Jabłka		średnie
Gruszki	luksusowe		
		Śliwki	niedojrzałe

□

```
\begin{tabbing}
\hspace*{2.5cm}\= \hspace{2.5cm} \=\hspace{4cm}\= \kill
\> Jabłka \> \> średnie \\
Gruszki \> luksusowe \>  \\
\> \> Śliwki \> niedojrzałe
\end{tabbing}
```

```
\begin{tabbing}
\hspace*{3cm}\=przykladowa kolumna \=\hspace{4cm} \= \kill
\>Jabłka \> Gruszki \\
\+ Dobre \> Niedobre  \\
Owoce \> Smaczne
\end{tabbing}
```

¶ Przykład 5.4.

	Jabłka	Gruszki
Dobre	Niedobre	
	Owoce	Smaczne

□

W przykładach 5.3 i 5.4 znalazły się jeszcze dwie niewyjaśnione instrukcje: `\kill` i znak `\+`. Instrukcja `\kill` umieszczona na końcu linii wzorca nie dopuszcza do jej wydrukowania. Umieszczenie na początku linii znaku `\>` powoduje umieszczenie wprowadzanego tekstu w drugiej kolumnie. Aby podobny efekt otrzymać w następnym wierszu należy też rozpocząć go od znaku `\>`. Jeśli w kilku kolejnych liniach chcemy zaczynać wprowadzać dane od np. drugiej kolumny (tzn. chcemy przesunąć lewy margines tabeli w prawo o jeden skok tabulatora), to w linii poprzedzającej można też umieścić znak `\+` (polecenie zaczyna działać od następnej linii). W jednej linii można umieścić tyle znaków `\+` ile jest zdefiniowanych skoków tabulatora.

Aby w pojedynczej linii zlikwidować efekt instrukcji `\+` należy użyć instrukcji `\<`.

¶ Przykład 5.5.

Owce	Barany			
		Słonie	Jaki	
			Małpy	Osły
	Krowy	Konie	Lwy	
			Woły	Kozy
		Psy	Koty	Kury

□

```
\begin{tabbing}
\hspace*{2.5cm}\=kolumna wzorca \=\hspace{2.5cm}
 \=\hspace{2cm}\= \kill
Owce \> Barany \+\+\\
\+ Słonie \> Jaki  \\
Małpy\> Osły\\
\< \< Krowy \> Konie\> Lwy\\
Woły\> Kozy\+\\
\< \< Psy\>Koty\>Kury
```

W przykładzie 5.5 na końcu pierwszej linii umieściliśmy dwukrotnie instrukcję \+ i tekst w następnym wierszu tabeli przesunął się o dwie kolumny w prawo.

Instrukcją działającą w sposób przeciwny do \+ jest instrukcja \−. Zastosowanie jej przesuwa margines tabeli o jeden skok tabulatora w lewo. Oczywiście nie można przesunąć marginesu poza zerowy skok tabulatora.

Ilość skoków tabulatora czyli ilość kolumn w tabeli możemy w każdej chwili zmienić (dodać lub usunąć kolumnę) instrukcją \=. Kolumna zostanie dodana, jeśli ilość znaków \> będzie wystarczająca do przejścia do ostatniego skoku tabulatora, w przeciwnym razie instrukcja ta usunie następny skok. Ilustrują to następne przykłady.

```
\begin{tabbing}
Pierwsza kolumna \= Druga kolumna  \\
Pierwsza kolumna \> Dodatkowa kolumna \= Druga kolumna\\
Pierwsza kolumna \> Dodatkowa kolumna \> Druga kolumna
\= Ekstra kolumna\\
Pierwsza kolumna \> Druga kolumna
\end{tabbing}
```

¶ Przykład 5.6.

Pierwszy wiersz określa ilość kolumn na początku tabeli. W drugim wierszu dodamy pomiędzy nimi dodatkową kolumnę. W trzecim dodamy jeszcze jedną kolumnę po prawej stronie. I w czwartym wierszu powrócimy do dwóch kolumn.

Pierwsza kolumna Druga kolumna
Pierwsza kolumna Dodatkowa kolumna Druga kolumna
Pierwsza kolumna Dodatkowa kolumna Druga kolumnaEkstra kolumna
Pierwsza kolumna Druga kolumna

□

Niekiedy chcemy tylko na chwilę (w jednej linii) zlikwidować jeden ze stopów tabulatora. Możemy wtedy użyć instrukcji `\pushtabs` która usuwa bieżący stop, pozostawiając go w pamięci. Wrócić do poprzednich ustawień można za pomocą instrukcji `\poptabs`. Obie one mogą występować dowolną ilość razy ale w jednym środowisku `tabbing` musi ich być tyle samo.

Tekst w wierszu tabeli wyrównujemy do lewego marginesu instrukcją `\'`, a do prawego instrukcją `\ `. W tym wierszu nie może być żadnego ze znaków `\>` i `\=`.

Uwaga: Środowiska `tabbing` nie możemy zagnieżdżać.

Inne przykłady zastosowania środowiska `tabbing`

Poniżej umieściliśmy przykłady ilustrujące większość omówionych własności środowiska `tabbing`.

¶ Przykład 5.7.

Ala ma kota Kota ma Ola One mają koty Iza ma psa
Ala ma Kota ma Ala ma kota i psa Iza ma
Ala ma kota Kota ma Ola One mają koty Iza ma psa

Widać, że zaprojektowano zbyt wąskie kolumny. □

```
\begin{tabbing}
\hspace*{3cm}\= \hspace*{1.5cm}\= \hspace*{3.5cm}\= \kill
Ala ma kota\>Kota ma Ola\>One mają koty\>Iza ma psa\\
Ala ma \> Kota ma\> Ala ma kota i psa\> Iza ma \\
Ala ma kota \>Kota ma Ola\>One mają koty\>Iza ma psa
\end{tabbing}
```

Aby uniknąć nasuwania się treści z sąsiadujących komórek, drugą kolumnę poszerzyliśmy do 2.5 cm.

```
\begin{tabbing}
\hspace*{3cm}\= \hspace*{2.5cm}\= \hspace*{3cm}\= \kill
Ala ma kota\>Kota ma Ola\>Ala i Ola mają kota\>Iza ma psa\\
```

¶ Przykład 5.8.

Ala ma kota Kota ma Ola Ala i Ola mają kota Iza ma psa
Ala ma Kota ma Ala ma i Ola Iza ma
Ala ma kota Kota ma Ola One mają koty Iza ma psa

□

W następnym przykładzie zastosujemy instrukcję \+ w celu przesunięcia zawartości drugiego wiersza o jedną kolumnę w prawo, potem przesuniemy w ten sam sposób zawartość trzeciego wiersza o jedną kolumnę w prawo, a następnie powrócimy do sytuacji z pierwszego wiersza umieszczając w trzecim dwukrotnie instrukcję \-.

¶ Przykład 5.9.

Ala ma kota Kota ma Ola Ala ma kota i Ola ma Iza ma psa
  Ala ma Kota ma Ala ma kota
    Ala ma kota Kota ma Ola
Ala Ela Ola Ula

□

```
\begin{tabbing}
\hspace*{2.5cm}\=\hspace*{2.5cm}\=\hspace*{3.5cm}\=\kill
Ala ma kota\>Kota ma Ola\>Ala ma kota i Ola ma\>
Iza ma psa\+\\Ala ma\>Kota ma\>Ala ma kota\+\\
Ala ma kota\>Kota ma Ola\-\-\\ Ala\>Ela\>Ola\>Ula
\end{tabbing}
```

Kolejny przykład zilustruje zastosowanie instrukcji \pushtabs i \poptabs.

```
\begin{tabbing}
\hspace*{2.5cm}\=\hspace*{2.5cm}\=\hspace*{4cm}\=\kill
Ala ma kota\>kota ma Ola\> Ala i Ola mają koty\>
Iza ma psa\\
Ala ma\> kota ma\> Ala ma kota i kota\>Iza ma\\
```

```
\pushtabs
(zobacz również: \= Iza \\ \> Ela\\ \> Ula )\\
\poptabs
Ala ma kota\>kota ma Ola\>Ala i Ola maja koty\>
Iza ma psa
\end{tabbing}
```

¶ Przykład 5.10.

Ala ma kota	kota ma Ola	Ala i Ola maja koty	Iza ma psa
Ala ma	kota ma	Ala ma kota i kota	Iza ma
(zobacz również: Iza			
	Ela		
	Ula)		
Ala ma kota	kota ma Ola	Ala i Ola maja koty	Iza ma psa

□

5.2. Składanie tabel

Aby złożyć zwykłą tabelkę z danymi możemy skorzystać z następujących środowisk:

```
\begin{array}[pozycja]{opis kolumny}  rzędy \end{array}
\begin{tabular}}[pozycja]{opis kolumny} rzędy \end{tabular}
\begin{tabular*}}{szerokość}[pozycja]{opis kolumny} rzędy
\end{tabular*}
```

Należy zauważyć, że środowisko `array` może być stosowane tylko w trybie matematycznym. Wszystkie te środowiska tworzą w dokumencie ministronę (pudełko). `tabular` tworzy tabelę o szerokości zależnej od treści umieszczanych w komórkach, natomiast `tabular*` dysponuje argumentem umożliwiającym ustalenie jej szerokości. Parametr `pozycja` określa pionowe położenie tabeli i może przyjmować następujące wartości:

- `t` – górny wiersz tabeli leży na linii bazowej
- `b` – dolny wiersz tabeli leży na linii bazowej

Parametr `opis kolumny` odpowiada za formatowanie tekstu w poszczególnych komórkach i jest obowiązkowy. Podstawowe jego wartości to `l,r,c` określające poziome położenie tekstu w komórce tabeli. Inne to na przykład parametr `p{2cm}`, który tworzy w komórce pudełko o szerokości 2 cm i łamie wstawiany tekst na wiersze o tej długości. Ten parametr jest równoważny wstawieniu pudełka poleceniem `\parbox[t]{2cm}{tekst}`. Inny parametr `*{3}{|r|}`, równoważny parametrowi `|r|r|r|`, tworzy w tabeli trzy kolumny oddzielone pionowymi liniami, w których tekst jest wyrównany do prawej strony komórki. Parametr `@{tekst}` wstawia we wszystkich wierszach tabeli w danej kolumnie `tekst`. Przykłady 5.13 i 5.14 pokazują zastosowanie omówionych parametrów.

Korzystanie z wymienionych środowisk przy tworzeniu tabel nie wymaga wczytywania dodatkowych pakietów, jednak uzyskane tabele nie zawsze spełniają nasze wymagania. Aby uzyskać dostęp do poleceń, które pozwolą na składanie bardziej wyrafinowanych pod względem wyglądu tabel, należy wczytać w preambule dokumentu pakiet `array` i pakiet `tabularx`. Wybrane narzędzia z tych pakietów opiszemy w dalszym ciągu tego rozdziału.

Tabelę utworzoną przez `tabular` możemy umieścić w dokumencie w dwojaki sposób: bezpośrednio w miejscu utworzenia lub umieszczając ją w środowisku `table` które jest jednym z tzw. środowisk pływających. Filozofię umieszczania obiektów pływających na stronie opisaliśmy w rozdziale 5.3. Środowisko `table` umożliwia wstawienie podpisu pod lub nad tabelą i, jeśli zaopatrzymy ją w etykietę, odwołanie się do niej w dalszej części dokumentu. Wszystkie tabele umieszczone w tym środowisku znajdą się również w wygenerowanym poleceniem `\listoftables` spisie tabel.

Każda tabela może być umieszczona pośrodku kolumny tekstu, czyli w środowisku `center`.

Prostą tabelę pokazuje przykład którego kod podajemy niżej.

```
\begin{tabular}{rlcc}
Miejsce & Drużyna &  Gole & Punkty\\[0.2ex]
```

```
1 & Legia & 12 & 36\\
2 & Górnik & 10 & 30\\
3 & Widzew & 8 & 24\\
4 & ŁKS&4&14\\
\end{tabular}
```

¶ Przykład 5.11.

Miejsce	Drużyna	Gole	Punkty
1	Legia	12	36
2	Górnik	10	30
3	Widzew	8	24
4	ŁKS	4	14

□

Bezpośrednio po zakończeniu pierwszego wiersza instrukcją \\ dodaliśmy niewielki odstęp, wstawiając w nawiasach kwadratowych jego wartość (tu 0.2 ex). Takie rozsunięcie wierszy w pionie zwiększa czytelność tabeli szczególnie wtedy, gdy nie oddzielamy poszczególnych wierszy poziomymi liniami.

Często tworzona przez nas tabela musi być podzielona na pojedyncze komórki liniami poziomymi i pionowymi. Pionowe linie uzyskuje się przez wstawienie znaku | (z klawiatury) pomiędzy parametry określające położenie tekstu, a poziome przez umieszczenie na początku każdego rzędu polecenia \hline.

```
\begin{center}
\begin{tabular}{|r|l|c|c|}
\hline
Miejsce & Drużyna &  Gole & Punkty \\[0.2ex]
\hline 1 & Legia & 12 & 36 \\\hline
2 & Górnik & 10 & 30        \\\hline
3 & Widzew & 8 & 24\\\hline
4 & ŁKS&4&14\\\hline
```

```
\end{tabular}
\end{center}
```

¶ Przykład 5.12.

Miejsce	Drużyna	Gole	Punkty
1	Legia	12	36
2	Górnik	10	30
3	Widzew	8	24
4	ŁKS	4	14

□

Nasza tabelka została umieszczona w środowisku `center`.

Poszczególne rzędy tabeli można także oddzielić podwójną linią poziomą lub pionową. Wstawiamy wtedy w każdym wierszu dwa polecenia `\hline`, a pomiędzy parametry określające położenie tekstu w komórkach dwie instrukcje `|`. Nasza tabela będzie miała podwójne linie poziome i pionowe, ale nie uzyskamy ich ciągłego połączenia. Jeśli zależy nam na podwójnych liniach powinniśmy poszukać rozwiązania w literaturze.

Następne przykłady ilustrują zastosowanie parametrów `p{2cm}` oraz `*{4}{|c|}`. Druga z tabel jest położona centralnie, umieszczona w środowisku `table`, z podpisem i etykietą.

```
\begin{tabular}{|r|l|c|c|p{2.4cm}|}\hline
Miejsce&Drużyna&Gole&Punkty&Sponsorzy wszystkich drużyn\\
\hline 1 & Legia & 12 & 36 &PKO BP\\\hline
```

```
\begin{table}[h!]
\begin{center}
\begin{tabular}{*{4}{|c|}}
\hline
Miejsce & Drużyna &  Gole & Punkty\\
\hline 1 & Legia & 12 & 36 \\\hline
```

```
\end{tabular}
\caption{Przykładowa tabelka z podpisem \label{nn}}
\end{center}
\end{table}
```

¶ Przykład 5.13.

Miejsce	Drużyna	Gole	Punkty	Sponsorzy wszystkich drużyn
1	Legia	12	36	PKO BP
2	Górnik	10	30	PKO SA
3	Widzew	8	24	MultiBank
4	ŁKS	7	21	Kredyt Bank

□

Miejsce	Drużyna	Gole	Punkty
1	Legia	12	36
2	Górnik	10	30
3	Widzew	8	24
4	ŁKS	7	21

Tabela 5.1: Przykładowa tabelka z podpisem

Scalenie komórek tabeli utworzonej w środowisku `tabular` uzyskamy za pomocą polecenia `\multicolumn{}{}{}`. W pierwszym nawiasie klamrowym umieszczamy liczbę określającą ilość scalanych komórek, w drugim podajemy sposób położenia zawartości w nowej komórce (np. `l,r,c,p`), w trzecim umieszczamy zawartość komórki. Polecenie `\multicolumn` musi znajdować się w pierwszej komórce wiersza lub bezpośrednio po znaku `&`, tak jak w przykładzie 5.14.

Jeżeli ilość scalanych kolumn jest równa 1, to polecenie to służy do chwilowego unieważnienia zadanego globalnie pozycjonowania tekstu w danej komórce (patrz przykład 5.15).

¶ Przykład 5.14.

Liczba szkół wyższych w Polsce po roku 1989		
1992/1993	124	18
2000/2001	310	195
2010/2011	470	338

□

```
\begin{tabular}{|l|l|r|}\hline
\multicolumn{3}{|c|}{Liczba szkół wyższych w Polsce
po roku 1989}\\\hline
1992/1993 & 124 & 18\\\hline
2000/2001 & 310 & 195\\\hline
2010/2011 & 470 & 338 \\\hline
\end{tabular}
```

```
\begin{tabular}{|l|r|r|}\hline
Alicja & Aleksandra & Eleonora\\\hline
421& 5& 88\\\hline
7 &\multicolumn{1}{l|}{12}&777\\\hline
\end{tabular}
```

¶ Przykład 5.15.

Alicja	Aleksandra	Eleonora
421	5	88
7	12	777

□

Środowisko `tabular` umożliwia również rysowanie linii poziomych i pionowych tylko dla wybranej liczby komórek. Linię poziomą od komórki w kolumnie n do komórki w kolumnie m otrzymamy poleceniem `\cline{n-m}`

wstawionym w odpowiednim wierszu tabeli. Pionową linię w jednej komórce rysuje polecenie `\vline` patrz przykłady 5.16 i 5.17.

¶ Przykład 5.16.

Sieć	Grupa	Symbol
Ukośnokątna	A	aa
	B	bb

□

```
\begin{tabular}{|c|c|c|}
\hline
Sieć&Grupa&Symbol\\\hline
Ukośnokątna&A&aa\\
\cline{2-3}&B&bb\\\hline
\end{tabular}
```

¶ Przykład 5.17.

Sieć	Grupa	Symbol	
Ukośnokątna	A	aa	aa1
	B	bb	

□

```
\begin{tabular}{|c|c|c|}
\hline
Sieć&Grupa&Symbol\\\hline
Ukośnokątna&A&aa\vline aa1\\
\cline{2-3}&B&bb\\\hline
\end{tabular}
```

W obu przykładach powyżej lepiej byłoby, żeby tekst „ukośnokątna" był wyśrodkowany w pionie. Należy więc przesunąć go w komórce poleceniem `\raisebox{arg1}{arg2}` (patrz strona 115). Pierwszy argument jest liczbą z mianem, np. 2ex, 0.5cm czy 4mm, drugi jest tekstem który ma być podniesiony (gdy pierwszy argument jest dodatni) lub opuszczony (gdy pierwszy

argument jest ujemny) w stosunku do linii bazowej. Następne przykłady pokazują dwa sposoby umieszczenia naszego tekstu w scalonej komórce.

¶ Przykład 5.18.

Sieć	Grupa	Symbol	
Ukośnokątna	A	aa	aa1
	B	bb	

□

```
\begin{tabular}{|c|c|c|}
\hline
Sieć&Grupa&Symbol\\\hline
\raisebox{-2.5ex}{Ukośnokątna}&A&aa\vline aa1\\
\cline{2-3}&B&bb\\\hline
\end{tabular}
```

¶ Przykład 5.19.

Sieć	Grupa	Symbol	
Ukośnokątna	A	aa	aa1
	B	bb	

□

```
\begin{tabular}{|c|c|c|}
\hline
Sieć&Grupa&Symbol\\\hline
 &A&aa\vline aa1\\
\cline{2-3}\raisebox{2.5ex}{Ukośnokątna}&B&bb\\\hline
\end{tabular}
```

Więcej informacji o tworzeniu tabel czytelnik znajdzie w literaturze.

5.3. Wstawianie rysunków i tabel

Jeżeli rysunki i tabele umieścimy w środowiskach `figure` i `table` odpowiednio, są one traktowane jako tzw. obiekty pływające czyli takie, których miej-

sce na stronie zależy od tego, gdzie w pliku tekstowym dany obiekt został umieszczony.

Zwykle autor umieszcza rysunek lub tabelę tam, gdzie po raz pierwszy jest o nim mowa. Po skompilowaniu dokumentu może się okazać, że obiekt został umieszczony gdzie indziej. Dzieje się tak dlatego, że algorytm którego LaTeX używa do rozmieszczania obiektów pływających ma pewne z góry zadane parametry które opiszemy w dalszym ciągu tego rozdziału.

Algorytm wstawiania obiektów pływających jest tak skonstruowany, że każdy rodzaj jest umieszczany po kolei tzn. tabela 1, tabela 2, ..., rysunek 1, rysunek 2, ... nawet wtedy, gdy miejsce ich pojawienia jest różne od tego w pliku tekstowym. Jeżeli w dokumencie pojawiają się rysunki i tabele, to mogą one się przeplatać, ale zawsze w dobrej kolejności.

Jeżeli dokument ma zawierać dużą ilość tabel i/lub rysunków autor powinien zdecydować, czy mają być one umieszczane wewnątrz tekstu czy lepiej umieścić je na osobnych stronach. Dobrze jest wtedy skorzystać z któregoś pakietu dostarczającego narzędzi do rozmieszczania na stronie obiektów pływających.

5.3.1. Parametry dla środowisk `table` i `figure`

LaTeX „wie" o dwóch miejscach w kolumnie tekstu, w których powinien umieszczać obiekty typu rysunek lub tabela. Są to góra (top) i dół (bottom) strony. Możemy także umieszczać obiekty w dowolnym miejscu w kolumnie tekstu (inline) ale należy wtedy dodać jeden z parametrów (w nawiasach []) których listę podajemy w tabeli 5.2.

Jako parametry środowiska można umieścić wszystkie te znaki. Ich kolejność nie wpływa na działanie algorytmu rozmieszczającego obiekty pływające na stronie. Jeśli któraś z wymienionych liter nie jest obecna, wtedy algorytm nie próbuje umieścić obiektu w odpowiadającym jej miejscu.

Domyślnie algorytm rozmieszcza na stronie jeden obiekt u góry (top) i jeden u dołu (bottom), znak ! znosi te ograniczenia.

Ilość obiektów pływających na stronie kontrolują cztery liczniki wymienione w tabeli 5.3. Ich wartości zmienia się poleceniem `\setcounter`.

Jeśli chcemy umieścić na stronie np. 4 małe rysunki wówczas powinniśmy ustalić wartość parametru `totalnumber` następująco

```
\setcounter{totalnumber}{4}
```

! –	informuje kompilator, że niektóre domyślne ograniczenia mogą być zignorowane,
h –	obiekt powinien być umieszczony tam, gdzie pojawił się w tekście,
t –	obiekt powinien być umieszczony u góry strony (top),
b –	obiekt powinien być umieszczony u dołu strony (bottom),
p –	obiekt powinien być umieszczony na oddzielnej stronie.

Tabela 5.2: Dopuszczalne parametry środowiska `table`

`totalnumber` (domyślnie =3)	– maksymalna ilość obiektów na stronie,
`topnumber` (domyślnie =2)	– maksymalna ilość obiektów w górnej części strony,
`bottomnumber` (domyślnie =1)	– maksymalna ilość obiektów w dolnej części strony,
`dbltopnumber` (domyślnie =3)	– maksymalna ilość obiektów na stronie złożonej w dwóch kolumnach.

Tabela 5.3: Liczniki kontrolujące miejsce umieszczenia obiektu

`\topfraction` (domyślnie =0.7)	– maksymalny rozmiar obiektu umieszczonego u góry strony,
`\bottomfraction` (domyślnie =0.3)	– maksymalny obszar zarezerwowany u dołu strony,
`\textfraction` (domyślnie =0.2)	– minimalny obszar zarezerwowany dla tekstu (poza obiektem) który musi pojawić się na stronie,
`\dbltopfraction` (domyślnie =0.2)	– maksymalny obszar zarezerwowany dla tekstu u góry strony przy składzie w dwóch kolumnach.

Tabela 5.4: Parametry kontrolujące położenie obiektu na stronie

Jak już wspomnieliśmy wcześniej LaTeX może umieścić tabelę lub rysunek w jednym z dwóch obszarów na stronie – u góry lub u dołu. Rozmiary tych obszarów są kontrolowane przez parametry przedstawione w tabeli 5.4 a ich wartości możemy zmieniać poleceniem `\renewcommand`. Zanim jednak zdecydujemy się na zmianę wartości parametrów wymienionych w tabeli 5.4 powinniśmy zapoznać się z następującymi informacjami wymienionymi w punktach 1–3.

1. `\textfraction`
   Jeżeli ten parametr będzie miał wartość mniejszą niż 0.15 wówczas czytelność naszego dokument będzie niedostateczna. Jeśli obiekt (tabela, rysunek) zajmuje więcej niż 85% obszaru strony, lepiej przenieść go na osobną stronę. Nie należy nigdy nadawać temu parametrowi wartości 0, gdyż może to zmylić LaTeX-a i spowodować złożenie strony niepoprawnej typograficznie.
2. `\topfraction`
   Nie należy ustalać tego parametru większego niż 1−`\textfraction` gdyż prowadzi to do sprzeczności z algorytmem rozmieszczającym obiekty pływające.

3. `\bottomfraction`

   Na dole strony nigdy nie należy umieszczać dużych obiektów pływających (tabel, rysunków), więc `\bottomfraction` jest zwykle mniejszy niż `\topfraction`. Nie należy zatem ustalać tego parametru większego niż 1−`\topfraction` gdyż prowadzi to, jak w poprzednim przypadku, do sprzeczności z algorytmem rozmieszczającym obiekty pływające.

¶ Przykład 5.20.

Zmiana wartości parametru `\textfraction`

```
\renewcommand{\textfraction}{.3}
```

oznacza, że obiekt (rysunek, tabela) może zajmować nie więcej niż 70% obszaru strony.

□

Odstępy, które oddzielają wstawiany obiekt od następującego po nim tekstu kontrolowane są przez parametry, których wielkość zależy od rozmiaru fontu użytego w dokumencie. Parametry te są wielkościami „rozciągliwymi", to znaczy mogą się zmieniać w ściśle określonych w definicji granicach. Poniżej podamy wielkości domyślne tych parametrów dla fontu 10 punktowego.

- `\floatsep` (domyślnie 12pt plus 2pt minus 2pt) – odstęp pomiędzy obiektami u góry lub u dołu strony,
- `\textfloatsep` (domyślnie 20pt plus 2pt minus 4pt) – odstęp pomiędzy górnym lub dolnym obszarem zarezerwowanym na obiekty pływające a tekstem.

Jeśli obiekt jest umieszczany wewnątrz tekstu na stronie (gdy używamy parametru **`h!`**), odstępy reguluje parametr `\intextsep` (domyślnie 12pt plus 2pt minus 2pt).

Należy pamiętać, że **`h!`** (here) jest parametrem warunkowym i nie zawsze znaczy „tutaj", ale często „tutaj jeśli się zmieści". Jeśli wczytamy w preambule

pakiet `float`, będziemy mogli użyć parametr `[H]` oznaczający „bezwarunkowo tutaj”. Należy jednak pamiętać, że nie może być on użyty z żadnym innym parametrem, nie możemy więc napisać `[Hhtbp]`.

6. Matematyka

6.1. Wyrażenia i wzory matematyczne

LaTeX przełącza się na tryb matematyczny gdy napotka znak dolara ($) lub dwóch dolarów ($$) a powraca do trybu tekstowego po napotkaniu następnego dolara lub dwóch. Zamiast znaku ($$) można też używać znaku \[do otwarcia środowiska matematycznego i \] do jego zamknięcia. Wzór eksponowany możemy także umieścić w środowisku `displaymath`. We wszystkich przypadkach nasz wzór nie będzie opatrzony numerem. Poniższy przykład ilustruje użycie opisanych instrukcji.

¶ Przykład 6.1.

$$\frac{1}{2} x^2+z=12$$

$$\lim_{x\to\infty} f(x)=3$$

$$\int_a^b g(x)\,dx = 5$$

□

```
$$\frac{1}{2} x^2+z=12$$
\begin{displaymath}
\lim_{x\to\infty} f(x)=3
\end{displaymath}
\[ \int_a^b g(x) dx = 5\]
```

Wzory matematyczne składane wewnątrz akapitu (a więc pomiędzy pojedynczymi znakami dolara) są składane w nieco inny sposób niż wzory eksponowane. Wszystkie znaki i formuły, które we wzorze eksponowanym będą umieszczone pod dowolnym operatorem matematycznym (np. całka, suma, granica), w akapicie będą umieszczone obok niego. Dzięki temu możemy zapobiec powstawaniu nadmiernie dużych odstępów międzywierszowych, szczególnie w dokumencie z dużą ilością wzorów matematycznych w tekście.

¶ Przykład 6.2.

To będzie przykład wyrażeń matematycznych $\frac{1}{2}x^2 + z = 12$ umieszczanych w tekście dokumentu $\lim_{x\to\infty} f(x) = 3$, a więc w postaci \$ wzór \$. I jeszcze jeden przykład $\sum_{i=1}^{\infty} a_i f(x) = 0$, dla $x \in [-3, 8)$.

□

Rozmiar czcionki jaką LaTeX wybiera do złożenia różnych partii wzoru może być zmieniony przez użytkownika. W trybie matematycznym dostępne są cztery rozmiary fontów, z których kompilator automatycznie wybiera właściwy dla danego wyrażenia. Są to:

Nazwa	Skrót	Opis
`\displaystyle`	D	rozmiar dla wzorów eksponowanych,
`\textstyle`	T	rozmiar wzorów w tekście,
`\scriptstyle`	S	rozmiar indeksów dolnych i górnych,
`\scriptscriptstyle`	SS	rozmiar indeksów drugiego rzędu.

Tabela 6.1: Rozmiary fontów matematycznych

Skróty podane w tabeli powyżej posłużą nam do wyjaśnienia mechanizmu działania wyboru rozmiaru fontu. Kiedy przełączamy tryb pracy na matematyczny, dla wzorów eksponowanych aktywny staje się rozmiar D natomiast dla wzorów umieszczanych w tekście rozmiar T. Tabela poniżej pokazuje zasady wyboru wielkości fontu.

Rozmiar aktywny	Ułamki		Indeksy
	licznik	mianownik	górny(dolny)
D	T	T	S
T	S	S	S
S	SS	SS	SS
SS	SS	SS	SS

Zawartość tabeli omówimy na przykładzie ułamka. Jeżeli aktywny jest font rozmiaru D (czyli ułamek znajduje się we wzorze eksponowanym), wówczas licznik i mianownik są rozmiaru T. W przypadku ułamka w tekście (czyli font w rozmiarze T) licznik i mianownik są rozmiaru S.

¶ Przykład 6.3.

Ten ułamek jest umieszczony w tekście $\frac{2x+4}{x-1}$, więc licznik i mianownik są złożone wielkością T. Zobaczmy, jak będzie wyglądał ułamek taki $\frac{\frac{1}{2}x+4}{2-y}$. I jeszcze ułamek z indeksami górnym i dolnym $\frac{x_1+y_1}{2x_1y_1^2}$.

□

Wielopiętrowe ułamki mogą być składane w dwojaki sposób. Jeśli pozwolimy LaTeX-owi dobrać wielkość czcionki, to otrzymamy ułamek zajmujący niewiele miejsca na stronie ale mało czytelny.

$$\frac{1}{x_1+\frac{1}{x_2+\frac{1}{x_3+\frac{1}{x_4+x_5}}}}$$

¶ Przykład 6.4.

Bardziej czytelny ułamek otrzymamy powiększając czcionkę w kolejnych ułamkach poleceniem `\displaystyle`.

$$\displaystyle\frac{1}{x_1+\displaystyle\frac{1}{x_2+\displaystyle\frac{1}{x_3+\displaystyle\frac{1}{x_4+x_5}}}}$$

□

```
$$\frac{1}{\displaystyle x_1+\frac{1}{\displaystyle x_2+%
\frac{1}{\displaystyle x_3+\frac{1}%
{\displaystyle x_4+x_5}}}}$$
```

Składając prace matematyczne LaTeX-em należy pamiętać o zapewnieniu sobie dostępu do specjalistycznych znaków i wygodnych w użyciu środowisk do składania twierdzeń czy definicji. Tych ułatwień dostarczają nam wczytane w preambule pakiety `amssymb`, `amsmath` i `amsthm`. W następnych rozdziałach opiszemy dwa z nich.

Mechanizm numerowania wzorów został zastosowany w środowisku `equation`. Jeśli pewnemu wyrażeniu matematycznemu nadamy numer, możemy wówczas skorzystać ze zdefiniowanych w LaTeX-u mechanizmów odsyłaczy (zobacz rozdział 4.4.).

Wszystkie środowiska LaTeX-owe mają taką samą składnię (zmienia się tylko nazwa), rozpoczynają się `\begin{nazwa}` i kończą `\end{nazwa}`. Niektóre z nich mogą mieć dodatkowe argumenty (obowiązkowe – w nawiasach klamrowych, lub opcjonalne – w nawiasach kwadratowych). Sposób użycia środowiska `equation` pokazuje następujący przykład.

¶ Przykład 6.5.

$$\forall x \in \mathbb{R} \quad x^2 + x + 1 \geqslant 0 \tag{6.1}$$

□

```
\begin{equation}
\forall x\in \mathbb R \quad x^2 + x +1 \ge 0
\end{equation}
```

Ze środowiska `equation` możemy korzystać bez wczytywania dodatkowych pakietów, natomiast w pakiecie `amsmath` zostało zdefiniowane środowisko `equation*`, które działa tak samo jak $$ czy środowisko `displaymath`, czyli nie numeruje wzoru.

Uwaga. Użyty w ostatnim przykładzie symbol `\forall` ($\forall$) jest tak zdefiniowany, że nie dopuszcza możliwości podpisania pod nim żadnego wyrażenia. Możemy tę definicję zmienić tak jak poniżej

```
\let\originalforall=\forall%
\renewcommand{\forall}%
{\mathop{\vcenter{\hbox{\Large$\originalforall$}}}}
```

Po takim przedefiniowaniu możemy już napisać `\forall_{x\in X}`.

$$\bigvee_{x\in X} \quad x^2 + x + 1 \geqslant 0 \tag{6.2}$$

Indeksy górny i dolny, pierwiastki

Większość instrukcji składu wzorów matematycznych odnosi się do jednego następującego po niej znaku. Jeśli więc chcemy, by instrukcja zastosowana była do więcej niż jednego znaku, należy je zgrupować czyli umieścić w nawiasach klamrowych.

Indeksy górne i wykładniki potęg wstawiamy instrukcją `^` (np. `$a^2$` a^2), a indeksy dolne instrukcją `_` (np. `$b_3$` b_3). Oba te znaki mogą być użyte tylko w środowisku matematycznym.

Pierwiastek kwadratowy z `wyrażenia` zapisujemy używając instrukcji `\sqrt{wyrażenie}`, natomiast pierwiastek stopnia różnego od dwóch zapiszemy instrukcją `\sqrt[n]{wyrażenie}`. Samodzielny znak pierwiastka generuje instrukcja `\surd`.

¶ Przykład 6.6.

$$\sqrt{\pi R^2 + xyz}$$
$$\sqrt[5]{\pi R^2 + xyz}$$

I tylko sam znak pierwiastka $\surd$.

□

```
$$\sqrt{\pi R^2 + xyz}$$
\begin{equation*}
\sqrt[5]{\pi R^2 + xyz}
\end{equation*}
I znak pierwiastka $\surd$.
```

Podkreślenia i klamry

Matematycy lubią używać różnych sposobów wyróżnień. Najprostsze z nich to podkreślenie – polecenie `\underline{...}`, umieszczenie poziomej kreski nad wyrażeniem – polecenie `\overline{...}` oraz umieszczanie poziomych klamer nad i pod wyrażeniem – polecenia `\overbrace{...}` i `\underbrace{...}`. Zastosowanie tych poleceń pokazuje

¶ Przykład 6.7.

$$\underline{x+y} + \overline{x-y} = 5, \qquad \overbrace{n_1+n_2+\dots n_k} \quad \underbrace{a \cdot a \cdot a \cdot \ldots \cdot a}_{n}$$

□

```
$$\underline{x+y} + \overline{x-y} =5, \qquad
\overbrace{n_1+n_2+ \dots n_k} \quad
\underbrace{a \cdot a \cdot a \cdot \ldots \cdot a}_n$$
```

Nawiasy

Do składu różnych ograniczników np. nawiasów okrągłych, kwadratowych i sześciennych (klamrowych) można używać symboli z klawiatury lub specjalnych poleceń. Nawiasy okrągłe i kwadratowe wpisujemy bezpośrednio z klawiatury, natomiast nawias klamrowy z klawiatury musimy poprzedzić znakiem \ rozpoczynającym każde polecenie czy instrukcję gdyż jest to znak specjalny TEX-a. Tak więc nawias klamrowy otrzymamy pisząc `\{ i \}`. Przykłady wprowadzania ograniczników pokazaliśmy w przykładzie 6.8.

W LaTeX-u znajdziemy również polecenia, które w ustalony sposób powiększają znaki nawiasów. Są to:

```
\big, \Big, \bigg i \Bigg.
```

Po każdym z tych poleceń trzeba umieścić właściwy nawias.

¶ Przykład 6.8.
Poniżej umieściliśmy znaki nawiasów okrągłego i klamrowego, otwierające i zamykające, najpierw wielkości bieżącego fontu a następnie powiększone instrukcjami `\big`, `\Big`, `\bigg` i `\Bigg` odpowiednio.

$$((\big(\big(\Big(\Big(\{ \{ \big\{ \big\{ \Big\{ \Big\{ \quad \Big\} \Big\} \big\} \big\} \} \})) \Big) \Big) \bigg) \bigg)$$

□

Poniżej wyjaśnienie, jak otrzymaliśmy te nawiasy.

```
$(␣\␣\big(␣\␣\Big(␣\␣\bigg(␣\␣\Bigg(␣\␣\{␣\␣\big\{␣\
\Big\{␣\␣\bigg\{␣\␣\Bigg\{␣\quad␣)␣\␣\big)␣\␣\Big)␣\␣\bigg)
\␣\Bigg)␣\␣\}␣\␣\big\}␣\␣\Big\}␣\␣\bigg\}␣\␣\Bigg\}␣$
```

Mamy również do dyspozycji instrukcję powiększającą dowolny nawias do wielkości proporcjonalnej do jego zawartości. Składa się ona z pary poleceń: `\left` i `\right`, po których należy umieścić odpowiedni znak nawiasu. Jeżeli nawias otwierający został umieszczony w jednej linii a nawias zamykający będzie dopiero w następnej, to na końcu pierwszej linii powinniśmy umieścić „pusty” `\right.` nawias zamykający a na początku drugiej „pusty” `\left.` nawias otwierający. W przeciwnym wypadku dostaniemy komunikat o wystąpieniu błędu kompilacji.

Operatory

Operatory sumy, całki i produktu (iloczynu) wprowadza się w następujący sposób:

$$\sum_{n=1}^{\infty}\quad \sum_{k=1}^{100}\quad \int_0^x f(x)dx \quad \int_0^{\frac{n}{2}} g(y)dy \quad \prod_k X_k$$

```
$$\sum_{n=1}^{\infty}\quad
\sum_{k=1}^{100}\quad
\int_0^x f(x)dx \quad
\int_0^{\frac{n}{2}} g(y)dy\quad
\prod_k X_k$$
```

Sumowanie po więcej niż jednym wskaźniku możemy zapisać wybierając jedną z dwóch możliwości. Jedna to skorzystanie ze środowiska `subarray`, a druga to użycie polecenia `\substack`. Środowisko `subarray` daje nam możliwość wyrównania granic sumowania do lewej lub prawej strony. W przypadku polecenia `\substack` wyrażenia określające granice sumowania są w kolejnych liniach centrowane.

$$\sum_{\substack{0<i<n\\0<j<m}} K_{ij}(x) \quad \sum_{\substack{i\in A\\0<j<m}} K_{ij}(x)$$

```
$$\sum_{\substack{0<i<n\\0<j<m}}
 K_{ij}(x)\quad
\sum_{\begin{subarray}{l}
i\in A\\0<j<m\end{subarray}}
K_{ij}(x)$$
```

Inne

Do napisania dowolnych znaków (np. strzałki lub litery) nad innym znakiem (np. nad równością czy znakiem nierówności) możemy użyć polecenia `\stackrel{}{}`, lub użyć instrukcji `\overset{}{}` i `\underset{}{}`. Pierwsza umieszcza wyrażenie z pierwszego nawiasu klamrowego nad tym z drugiego nawiasu, a druga wyrażenie z pierwszego nawiasu klamrowego pod tym z drugiego.

¶ Przykład 6.9.

$$\binom{n}{k} C_n^k \qquad x \stackrel{!}{=} y, \quad y \stackrel{?}{>} 6, \quad x_n \stackrel{pw}{\longrightarrow} \infty, \quad \stackrel{p.w.}{\equiv}, \quad a_n \underset{n\to\infty}{\longrightarrow} b.$$

□

```
$$\binom{n}{k} C_n^k\qquad x\stackrel{!}{=} y,
\quad y\stackrel{?}{>} 6,
\quad x_n\stackrel{pw}{\longrightarrow } \infty,
\quad \overset{p.w.}{\equiv},
\quad a_n\underset{n\to\infty}{\longrightarrow }b.$$
```

Dla oznaczenia zbioru liczb rzeczywistych czy całkowitych używamy najczęściej liter postaci $\mathbb{R}$ lub $\mathbb{C}$. Otrzymujemy je instrukcję `$\mathbb R$`, która dostępna jest po wczytaniu pakietu **amsmath**.

Wielokropek w trybie tekstowym to trzy kropki na linii tekstu. Generuje je instrukcja `\dots`. Wielokropek wstawiony tą instrukcją różni się od trzech kropek z klawiatury odstępami pomiędzy kropkami. W trybie matematycznym są dwa rodzaje wielokropka `\ldots` (kropki na poziomie linii bazowej tekstu) i `\cdots` (kropki umieszczone w połowie wysokości małej litery tekstu). Można też używać `\dots` które jest równoważne `\ldots`.

¶ Przykład 6.10.

Najpierw przykład wielokropka w trybie tekstowym generowanego instrukcją `\dots` ..., a teraz trzy kropki z klawiatury ... . Przechodzimy do trybu matematycznego – najpierw kropki na dole, czyli `\ldots`, a potem pośrodku, czyli `\cdots`

$$a_1, a_2, \ldots, \quad x_1 \cdot x_2 \cdots x_n.$$

□

```
$$a_1, a_2,\ldots,\quad  x_1\cdot x_2 \cdots x_n.$$
```

Macierze i wyznaczniki

Do składu macierzy, wyznaczników a także do wyrównywania tekstu w kolumnach spisów tabelarycznych można użyć zdefiniowanego w LaTeX-u środowiska `array` o następującej składni:

```
\begin{array}[pozycja]{kolumny} kolejne  rzędy \end{array}
```

gdzie parametr `pozycja` przyjmuje wartości opisane na stronie 115, zaś parametr `kolumny` opisany jest na stronie 133. Ich ilość musi być równa ilości kolumn macierzy.

¶ Przykład 6.11.

$$\begin{array}{cccccc}
a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\
a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\
a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \\
a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46}
\end{array}$$

□

```
$$
\begin{array}{cc}
 a_{11} &  a_{12} \\ a_{21} &  a_{22} \\
 a_{31} &  a_{32}\\a_{41} &  a_{42}
\end{array}
$$
```

Każdy element macierzy jest oddzielony od następnego znakiem `&`, koniec wiersza zaznaczony jest znakiem `\\`. Jeżeli elementy macierzy mają być umieszczone pomiędzy nawiasami, wtedy przed otwarciem środowiska umieszczamy `\left(` (nawiasy okrągłe), a po jego zamknięciu `\right)`. Zamiast nawiasów okrągłych można wstawić inne nawiasy lub proste kreski.

Należy zwrócić uwagę na to, że środowisko `array` może być użyte tylko w trybie matematycznym.

¶ Przykład 6.12.

$$X = \begin{pmatrix} a_{11} & a_{12} \\ b_{21} & b_{22} \end{pmatrix} \qquad X = \begin{bmatrix} a_{11} & a_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

$$X = \begin{Bmatrix} a_{11} & a_{12} \\ b_{21} & b_{22} \end{Bmatrix} \qquad X = \begin{vmatrix} a_{11} & a_{12} \\ b_{21} & b_{22} \end{vmatrix}$$

□

```
$$
X=\left\{ \begin{array}{cc}
a_{11}&a_{12}\\
b_{21}&b_{22}
\end{array} \right\}$$
```

Pakiet **amsmath** który omawiamy nieco dalej oferuje dużo wygodniejszy mechanizm składania wyznaczników i macierzy.

Wzory wieloliniowe

Do składania wzorów w kilku wierszach LaTeX udostępnia nam środowisko **eqnarray**. Używa się go w sposób następujący:

$$\begin{aligned} \sin x &= x - \frac{x^3}{3!} + \frac{x^5}{5!} - && (6.3) \\ &\quad -\frac{x^7}{7!} + \cdots && (6.4) \end{aligned}$$

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

W środowisku **eqnarray** każdy wiersz jest opatrywany kolejnym numerem. Aby nadać numer tylko jednemu z nich należy przed złamaniem każdego

z pozostałych wierszy umieścić instrukcję `\nonumber`. Omawiane środowisko, oparte na środowisku **array**, nie jest jednak w stanie sprostać wymaganiom wielu matematyków, dlatego też w pakiecie **amsmath** zdefiniowano kilka innych środowisk.

Aby nie numerować żadnego wiersza należy użyć środowiska **eqnarray***.

Symbole Newtona

Tak samo jak w przypadku ułamków symbole Newtona można pisać w dwóch rozmiarach, bez względu na to gdzie się znajdują.

¶ Przykład 6.13.

Duży symbol Newtona w tekście $\dbinom{a}{b}$ umieszczamy poleceniem `\dbinom{}{}`, natomiast mały $\binom{a}{b}$ w tekście umieszcza polecenie `\binom{}{}`. Jeśli we wzorze eksponowanym napiszemy `\tbinom{}{}` to otrzymamy symbol mały zamiast dużego wstawianego poleceniem `\binom{}{}`.

$$\tbinom{a-b}{c+d} \quad \binom{a-b}{c+d}$$

□

```
$$ \tbinom{a-b}{c+d} \quad \binom{a-b}{c+d}$$
```

Znak całki

Standardowo granice całkowania są umieszczane obok znaku całki.
W **amsmath** mamy polecenie `\limits`, które umieszcza granice pod i nad znakiem całki.

```
$$\int_a^b f(x)dx \qquad \int\limits_a^b f(x)dx$$
```

¶ Przykład 6.14.

$$\int_a^b f(x)dx \qquad \int\limits_a^b f(x)dx$$

□

6.2. Pakiet amsmath

Pakiet **amsmath** powstał przy współpracy Amerykańskiego Towarzystwa Matematycznego (AMS) ku wygodzie wszystkich matematyków. W tej części omówimy kilka przydatnych środowisk zdefiniowanych w pakiecie. Aby uzyskać do nich dostęp należy w preambule dokumentu wczytać poleceniem `\usepackage` pakiet **amsmath**, dodając w nawiasach kwadratowych jedną lub kilka jego opcji (patrz tabela 6.2).

W opcjach pakietu **amsmath** możemy także umieszczać te, które zwykle umieszcza się jako opcje klasy dokumentu, a mianowicie

leqno – umieszcza numer wzoru po stronie lewej
reqno – umieszcza numer wzoru po stronie prawej (ustawienie domyślne)
fleqn – umieszcza wzory w ustalonej odległości (zdefiniowanej poleceniem `\mathindent`) od lewego marginesu a nie wyśrodkowuje ich (nie dotyczy to wzorów ujętych w dwa $).

Jeżeli użyjemy pakietu **amsmath** z opcją **fleqn**, będziemy też mogli zmienić domyślną (dodatnią) odległość wzoru od lewego marginesu i wstawić wzór bez wcięcia. Należy wtedy zmienić wielkość `mathindent` poleceniem `\setlength{\mathindent}{0pt}`.

Ku wygodzie matematyków w pakiecie **amsmath** zostały zdefiniowane różne środowiska służące do wyrównywania i numerowania wzorów wieloliniowych. Omówimy kilka najczęściej używanych.

centertags –	w środowisku `split` umieszcza numer pośrodku wysokości wysokości wzoru (ustawienie domyślne).
tbtags –	„Top-or-bottom” w środowisku `split` umieszcza numer u góry lub u dołu odpowiednio.
sumlimits –	umieszcza granice pod i nad symbolami sumowania we wzorach centrowanych (ustawienie domyślne). Dotyczy to także symboli $\prod$, $\otimes$ i $\oplus$.
nosumlimits –	zawsze umieszcza ganice w symbolach sumowania z boku znaku.
intlimits –	tak jak sumlimits tylko stosuje się do znaków całkowania.
nointlimits –	przeciwna do intlimits (ustawienie domyślne).
namelimits –	umieszcza granice (indeks dolny) pod takimi symbolami jak max, min, lim, inf jeśli pojawią sie we wzorze centrowanym (ustawienie domyślne).
nonamelimits –	ustawnienie przeciwne do namelimits.

Tabela 6.2: Opcje pakietu `amsmath`

Środowisko `align`

Środowisko `align` pozwala na wyrównanie poszczególnych wierszy wzoru względem ustalonego miejsca. Odstępy między kolumnami są dopasowywane przez program według z góry zadanych reguł.

```
\begin{align}
x+y+z &= 23\\
2x-3y+z &=x^2-4y^2-1\\
..........
\end{align}
```

¶ Przykład 6.15.

$$x + y + z = 23 \tag{6.5}$$
$$2x - 3y + z = x^2 - 4y^2 - 1 \tag{6.6}$$
$$y^4 - 4z^2 = 21 \tag{6.7}$$
$$x + y = 12k + 3w \tag{6.8}$$
$$k + w = z^2 \tag{6.9}$$

□

Znak & wskazuje miejsce wyrównywania, w każdej linii wzoru musi być tylko jeden taki znak a każdy wiersz wzoru musi być zakończony znakiem \\. W tym środowisku każdy wiersz jest opatrzony numerem. Jeśli użyjemy formy `align*` żden wzór nie będzie miał numeru, natomiast aby opatrzyć numerem tylko jedną linię, w pozostałych, przed zakończeniem, umieszczamy instrukcję `\nonumber` lub `\notag`.

Należy dbać o to, by znak & był w każdym wierszu umieszczany przed miejscem do którego będą one wyrównywane, zapewni to poprawne odstępy we wzorach. Różne rozmieszczenie znaków & spowoduje niechciane wyrównywanie poszczególnych wzorów tak jak w przykładzie 6.16.

¶ Przykład 6.16.

$$x + 2y = y^2 - 1$$
$$2x^2 - y = 24$$
$$-5x + 6y = -15$$

□

```
\begin{align*}
x + 2y &= y^2 -1\\
& 2x^2-y  = 24\\
-5x&+6y=-15
\end{align*}
```

Środowisko `align` umożliwia też umieszczenie obok siebie dwóch (lub więcej) oddzielnie wyrównanych układów równości, jak pokazaliśmy w przykładzie poniżej.
W dwóch kolumnach.

```
\begin{align}
 x & =y & z & =11\\ y & =2x+c & z & =x+y\\
 x_{1}+ k_2 & =cx^{2}+xy+1 & z & =x^{2}-12
 \end{align}
```

W trzech kolumnach.

```
\begin{align}
a&=2b & c&=d-1  & 2x&&=y+3\\
a-8&=b^2-2b+1 & c^3-1&=12d & x&=2y^2-2
\end{align}
```

¶ Przykład 6.17.

$$x = y \qquad z = 11 \tag{6.10}$$

$$y = 2x + c \qquad z = x + y \tag{6.11}$$

$$x_1 + k_2 = cx^2 + xy + 1 \qquad z = x^2 - 12 \tag{6.12}$$

i w trzech kolumnach

$$a = 2b \qquad c = d - 1 \qquad 2x = y + 3 \tag{6.13}$$

$$a - 8 = b^2 - 2b + 1 \qquad c^3 - 1 = 12d \qquad x = 2y^2 - 2 \tag{6.14}$$

□

W obu przykładach każdy wiersz jest numerowany i zagospodarowana jest cała szerokość tekstu.

Środowisko `gather`

Środowisko `gather` stosuje się do zapisywania wielu wzorów jeden pod drugim. Wzory są umieszczane w osobnych liniach, każdy jest wyśrodkowany

i zanumerowany, nie ma natomiast wyrównywania pionowego w kolumnach. Każdy wiersz z wyjątkiem ostatniego, musi być zakończony znakiem \\. Nie może być pustych linii wewnątrz środowiska.

¶ Przykład 6.18.

$$x^2 + 2xy - y^2 = 3 - xy \tag{6.15}$$
$$3x - 2y^3 + 13 = 2 - x^2 \tag{6.16}$$
$$x + y - 1 = 3xy \tag{6.17}$$
$$y = -3 \tag{6.18}$$

□

```
\begin{gather}
x^2+2xy-y^2=3-xy\\
3x-2y^3+13=2-x^2\\
x+y-1=3xy\\
y=-3
\end{gather}
```

Tak jak poprzednio, jeżeli któryś z wierszy ma nie mieć numeru, przed złamaniem umieszczamy instrukcję \notag lub \nonumber. Jeśli nie chcemy numerować żadnego wiersza, używamy środowiska gather*.

Środowiska aligned i gathered

Środowiska aligned i gathered są podobne do align i gather, lecz traktują wieloliniowy wzór jako jedną całość. aligned i gathered wymagają umieszczenia w środowisku matematycznym np. numerowanym equation lub nienumerowanym, np. pomiędzy znakami dwóch dolarów. W środowisku gathered wyrażenia są umieszczane centralnie w każdej linii tak samo jak to się dzieje w środowisku gather. Numer cała grupa bierze od zawierającego ją środowiska equation. Ponieważ wszystkie linie wzoru są tu traktowane

jak jedno wyrażenie, możliwe jest umieszczenie obok siebie dwóch wieloliniowych wzorów, jak również łączenie obu tych środowisk, tak jak w poniższych przykładach.

Układ równań w środowisku `aligned`, bez numeru:

¶ Przykład 6.19.

$$\begin{aligned} x+y+z &= 1 \\ \frac{1}{x}+\frac{1}{y}+\frac{1}{z} &= 1 \\ xyz &= -1 \\ x-z &= \frac{1}{y}-7xy \end{aligned}$$

□

```
$$
\begin{aligned}
x+y+z&=1\\
\frac{1}{x}+\frac{1}{y}+\frac{1}{z}&=1\\
xyz&=-1\\
x-z&=\frac1y-7xy
\end{aligned}
$$
```

Układ równań w środowisku `gathered`, bez numeru:

¶ Przykład 6.20.

$$\begin{gathered} x(x+y) = 3-4xy^2 \\ x(x+z) = 1 \\ x(y+z) = 2+3z^2 \\ x = -y+5z \end{gathered}$$

□

```
$$\begin{gathered}
x(x+y)=3\\x(x+z)=1\\ x(y+z)=2\\
```

```
x=-y+5z
\end{gathered}$$
```

Układ równań w środowiskach `aligned` i `gathered` razem, w jednej linii i z numerem:

¶ Przykład 6.21.

$$\begin{aligned} x+2y+(a+3)z&=8\\ 2x+3y+(a+4)z&=12\\ 3x+(6a+5)y+7z&=20 \end{aligned} \quad \text{gdy} \quad \begin{aligned} 4x+ky&=9\\ 2kx+18y&=-27 \end{aligned} \tag{6.19}$$

□

```
\begin{equation}\begin{aligned}
x+2y+(a+3)z&=8\\
2x+3y+(a+4)z&=12\\
3x+(6a+5)y+7z&=20
\end{aligned}
\textrm{\qquad gdy \qquad}
\begin{gathered}
4x+ky=9\\2kx+18y=-27
\end{gathered}\end{equation}
```

I dwa środowiska `aligned` w jednej linii i z jednym numerem:

¶ Przykład 6.22.

$$\begin{aligned} 4x+ky&=9\\ 2kx+18y&=-27 \end{aligned} \quad \text{gdy} \quad \begin{aligned} x+2y+(a+3)z&=8\\ 2x+3y+(a+4)z&=12\\ 3x+(6a+5)y+7z&=20 \end{aligned} \tag{6.20}$$

□

```
\begin{equation}\begin{aligned}
4x+ky&=9\\2kx+18y&=-27
\end{aligned}
```

```
\textrm{\quad gdy \quad}
\begin{aligned}
x+2y+(a+3)z&=8\\
2x+3y+(a+4)z&=12\\
3x+(6a+5)y+7z&=20
\end{aligned}\end{equation}
```

Uwaga: Oba środowiska mogą być użyte jedynie w trybie matematycznym.

Środowisko `split`

Środowisko **`split`** służy do rozmieszczenia na stronie wyrażeń dłuższych niż szerokość tekstu. Musi ono być umieszczone w środowisku matematycznym (numerowanym lub nie), na przykład **`equation`** lub **`equation*`**.

Uwaga: Środowiska **`split`** nie należy umieszczać w dwóch dolarach ($$) natomiast można pomiędzy \[\].

W środowisku tym możemy umieścić w każdej linii tylko jeden znak `&` (wyrównania poziomego). Jeżeli sami nie zaznaczymy miejsca wyrównania poziomego a zaznaczymy tylko miejsca złamania, nasze wyrażenie będzie sformatowane tak, jak w przykładzie 6.23, gdy wstawimy znak `&` będzie ono sformatowane inaczej, patrz przykład 6.24.

¶ Przykład 6.23.

$$X = \framebox[0.35\textwidth]{A}\\ \framebox[0.45\textwidth]{B}\\ \framebox[0.65\textwidth]{C}$$

□

```
\[ \begin{split}
X=\framebox[0.35\textwidth]{A}\\
\framebox[0.45\textwidth]{B}\\
\framebox[0.65\textwidth]{C}
\end{split} \]
```

¶ Przykład 6.24.

$$X = \begin{array}{|c|}\hline \text{A} \\ \hline \end{array} \\ \begin{array}{|c|}\hline \text{B} \\ \hline \end{array} \\ \begin{array}{|c|}\hline \text{C} \\ \hline \end{array} \\ \begin{array}{|c|}\hline \text{D} \\ \hline \end{array} \\ \begin{array}{|c|}\hline \text{E} \\ \hline \end{array} \tag{6.21}$$

□

```
\begin{equation}\begin{split}
X= &\framebox[0.35\textwidth]{A}\\
&\framebox[0.45\textwidth]{B}\\
&\framebox[0.65\textwidth]{C}\\
&\framebox[0.45\textwidth]{D}\\
&\framebox[0.15\textwidth]{E}
\end{split}\end{equation}
```

Środowisko multline

W środowisku **multline** kolejne wiersze wzoru rozmieszczane są w taki sposób, że pierwszy jest dosunięty do lewego marginesu, ostatni do prawego a środkowe w miarę możliwości wycentrowane. Domyślnie wzór jest numerowany jednym numerem umieszczonym w ostatniej linii. Jeśli nie chcemy numerować wzoru to, jak zwykle, umieszczamy go w środowisku z gwiazdką (**multline***). Każda linia, z wyjątkiem ostatniej, ma być zakończona znakiem \\.

Uwaga: Wewnątrz środowiska nie może być pustych linii.

```
\begin{multline}
x_1+x_2+x_3 +2k(x_1+x_2+x_3)+3z(x_1+x_2+x_3)\\
+x_4+x_5+x_6+x_7 -4y(x_1+x_2+x_3)\\
+x_8+x_9 -z(x_1+x_2+x_3 )
\end{multline}
```

¶ Przykład 6.25.

$$\begin{aligned} x_1 + x_2 + x_3 + 2k(x_1 + x_2 + x_3) + 3z(x_1 + x_2 + x_3) \\ + x_4 + x_5 + x_6 + x_7 - 4y(x_1 + x_2 + x_3) \\ + x_8 + x_9 - z(x_1 + x_2 + x_3) \\ - x_1 - 2x_2 - 3kx_3 \end{aligned} \quad (6.22)$$

□

Środowisko `alignat`

Środowisko `alignat` również pozwala na rozmieszczenie wzorów w wyrównanych kolumnach ale, inaczej niż w przypadku `align`, wielkość odstępu między wyrównywanymi kolumnami nie są wyliczane automatycznie. Użytkownik musi tę odległość podać sam. Należy również podać, w ilu kolumnach będziemy umieszczać i wyrównywać wyrażenia.

Każda linia umieszczona w tym środowisku posiada numer, a jeśli nie chcemy numerować wzoru, wtedy należy użyć formy `alignat*`, natomiast gdy chcemy zanumerować tylko jedną linię to, tak jak w innych środowiskach, przed zakończeniem linii umieszczamy polecenie `\nonumber` lub `\notag`. Środowisko to jest szczególnie przydatne wtedy, gdy chcemy w dokumencie umieścić kilka kolumn z krótkimi, wyrównanymi wyrażeniami.

¶ Przykład 6.26.

$$f(x) = x + 2y \qquad g(x) = x - y \quad (6.23)$$

$$h(x) = x^2 + 1 \qquad k(x) = y^2 \quad (6.24)$$

□

```
\begin{alignat}{2}
f(x) &= x+2y  & \qquad g(x)&= x-y \\
h(x) &= x^2+1 & \qquad k(x) &= y^2
\end{alignat}
```

Uwaga: Jeżeli wyrównywanych jest n kolumn w środowisku `alignat`, to wówczas musimy umieścić w każdej linii $2n-1$ znaków `&`.

Środowisko `cases`

Korzystając z możliwości pakietu `amsmath` bardzo łatwo napiszemy następujący wzór:

$$f(x) = \begin{cases} x & \text{gdy } x > 5 \\ 0 & \text{gdy } 0 \geqslant x \geqslant 5 \\ -x & \text{dla pozostałych} \end{cases} \tag{6.25}$$

lub wyrównany inaczej i bez numeru

$$f(x) = \begin{cases} x & \text{gdy } x > 5 \\ 0 & \text{gdy } 0 \geqslant x \geqslant 5 \\ -x & \text{dla pozostałych} \end{cases}$$

```
\begin{equation}
f(x)=
\begin{cases}
x␣&\text{gdy␣}␣x>5\\
0␣&\text{gdy␣}␣0\ge␣x\ge␣5\\
-x␣&\text{dla␣pozostałych}
\end{cases}
\end{equation}
```

Po przeanalizowaniu powyższego przykładu zauważamy, że użyte zostały dwa nowe polecenia. Pierwsze to wywołanie środowiska `cases`, drugie to polecenie `\text{}`. Jest ono zdefiniowane w pakiecie `amsmath` i jest równoważne poleceniu `\textrm{}`. Oznacza, że zawartość nawiasu klamrowego będzie pisana czcionką tekstową czyli antykwą. Spacja po słowie „gdy” jest konieczna aby uzyskać dodatkowy odstęp. Środowisko `cases` jest szczególnym przypadkiem środowiska `arrary`.

Może się tak zdarzyć, że w którymś wierszu wzoru będziemy musieli umieścić dość długi tekst. Wtedy należy ten tekst umieścić w pudełku (np. w `parbox`, patrz przykład 6.27) .

¶ Przykład 6.27.

$$\begin{cases} x & \text{gdy } 5 < x < 7 \\ 0 & \text{gdy } 0 \geqslant x \geqslant 5 \\ -x & \text{dla wszystkich pozostałych } x \text{ ze zbioru rozwiązań równania (2)} \end{cases} \tag{6.26}$$

□

Jeżeli tekst w ostatnim wierszu umieścimy w pudełku `parbox` z argumentem `[t]`, warunek przyjmie postać bardziej elegancką.

$$\begin{cases} x & \text{gdy } 5 < x < 7 \\ 0 & \text{gdy } 0 \geqslant x \geqslant 5 \\ -x & \text{dla wszystkich pozostałych } x \\ & \text{ze zbioru rozwiązań równania (2)} \end{cases} \tag{6.27}$$

Teraz nasz wzór wygląda o wiele lepiej. Poniżej podajemy kod całego przykładu.

```
\begin{equation}
\begin{cases}
x &\text{gdy } 5< x<7\\
0 &\text{gdy } 0\ge x\ge 5\\
-x &\text{\parbox{5.5cm}{dla wszystkich pozostałych $x$\\
ze zbioru wszystkich rozwiązań\\ równania (2.1)}}
\end{cases}
\end{equation}
```

Macierze i tablice

W pakiecie **amsmath** mamy zdefiniowane środowiska do składania macierzy, wyznaczników i innych rodzajów układów tabelarycznych używanych w matematyce. Są to `Vmatrix`, `vmatrix`, `matrix`, `Bmatrix`, `bmatrix`, `pmatrix` i `smallmatrix`. Każde z nich wywołuje się poleceniem `\begin{nazwa}` a kończy poleceniem `\end{nazwa}` i każde trzeba umieścić w odpowiednim środowisku matematycznym.

$$\begin{Vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{Vmatrix}$$ (Vmatrix)

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$ (vmatrix)

$$\begin{matrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{matrix}$$ (matrix)

$$\begin{Bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{Bmatrix}$$ (Bmatrix)

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$ (bmatrix)

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$ (pmatrix)

$\left(\begin{smallmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{smallmatrix}\right)$ (smallmatrix)

```
\begin{equation}
\begin{smallmatrix}
a_{11}&a_{12}&a_{13}\\a_{21}&a_{22}&a_{23}
\\ a_{31}&a_{32}&a_{33}
\end{smallmatrix}
\end{equation}
```

Pionowe kropki zastępujące kolejne wiersze uzyskamy stosując instrukcję `\vdots` zaś cały wiersz wypełniony kropkami zapewni nam instrukcja

```
\hdotsfor[odstęp]{ilość kolumn}
```

¶ Przykład 6.28.

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} & a_{17} \\ \hdots & & & & & & \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} & a_{27} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} & a_{47} \end{vmatrix}$$

□

Parametr `odstęp` jest to liczba określająca wielkość odstępu pomiędzy kropkami (domyślnie 1). W wyznaczniku z przykładu 6.28 odstęp wynosi 3.

Numerowanie wzorów

Wszystkie wzory umieszczane w dokumencie składanym w LaTeX-u mogą być numerowane. Numery mogą wskazywać także położenie wzoru, czyli oprócz numeru kolejnego posiadać również numer rozdziału lub/i punktu w którym się znajdują. Wzory wieloliniowe mogą mieć ponumerowaną każdą linię lub tylko jedną. Jeśli środowisko domyślnie numeruje wszystkie linie, to aby części z nich nie numerowało, należy przed złamaniem linii umieścić znacznik `\nonumber`.

Aby pozbyć się nawiasów w numerach wzorów, w preambule należy umieścić następujące instrukcje (również dostarczane przez pakiet `amsmath`)

```
\makeatletter
\def\tagform@#1{\maketag@@@{%
\ignorespaces#1\unskip\@@italiccorr}}
\makeatother
```

Aby zmienić sposób numerowania wzorów w całym dokumencie to, w zależności od potrzeb, możemy wybrać jedną z następujących możliwości.

1. Umieścić w preambule dostarczaną przez pakiet amsmath instrukcję `\numberwithin{equation}{section}`. Wzory będą numerowane dwoma liczbami, z których pierwsza to numer rozdziału a druga to kolejny numer wzoru w tym rozdziale. Numer wzoru będzie ujęty w nawiasy okrągłe.

2. Umieścić w preambule instrukcję `\renewcommand{\theequation}{\thesection--\arabic{equation}}`. Wzory będą numerowane dwoma liczbami oddzielonymi półpauzą, z których pierwsza to numer rozdziału a druga to kolejny numer wzoru w tym rozdziale. Nadal numer wzoru będzie ujęty w nawiasy okrągłe.

Etykieta zamiast numeru wzoru

Pakiet amsmath pozwala na umieszczenie własnej etykiety zamiast numeru wzoru. Realizuje to instrukcja `\tag{etykieta}` której zastosowanie pokazuje następny przykład.

¶ Przykład 6.29.

$$f(x) = ax + b \qquad \text{(funkcja liniowa)}$$
$$f(x) = ax^2 + bx + c \qquad \text{(funkcja kwadratowa)}$$
$$f(x) = \sin x + 2\cos x \qquad \text{(funkcja trygonometryczna)}$$

□

```
\begin{gather}
f(x)=ax+b \tag{funkcja liniowa}\\
f(x)=ax^2+bx+c \tag{funkcja kwadratowa}\\
f(x)= \sin x + 2 \cos x \tag{funkcja trygonometryczna}
\end{gather}
```

- Instrukcję `\tag{}` można też umieszczać w środowisku \[\], natomiast nie można używać znaków $$. W takich przypadkach LaTeX zmienia swoje zwyczaje gdy napotka `\tag{}`.
- Jest też wersja `\tag*{}` która drukuje etykietę bez nawiasów.

¶ Przykład 6.30.
Przykład etykiety dodanej do wzoru ujętego w znaki `\[ \]` lub umieszczonego w środowisku **displaymath** (żadne z tych środowisk nie jest numerowane).

$$f(x) = ax + b \qquad \text{(funkcja liniowa)}$$

I przykład zastosowania wersji z *.

$$f(x) = ax^2 + bx + c \qquad \text{funkcja kwadratowa}$$

□

```
\[ f(x)=ax+b \tag{funkcja liniowa} \]
\begin{displaymath}
f(x)=ax^2+bx+c \tag*{funkcja kwadratowa}
\end{displaymath}
```

Kiedy w preambule dokumentu wczytamy pakiet **amsmath** będziemy mogli skorzystać też ze zdefiniowanego tam środowiska **subequations**. Użycie go pozwala na numerowanie wzorów numerem kolejnym i literą, jak w przykładzie następnym. Odwołania do poszczególnych równań są realizowane w taki sam sposób jak dotychczas. Należy również pamiętać, że wewnątrz środowiska **subequations** należy umieścić kompletne środowisko matematyczne typu `\begin{nazwa} ... \end{nazwa}`.

```
\begin{subequations}
\begin{align}
f(x)&=2x+7\label{z1}\\
```

```
f(x)&=x^2-2x+3\label{z2}\\
f(x)&=x^3-2x^2\label{z3}
\end{align}
\end{subequations}
```

¶ Przykład 6.31.

$$f(x) = 2x + 7 \tag{1a}$$
$$f(x) = x^2 - 2x + 3 \tag{1b}$$
$$f(x) = x^3 - 2x^2 \tag{1c}$$

W równaniu (1a) mamy funkcję liniową, w (1b) funkcję kwadratową natomiast w (1c) funkcję stopnia trzeciego. Do wzorów odesłaliśmy czytelnika instrukcją \eqref. □

6.3. Pakiet amsthm

Pakiet **amsthm** dostarcza użytkownikom rozszerzonej wersji polecenia \newtheorem, definiuje trzy rodzaje stylów, które wybiera się poleceniem \theoremstyle oraz definiuje środowisko **proof**. Polecenie \newtheorem posiada także formę z gwiazdką (\newtheorem*) do definiowania środowisk nienumerowanych.

¶ Przykład 6.32.

Lemat 1. *Treść lematu.*

□

```
\newtheorem{lemm}{Lemat}
\begin{lemm}
Treść lematu.
\end{lemm}
```

¶ Przykład 6.33.

Uwaga. *Treść uwagi (bez numeru).*

□

```
\newtheorem*{uwa}{Uwaga}
\begin{uwa}
Treść uwagi (bez numeru).
\end{uwa}
```

Dodatkowe informacje o twierdzeniu (na przykład jego nazwę) umieszcza się bezpośrednio po wywołaniu środowiska jako argument nieobowiązkowy, czyli w nawiasach kwadratowych.

¶ Przykład 6.34.

Lemat 1 (Bardzo ważny)**.** *Treść lematu.*

□

```
\newtheorem{leme}{Lemat}
\begin{leme}[Bardzo ważny]
Treść lematu.
\end{leme}
```

Trzy style zdefiniowane w pakiecie dostarczają możliwości zróżnicowania wizualnego używanych środowisk. Są to style `plain` – domyślny, `definition` i `remark`. Każdy z nich wywołuje się poleceniem `\theoremstyle`. Wszystkie zdefiniowane w dokumencie polecenia `\newtheorem` należy podzielić na grupy i poprzedzić każdą z nich właściwym stylem.

```
\theoremstyle{plain}
\newtheorem{tw}{Twierdzenie}
\newtheorem{lem}{Lemat}
\newtheorem{prop}{Propozycja}
```

```
\theoremstyle{definition}
\newtheorem{dfn}{Definicja}
\theoremstyle{remark}
\newtheorem{uw}{Uwaga}
\newtheorem*{nota}{Notatka}
```

Aby etykieta środowiska **newtheorem** rozpoczynała się numerem jego definicję należy poprzedzić instrukcją \swapnumbers.

¶ Przykład 6.35.

1 Lemat. *Treść lematu.*

□

```
\swapnumbers
\begin{lemma}
Treść  lematu.
\end{lemma}
```

Tworzenie nowego stylu

Nowy styl środowiska **theorem** możemy utworzyć korzystając z polecenia \newtheoremstyle w sposób opisany poniżej.

```
\newtheoremstyle{arg1}{arg2}{arg3}{arg4}{arg5}{arg6}%
{arg7}{arg8}{arg9}
```

gdzie

arg1	Nazwa środowiska.
arg2	Tu podajemy wielkość odstępu od ostatniego wiersza tekstu.
arg3	Tu podajemy wielkość odstępu od następnego wiersza tekstu.
arg4	Rodzaj fontu którym ma być złożone twierdzenie.
arg5	Wielkość wcięcia rozpoczynającego środowisko. Może przybierać wartość **empty** – brak wcięcia lub \partindent – wcięcie akapitowe.

`arg6` Font etykiety (nagłówka).

`arg7` Rodzaj znaku przestankowego po nagłówku.

`arg8` Odstęp po nagłówku. Może przyjmować wartość np. 0.5em, może też przechodzić do następnej linii (`\linebreak`)

`arg9` Umieszcza się tu dodatkowe informacje które mogą znaleźć się w nagłówku twierdzenia, argument może być pusty – co oznacza nagłówek standardowy.

¶ Przykład 6.36.

Przypisek 1.
To jest przykład nowo zdefiniowanego środowiska.

□

```
\newtheoremstyle{break}{2ex}{2ex}{\slshape}%
{}{\scshape}{.}{\newline}{}
\theoremstyle{break}
\newtheorem{przypis}{Przypisek}
\begin{przypis}
To jest przykład nowo zdefiniowanego środowiska.
\end{przypis}
```

Środowisko `proof`

¶ Przykład 6.37.

Dowód. To jest przykład użycia środowiska `proof`. □

Dowód Twierdzenia 1.2. To jest inny przykład. □

Dowód(⇒). I jeszcze jeden przykład. □

□

```
\begin{proof}
To jest przykład użycia środowiska \texttt{proof}.
\end{proof}
\begin{proof}[Dowód Twierdzenia 1.2]
To jest inny przykład.
\end{proof}
\begin{proof}[Dowód\textup($\Rightarrow$\textup)]
To jest inny przykład użycia środowiska \texttt{proof}.
\end{proof}
```

6.4. Wybrane symbole matematyczne

1. Podstawowe symbole relacji

$<$ `<`	$>$ `>`	$\leqslant$ `\le`	$\geqslant$ `\ge`
$\ll$ `\ll`	$\gg$ `\gg`	$\doteq$ `\doteq`	$\simeq$ `\simeq`
$=$ `=`	$\equiv$ `\equiv`	$\sim$ `\sim`	$\approx$ `\approx`
$\subset$ `\subset`	$\supset$ `\supset`	$\in$ `\in`	$\ni$ `\ni`
$\mid$ `\mid`	$\parallel$ `\parallel`	$\notin$ `\notin`	$\neq$ `\neq`
$\neq$ `\not=`	$\ngeqslant$ `\not\ge`	$\not>$ `\not>`	$\not<$ `\not<`

2. Symbole niektórych operacji dwuargumentowych

$+$ `+`	$-$ `-`	$\pm$ `\pm`	$\mp$ `\mp`
$\cdot$ `\cdot`	$\div$ `\div`	$\times$ `\times`	$\setminus$ `\setminus`
$\star$ `\star`	$*$ `\ast`	$\circ$ `\circ`	$\bullet$ `\bullet`
$\cup$ `\cup`	$\cap$ `\cap`	$\vee$ `\vee`	$\wedge$ `\wedge`
$\oplus$ `\oplus`	$\ominus$ `\ominus`	$\odot$ `\odot`	$\otimes$ `\otimes`

3. Akcenty w trybie matematycznym

$\hat{o}$ `\hat{o}`	$\tilde{o}$ `\tilde{o}`	$\widehat{o}$ `\widehat{o}`
$\widetilde{o}$ `\widetilde{o}`	$\check{o}$ `\check{o}`	$\breve{o}$ `\breve{o}`

$\dot{o}$ \dot{o}	$\ddot{o}$ \ddot{o}	$\bar{o}$ \bar{o}
$\vec{o}$ \vec{o}	$\grave{o}$ \grave{o}	$\acute{o}$ \acute{o}

4. Symbole zmiennej wielkości

$\sum$ \sum	$\bigcap$ \bigcap	$\bigcup$ \bigcup
$\bigsqcup$ \bigsqcup	$\prod$ \prod	$\coprod$ \coprod
$\bigvee$ \bigvee	$\bigwedge$ \bigwedge	$\bigoplus$ \bigoplus
$\int$ \int	$\iint$ \iint	$\oint$ \oint
$\bigodot$ \bigodot	$\biguplus$ \biguplus	$\bigotimes$ \bigotimes

5. Niektóre ograniczniki

$\rangle$ \rangle	$\langle$ \langle	$\rfloor$ \rfloor
$\lfloor$ \lfloor	$\vert$ \vert lub |	$\Vert$ \\| lub \Vert
$\backslash$ \backslash	/ /	

6. Litery greckie

α \alpha	θ \theta	\o	υ \upsilon
β \beta	ϑ \vartheta	π \pi	ϕ \phi
γ \gamma	ι \iota	ϖ \varpi	φ \varphi
δ \delta	κ \kappa	ρ \rho	χ \chi
ϵ \epsilon	λ \lambda	ϱ \varrho	ψ \psi
ε \varepsilon	μ \mu	σ \sigma	ω \omega
ζ \zeta	ν \nu	ς \varsigma	$\varkappa$ \varkappa
η \eta	ξ \xi	τ \tau	$\digamma$ \digamma
$\aleph$ \aleph	$\beth$ \beth	$\daleth$ \daleth	$\gimel$ \gimel
Γ \Gamma	Λ \Lambda	Σ \Sigma	Ψ \Psi
Δ \Delta	Ξ \Xi	Υ \Upsilon	Ω \Omega
Θ \Theta	Π \Pi	Φ \Phi	

7. Niektóre strzałki

$\leftarrow$ \leftarrow	$\rightarrow$ \rightarrow lub \to
$\longleftarrow$ \longleftarrow	$\longrightarrow$ \longrightarrow
$\Leftarrow$ \Leftarrow	$\Rightarrow$ \Rightarrow
$\Longleftarrow$ \Longleftarrow	$\Longrightarrow$ \Longrightarrow
$\leftrightarrow$ \leftrightarrow	$\longleftrightarrow$ \longleftrightarrow
$\Leftrightarrow$ \Leftrightarrow	$\Longleftrightarrow$ \Longleftrightarrow
$\mapsto$ \mapsto	$\longmapsto$ \longmapsto
$\hookleftarrow$ \hookleftarrow	$\hookrightarrow$ \hookrightarrow
$\uparrow$ \uparrow	$\downarrow$ \downarrow
$\Uparrow$ \Uparrow	$\Downarrow$ \Downarrow
$\updownarrow$ \updownarrow	$\Updownarrow$ \Updownarrow
$\nearrow$ \nearrow	$\searrow$ \searrow
$\nwarrow$ \nwarrow	$\swarrow$ \swarrow

8. Litery – różne style (tylko w środowisku matematycznym)

Kaligraficzne: $\mathcal{A}$ `\mathcal{A}` ... itd.
$\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}, \mathcal{E}, \mathcal{F}, \mathcal{G}, \mathcal{H}, \mathcal{I}, \mathcal{J}, \mathcal{K}, \mathcal{L}, \mathcal{M}, \mathcal{N}, \mathcal{O}, \mathcal{P}, \mathcal{Q}, \mathcal{R}, \mathcal{S}, \mathcal{T}, \mathcal{U}, \mathcal{V}, \mathcal{W}, \mathcal{Y}, \mathcal{Z}$
Mathbb: $\mathbb{A}$ `\mathbb{A}` ... itd.
$\mathbb{A}, \mathbb{B}, \mathbb{C}, \mathbb{D}, \mathbb{E}, \mathbb{F}, \mathbb{G}, \mathbb{H}, \mathbb{I}, \mathbb{J}, \mathbb{K}, \mathbb{L}, \mathbb{M}, \mathbb{N}, \mathbb{O}, \mathbb{P}, \mathbb{Q}, \mathbb{R}, \mathbb{S}, \mathbb{T}, \mathbb{U}, \mathbb{V}, \mathbb{W}, \mathbb{Y}, \mathbb{Z}$
Mathfrak: $\mathfrak{A}$ `\mathfrak{A}` ... itd.
$\mathfrak{A}, \mathfrak{B}, \mathfrak{C}, \mathfrak{D}, \mathfrak{E}, \mathfrak{F}, \mathfrak{G}, \mathfrak{H}, \mathfrak{I}, \mathfrak{J}, \mathfrak{K}, \mathfrak{L}, \mathfrak{M}, \mathfrak{N}, \mathfrak{O}, \mathfrak{P}, \mathfrak{Q}, \mathfrak{R}, \mathfrak{S}, \mathfrak{T}, \mathfrak{U}, \mathfrak{V}, \mathfrak{W}, \mathfrak{Y}, \mathfrak{Z},$
$\mathfrak{a}, \mathfrak{b}, \mathfrak{c}, \mathfrak{1}, \mathfrak{2}, \mathfrak{3}$
Matematyczne bezszeryfowe: A `\mathsf{A}` ... itd.
$\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}, \mathsf{F}, \mathsf{G}, \mathsf{H}, \mathsf{I}, \mathsf{J}, \mathsf{K}, \mathsf{L}, \mathsf{M}, \mathsf{N}, \mathsf{O}, \mathsf{P}, \mathsf{Q}, \mathsf{R}, \mathsf{S}, \mathsf{T}, \mathsf{U}, \mathsf{V}, \mathsf{W}, \mathsf{Y}, \mathsf{Z},$
$\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{1}, \mathsf{2}, \mathsf{3}$
Matematyczne pogrubione: $\mathbf{A}$ `\mathbf{A}` ... itd.
$\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}, \mathbf{F}, \mathbf{G}, \mathbf{H}, \mathbf{I}, \mathbf{J}, \mathbf{K}, \mathbf{L}, \mathbf{M}, \mathbf{N}, \mathbf{O}, \mathbf{P}, \mathbf{Q}, \mathbf{R}, \mathbf{S}, \mathbf{T}, \mathbf{U}, \mathbf{V},$
$\mathbf{W}, \mathbf{Y}, \mathbf{Z}, \mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{1}, \mathbf{2}, \mathbf{3}$

9. Różne przydatne symbole

$a \ldots b$ \dots	$a \cdots b$ \cdots	$\vdots$ \vdots	$\ddots$ \ddots
$\Re$ \Re	$\Im$ \Im	$\aleph$ \aleph	$\wp$ \wp
$\forall$ \forall	$\exists$ \exists	∂ \partial	ℓ \ell
$\emptyset$ \emptyset	∞ \infty	$\angle$ \angle	$\Box$ \Box

7. Grafika

7.1. Kolory

Modele kolorów

Używanie koloru w dokumencie wymaga uświadomienia sobie, że istnieją dwa ważne modele kolorów CMYK i RGB. Wybór któregoś z nich zależy od sposobu publikacji naszego dokumentu.

Nazwa CMYK to skrót utworzony z pierwszych liter angielskich nazw kolorów: `cyan` – odcień niebieskiego, `magenta` – czerwono-różowy, `yellow` – żółty oraz czwarty kolor zwany `key color` – z reguły czarny. Kolory te mieszają się ze sobą (np. niebieski z żółtym daje zielony) i pozwalają uzyskać tysiące innych kolorów. Mówimy wówczas o druku czterokolorowym. Model CMYK powinien być stosowany w tych dokumentach, które docelowo mają być drukowane za pomocą prasy drukarskiej, na przykład techniką offsetową. W modelu kolorów CMYK barwa biała to brak wszystkich kolorów, a barwa czarna otrzymywana jest gdy połączymy (zmieszamy) wszystkie cztery kolory.

Nazwa RGB to skrót utworzony z pierwszych liter angielskich nazw kolorów: `red` – kolor czerwony, green – kolor zielony oraz `blue` – kolor niebieski. Połączenie wszystkich kolorów w tym modelu daje nam barwę białą, a ich brak – barwę czarną. Model RGB powinien być stosowany w tych kolorowych dokumentach, które mają być oglądane na monitorze telewizora, komputera czy telefonu.

Po wczytaniu pakietu `color` w preambule dokumentu, możemy używać obu tych modeli, a także dodatkowego modelu `gray` – umożliwiającego uzyskanie na rysunku różnych odcieni szarości. Mamy też możliwość skorzystania z listy kolorów o nazwie `named`. Taki kolor wywołuje się podając jego nazwę, np. `red` lub `yellow`. Poniżej opisujemy sposoby określania kolorów w poszczególnych modelach.

Model RGB:	trzy liczby rzeczywiste pomiędzy 0 i 1 oddzielone przecinkiem określają ilość każdego z komponentów,
Model CMYK:	cztery liczby rzeczywiste pomiędzy 0 i 1 oddzielone przecinkami, określające ilość każdego z komponentów,
Skala szarości:	liczba rzeczywista pomiędzy 0 (czarny) i 1 (biały),
Model named:	kolory dostępne przez podanie ich nazwy, np. 'red' .

Definiowanie i używanie kolorów

W LaTeX-u kolory przywołujemy używając ich nazwy lub poprzez konstrukcję `[model]{specyfikacja}`, gdzie `model` jest jednym z modeli `rgb`, `cmyk`, `named` lub `gray`.

Nowy kolor możemy zdefiniować poleceniem `\definecolor`. Możemy je umieścić gdziekolwiek w dokumencie, ale przed przywołaniem definiowanego koloru.

```
\definecolor{nazwa}{model}{określenie koloru}
```

¶ Przykład 7.1.

Napisaliśmy tekst kolorem białym na szarym tle. Kolor szary zdefiniowaliśmy sami. □

```
\definecolor{mojszary}{gray}{0.75}
\colorbox{mojszary}{\textcolor{white}{\bf Napisaliśmy}}
tekst kolorem białym na szarym tle. Kolor szary
zdefiniowaliśmy sami.
```

Aby tekst został napisany kolorem innym niż czarny należy użyć polecenia `\textcolor{nazwa}{tekst}` i umieścić go jako drugi argument. Pierwszym argumentem jest zawsze nazwa koloru.

¶ Przykład 7.2.

Dwa wyrazy będą wyróżnione innym kolorem. □

```
\textcolor[cmyk]{0.8,0.1,0.3,0.2}{Dwa wyrazy} będą
wyróżnione \textcolor[rgb]{0.2,0.9,0.2}{innym} kolorem.
```

Poniżej w tabeli umieściliśmy instrukcje zmieniające kolor poszczególnych elementów tekstu.

`\color{nazwa}`	globalna deklaracja koloru tekstu,
`\color[model]{specyfikacja}`	globalna deklaracja koloru określonego poprzez podanie modelu koloru i jego specyfikacji,
`\textcolor{nazwa}{tekst}`	lokalna deklaracja koloru tekstu,
`\textcolor[model]{specyfikacja}`	lokalna deklaracja koloru określonego poprzez podanie modelu koloru i jego specyfikacji,
`\colorbox{nazwa}{tekst}`	kolor tła pudełka tekstu,
`\colorbox[model]{specyfikacja}`	kolor tła pudełka tekstu określony poprzez podanie modelu koloru i jego specyfikacji,
`\fcolorbox{nazwa}{tekst}`	kolor tła pudełka z ramką,
`\fcolorbox[model]{specyfikacja}`	kolor tła pudełka z ramką, określony poprzez podanie modelu koloru i jego specyfikacji,

`\fcolorbox{nazwa1}{nazwa2}{tekst}`	kolor tła pudełka z ramką oraz kolor ramki,
`\fcolorbox[model]{spec1}{spec2}`	kolor tła pudełka z ramką oraz kolor ramki, określony poprzez podanie modelu koloru i jego specyfikacji,
`\pagecolor{nazwa}{tekst}`	zmienia kolor tła strony bieżącej i następnych
`\normalcolor`	przywraca domyślny kolor dokumentu.

Przykłady zilustrują zastosowanie niektórych poleceń z tabelki.

¶ Przykład 7.3.

... Antykwy są na ogół proste, a kursywy pochylone w prawo. Tym jednak, co odróżnia te dwie odmiany, jest nie pochylenie, lecz płynność pisma. Kursywa ma bardziej płynny dukt niż antykwa, a to oznacza, że swoją ogólną formą bardziej przypomina pismo odręczne... [1]. □

```
\colorbox{green}{\textcolor{yellow}{Antykwy}} są na ogół
proste, a \fcolorbox{red}{white}{kursywy} pochylone ...
\fcolorbox{blue}{yellow}{Kursywa} ma bardziej płynny ...
```

¶ Przykład 7.4.

... Antykwy są na ogół proste, a kursywy pochylone w prawo. Tym jednak, co odróżnia te dwie odmiany, jest nie pochylenie, lecz płynność pisma. Kursywa ma bardziej płynny dukt niż antykwa a to oznacza, że swoją ogólną formą bardziej przypomina pismo odręczne. Szeryfy kursywy ... [2]. □

[1] Z książki *Elemetarz stylu w Typografii* Roberta Bringhursta
[2] Z książki *Elemetarz stylu w Typografii* Roberta Bringhursta

```
\colorbox[rgb]{0.3,0.6,0} {Antykwy} są na ogół proste,
a \fcolorbox[rgb]{1,0,0}{1,1,1}{kursywy} pochylone
(...) \color[rgb]{1,1,0.1}{Kursywa ma bardziej płynny
dukt niż antykwa} \normalcolor a to oznacza, że swoją
ogólną formą bardziej przypomina ...
```

7.2. Pakiet graphicx

Możliwości graficzne LaTeX-a możemy rozszerzyć wczytując pakiet **graphicx**. Jest on rozszerzeniem wcześniejszego pakietu `graphics`, pakiety te różnią się nieznacznie, głównie formatem argumentów opcjonalnych w poleceniach. W typowej dystrybucji systemu TeX są one instalowane domyślnie.

Opcje pakietu

draft Powoduje wydrukowanie dokumentu bez wstawionej grafiki, zaznacza jednak wymiary pudełka, w które ten obrazek ma być wstawiony.

final Przeciwne do opcji poprzedniej. Używane często do nadpisania opcji [draft] zadeklarowanej w opcjach klasy wczytywanych wraz z poleceniem `\documentclass`.

hiderotate Nie pokazuje obróconego tekstu.

hidescale Nie pokazuje skalowanego tekstu.

hiresbb Poszukuje specyfikacji rozmiaru obrazka w postaci `%%HiResBoundingBox` zamiast `%%BoundingBox` lines.

demo Zamiast obrazka wstawianego instrukcją `\includegraphics` rysuje prostokąt o wymiarach 150pt na 100pt, nawet gdy rozmiar obrazka jest inny.

Najczęściej używane opcje to `draft` i `final`. Pakiet umożliwia wstawianie grafiki do dokumentu oraz obracanie i skalowanie obrazów i tekstu.

Obracanie

Obrót jest realizowany za pomocą instrukcji `\rotate`, którą stosujemy następująco:

```
\rotatebox{kąt}{text}
\rotatebox{lista argumentów}{kąt}{text}
```

Pierwsza instrukcja pochodzi z pakietu `graphics`, wstawia obiekt (tekst) w pudełko, na przykład `\mbox` i obraca o kąt podany jako argument. Jako środek obrotu wybierany jest w tym przypadku lewy dolny wierzchołek pudełka.

¶ Przykład 7.5.

Jako środek obrotu wybierany jest w tym przypadku lewy dolny wierzchołek pudełka.

□

```
\rotatebox{30}{Jako środek} obrotu (...)
\rotatebox{45}{lewy dolny} wierzchołek
\rotatebox{60}{pudełka}.
```

Jeśli za środek obrotu chcemy przyjąć inny punkt powinniśmy użyć drugiego polecenia, wybierając właściwy argument z „listy". Są to

- `origin = <etykieta>`
- `x = <rozmiar>`
- `y = <rozmiar>`
- `units = <numer>`.

Można podać obydwie współrzędne (x,y) środka obrotu albo w argumencie 'origin' jako etykietę podać jedną lub dwie litery z następujących |lrctbB|, gdzie |B| oznacza linię bazową. Na przykład obrót o 180°

bez przesunięcia środka obrotu ...Przykładowy tekst...

i po przesunięciu na poziom linii bazowej (c)

...Przykładowy tekst...a teraz w prawo do góry (tr)

...Przykładowy tekst...

Można także zmieniać domyślny kierunek obrotu (przeciwnie do ruchu wskazówek zegara) podając wartość argumentu 'units' w stopniach lub radianach, na przykład units = -360 dla obrotu w kierunku ruchu zegara (units= 6.283185 dla obrotu o kąt podany w radianach).

Skalowanie

Pakiet graphicx umożliwia też skalowanie wstawianych obiektów. Składnia polecenia jest następująca:

```
\scalebox {h-skala}[v-skala]{tekst}
```

Można podać tylko poziomy argument skalowania, pionowy zostanie wtedy dobrany proporcjonalnie.

¶ Przykład 7.6.

Nowy przykład.
Tekst powiększony
i rozciągnięty.

□

```
\scalebox{3}[.7]{Nowy przykład. Tekst powiększony ...}
```

Odbicie lustrzane realizuje polecenie `\reflectbox{tekst}`. Jest to skrócona forma `\scalebox{-1}[1]{tekst}`.

¶ Przykład 7.7.

.mrzsrtsuJ ucicibo w ɟǝʞsT

□

```
\reflectbox{Tekst w odbiciu lustrzanym.}
```

Możemy także przeskalować obiekt do zadanej wielkości.

¶ Przykład 7.8.

Jakiś tekst. Tekst ma szerokość 1 cala i wysokość tekstu bieżącego.

□

```
\resizebox{1in}{\height}{Jakiś tekst}
```

¶ Przykład 7.9.

Jakiś tekst. Tu wysokość została wyliczona stosownie do szerokości.

□

```
\resizebox{2in}{!}{Jakiś tekst}
```

7.3. Wstawianie plików graficznych

Pliki graficzne do dokumentu możemy wstawiać poleceniem

```
\includegraphics[argument]{plik}
```

Możemy sterować rozmiarami i położeniem rysunku, umieszczając w miejscu argumentu następujące instrukcje:

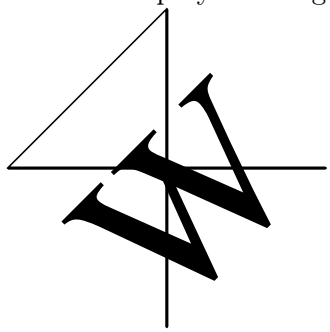
`angle` – kąt obrotu grafiki,
`width=<liczba>` – szerokość,

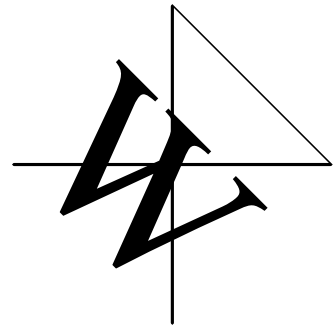
`height=<liczba>`	–	wysokość,
`origin`	–	środek obrotu - patrz `\rotatebox`,
`scale=<liczba>`	–	liczba określająca skalę powiększenia.

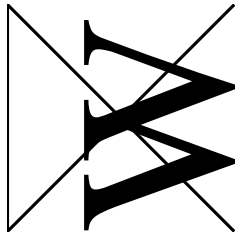
Należy pamiętać, że LaTeX nie rozumie wszystkich formatów graficznych. Jeżeli przetwarzamy nasz dokument *latexem*, wówczas grafika musi być w formacie eps. Jeżeli używamy *pdflatex-a* i mamy wersję systemu TeXLive starszą niż 2011, możemy wstawiać do dokumentu grafikę tylko w formatach pdf i jpg. Jeśli zaś mamy wersję systemu 2011 i nowszą, możemy z *pdflatex-em* używać oprócz pdf i jpg także plików w formacie eps, bo te zostaną „w locie" zamienione na pdf-y.

¶ Przykład 7.10.

A oto kilka przykładów graficznych.







□

```
\includegraphics[width=3cm,angle=45]{graf.pdf}
\qquad \includegraphics[width=3cm,angle=-45]{graf.pdf}
\qquad \includegraphics[width=3cm,angle=-270]{graf.pdf}
```

Pliki graficzne mogą znajdować się w dowolnych katalogach ale trzeba podać jakie katalogi mają być przeszukiwane poleceniem
`\graphicspath{lista katalogów}`
lub całą ścieżkę dostępu do plików graficznych
`\includegraphics[width=330pt]{d:/Fol2012/cgraf/(2,2,2).pdf}`
Można też ułatwić systemowi zadanie odnalezienia właściwego pliku graficznego deklarując poleceniem
`\DeclareGraphicsExtensions{lista rozszerzeń}`
jakiego typu plików graficznych będziemy używać. Lista rozszerzeń to nazwy typów oddzielone od siebie przecinkiem.

8. Dodatek

8.1. Typografia, typograf

Typografia zajmuje się reprezentacją graficzną znaków pisarskich. Dostarcza narzędzi (liter, znaków) wraz ze zbiorem zasad, zwyczajów i konwencji dotyczących ich używania. Na przestrzeni wieków kształty liter ulegały ciągłym, choć nieznacznym zmianom, powstawały i powstają wciąż nowe kroje, a także zmieniają się narzędzia używane do składania tekstu. Zasady typograficznej przejrzystości tekstu nie zmieniły się istotnie od czasu druku pierwszej książki w XV wieku.

Istnieje wiele rozmaitych krojów pisma ale nie ma takiego kroju który nadawałby się do przedstawienia każdego tekstu. Dobranie odpowiedniego kroju pisma do treści jakie mają być przy jego pomocy utrwalone jest zadaniem typografa.

Rolę typografa w projektowaniu dokumentu trafnie wyjaśnia cytat z książki Roberta Bringhursta [1]:

> *Typograf jest dla tekstu tym, kim reżyser dla scenariusza, czy też muzyk dla partytury. (. . .) Dowolny przyrząd do składania tekstu, czy to komputer, czy wierszownik, działa jak warsztat tkacki. A typograf niczym skryba, pragnie utkać jak najbardziej jednolity tekst.*

Litery wyjaśniają to co widzimy, ich kształt zatem nie może się kłócić z treścią. Należy kierować się zasadą, że tekst musi swoim wyglądem zachęcać

do przeczytania. Innym krojem należy więc składać karty wizytowe a innym duże plakaty czy banery.

¶ Przykład 8.1.

Napis „Proszę zamykać drzwi" złożony kursywą będzie mniej czytelny niż ten sam napis złożony pogrubionymi wersalikami.

PROSZĘ ZAMYKAĆ DRZWI *Proszę zamykać drzwi*

□

8.2. Strona tekstu

Gęstość struktury tekstu zadrukowanego na stronie nazywana jest kolorem. Na etapie projektowania dokumentu: książki, ulotki czy plakatu należy, poza spełnieniem wymogów czytelności i porządku logicznego składanego dokumentu, zadbać o osiągnięcie tzw. równomiernej szarości kolumny. Uzyskuje się ją dobierając właściwy dla danego dokumentu krój pisma, jego wielkość oraz odstępy między wyrazami i wierszami.

Ciąg znaków danego kroju tworzący wyrazy i zdania ma zawsze określoną szerokość tzn. na pole znakowe które zajmuje litera składa się jej szerokość i połowy odległości od poprzedniego i następnego znaku. Aby zwiększyć czytelność tekstu zapisanego konkretnym krojem pisma stosuje się tzw. *kerning* (*podcinanie*), który polega na przysuwaniu do siebie liter w celu poprawy świateł międzywyrazowych. Komputerowe systemy składu modyfikują te szerokości za pomocą tablic kerningu. TEX również posiada taką własność.

Rozmieszczenie tekstu na stronie

Tekst ciągły może być rozmieszczony na stronie na dwa sposoby:

- wyrównany lewostronnie (tzw. skład w chorągiewkę) – odstępy międzywyrazowe będą stałe,
- wyrównany obustronnie (tzw. wyjustowany) – odstępy pomiędzy wyrazami na ogół nie są jednakowe.

Przy wąskich kolumnach najlepiej jest stosować skład w chorągiewkę, zapobiegniemy wtedy pojawianiu się nadmiernej ilości przenoszonych wyrazów, tzw. „epidemii dywizów". Skład w chorągiewkę należy również stosować, gdy w tekście pojawia się dużo wyrazów trudnych do przeniesienia, np. nazw własnych. Zapobiegniemy wówczas powstawaniu zbyt dużych odstępów międzywyrazowych i tworzeniu się niechcianych dziur w tekście. W składzie w chorągiewkę lepiej prezentują się kroje bezszeryfowe.

W składzie wyjustowanym trzeba zadbać o umiar pomiędzy częstotliwością przenoszenia wierszy i powiększaniem odstępów międzywyrazowych.

Uważa się, że w składzie jednołamowym (jedna kolumna tekstu na stronie) optymalna ilość znaków w wierszu złożonym pismem szeryfowym w stopniu tekstowym to 66, licząc litery i odstępy. W układzie wielołamowym średnia liczba znaków w wierszu powinna wynosić od 40 do 50.

Dobrze jest pamiętać i stosować poniższe zasady.

1. Odstępy międzywyrazowe powinny być dobrane do stopnia pisma i jego rodzaju (odmiany).
2. Szerokość tekstu w kolumnie strony powinna być dobrana do treści.
3. Skład w chorągiewkę należy stosować wszędzie tam, gdzie tekst i układ strony tego wymagają.
4. Między wyrazami stosuje się pojedynczy odstęp międzywyrazowy.
5. Interlinia podstawowa to odstęp między wierszami. Jej wielkość zależy od stopnia pisma, kroju pisma oraz szerokości łamu. Nie należy zwiększać interlinii bez wyraźnej potrzeby.
6. Należy pamiętać, że strona, na której umieszczono zbyt wiele linii tekstu „nie oddycha".

> *W świecie cyfrowym projektant lub składacz pozbawiony szacunku dla liter bez trudu może je wcisnąć w bydlęce wagony i wysłać na rzeź*[1].

[1]Cytat z książki *Elementarz stylu w typografii* Roberta Bringhursta

8.3. Pismo

Aby z literami i wyrazami nie stało się to, o czym mówi przytoczony wyżej cytat, należy zacząć od umiejętności rozróżniania krojów pisma. Nie chodzi tu tylko o nazwy konkretnych krojów, ale przede wszystkim o odróżnienie np. krojów szeryfowych od bezszeryfowych, odmian wąskich od szerokich.

Od najdawniejszych czasów pisano na skałach lub budowlach (dłutem lub rylcem) oraz na pergaminie, później na papierze odręcznie tuszem. Do napisów odświętnych (szczególnie na murach) używane były duże litery i stąd ich nazwa MAJUSKUŁY od słowa łacińskiego maiusculus – większy. Małe litery to MINUSKUŁY, od słowa minusculus – dość mały.

Pismo jest to zbiór znaków pozwalający na widzialne przedstawianie myśli lub mowy. Pismem jest na przykład alfabet polski. Pisma europejskie używają jednego alfabetu składającego się z 26 liter, do którego w zależności od potrzeb danego języka włączono litery ze specyficznymi akcentami graficznymi – tzw. *znakami diakrytycznymi*. I tak na przykład język hiszpański ma 30 liter, polski 32 litery, a włoski tylko 21 liter – bez j, k, w, x oraz y. Te litery występują tylko w wyrazach obcego pochodzenia.

Znaki pisma nie posiadają ściśle określonego kształtu, a jedynie pewną ogólną strukturę (określającą układ i wzajemne relacje elementów). Kiedy poszczególnym znakom przyporządkowuje się konkretne kształty, to powstaje krój pisma (ang. typeface). Dobrze jest, gdy wszystkie znaki w danym kroju są zaprojektowane tak, by posiadały jednolite cechy graficzne (np. styl, rytm, dukt itp.).

Krój pisma utrwala się na nośniku. Nośnikiem takim może być metalowa czcionka, nośnikiem takim może być także font, który jest plikiem komputerowym, w którym kształty wszystkich znaków jednego kroju są zakodowane w postaci cyfrowej.

W tradycyjnym drukarstwie odpowiednikiem fontu jest komplet czcionek lub garnitur pisma, więc nazywanie fontu czcionką nie jest właściwe, choć

często używa się tych określeń zamiennie.

Możemy przyjąć, że font (także czcionka) ma trzy wyróżniające go parametry:

- Krój (ang. typeface) – odpowiada za wygląd liter,
- Odmiana – określa grubość linii w literach oraz (lub) stopień pochylenia liter,
- Stopień – określa wielkość liter.

Zbiór wszystkich odmian danego fontu to rodzina. Wśród odmian są na przykład: light, regular, italic, bold, medium italic, bold italic. Nie każda rodzina fontów składa się z wszystkich wymienionych odmian.

Podstawowa odmiana kroju określana jest najczęściej jako regular lub normal (też antykwa), jego cieńsza wersja jako light, a grubsza jako bold. W danym kroju zazwyczaj istnieje odmiana pochyła zwana kursywą (ang. *italic*), wzorowana na kursywie renesansowej. Projektant kroju może także stworzyć inną odmianę pochyłą (ang. *slanted*), polegającą na pochyleniu kresek znaków prostych. Wiele krojów posiada także odmiany szeroką lub wąską oraz imitującą pismo maszynowe.

Ze względu na wygląd (kształt) możemy fonty (także czcionki) podzielić na kilka grup.

- Kroje proste i pochyłe.
- Kroje szeryfowe i bezszeryfowe.
- Kroje jednoelementowe i dwuelementowe.
- Kroje stałe i proporcjonalne.

Inną kategorię krojów stanowią kroje imitujące pismo odręczne czyli tzw. pisanki.

O czym należy pamiętać

Podamy kilka istotnych informacji (patrz [2]), o których należy pamiętać w trakcie projektowania i tworzenia dokumentu.

Litery

Wersaliki, inaczej majuskuły lub litery wielkie, charakteryzują się jednakową wysokością. Wywodzą się z przyjętego przez Rzymian w VI wieku p.n.e. pisma greckiego.

Minuskuły, inaczej litery tekstowe lub małe to litery z wydłużeniami dolnymi i górnymi. Rozróżniamy następujące rodzaje liter tekstowych:

litery bez wydłużeń	a e z c n m
litery z wydłużeniami dolnymi	g j y p q
litery z wydłużeniami górnymi	k l f d b t

Kapitaliki, inaczej mediuskuły, to litery wyglądające jak majuskuły ale mieszczące się zazwyczaj pomiędzy linią podstawową i środkową pisma. Kapitaliki stosowane są często do wyróżnień i zapisywania skrótowców np. IBM, PKP

Liczby rzymskie i cyfry arabskie

Rzymianie używali 7 ze swoich liter do zapisu podstawowych liczb

$$\text{I} - 1,\ \text{V} - 5,\ \text{X} - 10,\ \text{L} - 50,\ \text{C} - 100,\ \text{D} - 500,\ \text{M} - 1000$$

Rozwój kształtów cyfr arabskich następował powoli i niezależnie od kształtów liter. W niektórych krojach pism możemy znaleźć cyfry nautyczne, zwane także mediewalnymi. Bazują one na renesansowej tradycji typograficznej. Są tak zaprojektowane, aby wizualnie nie różniły się od liter tekstowych. Osiągnięto to poprzez zastosowanie do niektórych z nich wydłużeń górnych lub dolnych. Poniżej przykład cyfr nautycznych i cyfr zwykłych.

1234567890 1234567890

Tytuły, łamanie i przenoszenie wyrazów

1. W tytułach nie pozostawia się wyrazów jednoliterowych na końcu wiersza.
2. Tytułów nie kończy się kropką.
3. Treść w tytułach wielowierszowych dzieli się tak, aby każdy wiersz zawierał logicznie zakończone pojęcie.
4. Nie dopuszcza się rozpoczynania nowej kolumny (strony) ostatnim wierszem akapitu z poprzedniej kolumny (strony), tzw. bękartem.
5. Dopuszcza się pozostawienie u dołu kolumny (strony) pierwszego wiersza akapitu, tzw. szewca.
6. Wyrazów jednosylabowych nie dzieli się i nie przenosi.
7. Liczb nie dzieli się i nie przenosi.
8. Nie dzieli się i nie przenosi wyrazów po pierwszej literze.
9. Długość ostatniego wiersza akapitu powinna być większa niż wcięcie akapitowe.
10. Nie pozostawia się na końcu wiersza wyrazów jednoliterowych (a, i, o, u, w oraz z).

Znaki interpunkcyjne

1. Przed kropką lub innym znakiem interpunkcyjnym kończącym zdanie nie umieszcza się dostępu.
2. Między dwoma zdaniami a więc po kropce lub innym znaku interpunkcyjnym kończącym zdanie należy stosować pojedynczy odstęp. Odstępy większe same są już znakami interpunkcyjnymi.
3. W ciągach inicjałów A.K. Kowalski, R.M.T. Nowak nie wstawia się spacji po kropce. Zwykła spacja międzywyrazowa występuje w ciągu inicjałów po ostatniej kropce.

4. Znak cudzysłowu otwierającego położony jest na podstawowej linii pisma („). Jego kształt zależy od użytej czcionki.

5. Znak cudzysłowu zamykającego to identycznie zwrócony znak, ale położony na górnej linii pisma (").

6. Zgodnie z polskimi zasadami interpunkcji przecinki stawia się na zewnątrz cudzysłowów a kropkę po cudzysłowie zamykającym np.: „Ach, Misiu", powiedział Krzyś, „Tak bardzo Cię kocham".

7. Dwukropki i średniki umieszczamy zawsze po cudzysłowie zamykającym i zawsze wstawiamy po nich odstęp: Marek przeczytał w „Polityce": „Benzyna znowu będzie droższa"; potem okazało się, że jednak staniała.

8. Zgodnie z polskimi zasadami pisowni i interpunkcji w datach pisanych w całości cyframi arabskimi stawiamy kropki 16.09.2008, 2.8.1935 roku, 1.05.2006 r. Nie stawiamy kropki, gdy liczba miesiąca zapisana jest cyfrą rzymską 14 X 1988 r. Nie należy stosować innych sposobów oddzielania poszczególnych składników daty (wyjąwszy szczególne przypadki takie jak wymogi komputerowego przetwarzania danych). Nie należy też w języku polskim stosować innego szyku niż dzień, miesiąc, rok.

9. Zgodnie z Polską Normą PN-83/P-55366 *Zasady składania tekstów w języku polskim*, wielokropek należy składać bez świateł między kropkami i bez odstępu między wielokropkiem a wyrazem, do którego się odnosi.

10. Wykropkowaniem zastępuje się litery pominięte w wyrazach wulgarnych. Ilość kropek jest taka sama jak ilość pominiętych liter. Pozostawia się pierwszą literę wykropkowanego wyrazu, dopuszcza się także zostawienie ostatniej litery. „Ten pomysł jest do d . . . "

11. Po wykrzykniku stawia się normalny odstęp międzywyrazowy. Pomiędzy wykrzyknikiem a poprzedzającym go elementem tekstu nie wstawia się dodatkowych spacji.
12. Kropka jest stawiana po cudzysłowie zamykającym.
13. W przypadku zbiegu wielokropka z kropką, przecinkiem i średnikiem, znaki te opuszcza się.
14. W przypadku zbiegu wielokropka z wykrzyknikiem lub znakiem zapytania, znaków tych się nie opuszcza.

Dywiz, półpauza i pauza

Każdy font pisma dziełowego zawiera trzy rodzaje kresek poziomych: dywiz zwany też łącznikiem (-), półpauzę zwaną myślnikiem (–) i pauzę (—). Długość każdej z kresek nie jest ustalona i zależy od użytego fontu.

1. Dla wyodrębnienia wtrąceń używa się półpauzy(myślnika) ze zwykłymi odstępami po obu stronach (Acha – powiedział Jan – zgadza się.) Zasady polskiej interpunkcji dla składu komputerowego zalecają też stosowanie myślnika pomiędzy cyframi oznaczającymi wartość przybliżoną (2–3 razy w roku).
2. Myślnika używa się też w zestawieniach dwóch wyrazów o znaczeniach przeciwstawnych (czarny – biały, zimny – ciepły).
3. Łącznik (dywiz) stosujemy zapisując np. nazwiska dwuczłonowe np. Kowalska-Jastrząb, w złożeniach dwóch kolorów np. biało-czerwony (mający dwa kolory biały i czerwony). Nie wstawia się odstępu przed i po dywizie.
4. Łącznik (dywiz) stosuje się w wyrazach złożonych, w których pierwszej części występuje liczba (Ta 1-majowa manifestacja...)

Spis treści

W spisie treści wiersze niepełne można uzupełnić kropkowaniem. Odległości między kropkami mogą wynosić od pół firetu[2] do dwóch firetów. Odstęp między tekstem a pierwszą kropką nie może być mniejszy niż pół firetu. Odstęp pomiędzy ostatnią kropką a największą liczbą oznaczającą numer strony powinien być taki sam jak odstęp między kropkami. Kropki ustawia się w równej linii pionowej. Przy wielostopniowym podziale tekstu można nie podawać tytułów partii niższego stopnia.

[2]Firet – kwadrat o boku równym danemu stopniowi pisma

Spis tabel

2.1 Standardowe opcje klasy 23
2.2 Znaki specjalne 25
2.3 Akcenty 25
2.4 Stałe jednostki miary rozumiane przez TEX-a 31

3.1 Geometria strony 62

4.1 Instrukcje zmieniające odmiany czcionek 72
4.2 Predefiniowane wielkości czcionek 75
4.3 Wybrane języki programowania dostępne w pakiecie `listings` 120

5.1 Przykładowa tabelka z podpisem 136
5.2 Dopuszczalne parametry środowiska `table` 141
5.3 Liczniki kontrolujące miejsce umieszczenia obiektu 141
5.4 Parametry kontrolujące położenie obiektu na stronie 142

6.1 Rozmiary fontów matematycznych 146
6.2 Opcje pakietu `amsmath` 158

Skorowidz

\−, 131
\, 24
\+, 128, 131
\=, 125, 130
\>, 125, 130
&, 154

A

Acrobat Reader, 16
\addcontentsline, 50
akcenty, 24
amsthm pakiet, 173
Antykwa Półtawskiego, 37
Antykwa Toruńska, 36
autor, 47, 48

B

\baselineskip, 80
\baselinestretch, 80, 81
\begin{document}, 24
\bibitem, 51
\bigskip, 82

C

\centerline, 83
\chapter, 49
\cite, 51
CMYK, 181
cp1250, 27
Cyklop, 37

D

data, 48
\DeclareGraphicsExtensions, 190
\definecolor, 182
dviout, 16

E

edytor, 13
\end{document}, 24
\enlargethispage, 56
\evensidemargin, 47, 56

F

fantom, 93
\fbox, 114
\fcolorbox, 184
font, 32
\fontfamily, 38, 41
fontów
    rodzina, 33
\footnote, 99

\footnotemark, 100
\footnotetext, 100
\footrule, 59
\footskip, 47
\framebox, 114
G
GhostScript, 16
\graphicspath, 190
GSview, 16
H
\headheight, 47
\headrule, 59
\headsep, 47
\hline, 134
I
\include, 104
includefoot, 62
includehead, 62
\includegraphics, 188
indeks
    dolny, 149
    górny, 149
\input, 104
interlinia, 193
iwona, 34
K
\kill, 128
klamry, 150
klasa
    article, 20, 23, 45, 99
    book, 23, 45, 99
    dokumentu, 22
    exsizes, 61
    mwart, 23
    mwbk, 23
    mwrep, 23
    report, 23, 45, 99
kodowanie, 22
krój
    bezszeryfowy, 39, 40, 72, 194
    dwuelementowy, 33
    fontu, 33
    jednoelementowy, 33
    odmiana, 71
    pisma, 195
    szeryfowy, 39, 40, 72, 194
kurier, 35
L
\linespread, 80
lista opcji, 19, 23
listings pakiet, 118
\lstinline, 119
\lstinputlisting, 120
\lstset, 119
lustrzane odbicie, 187
M
macierze, 154, 168
majuskuły, 194

\makebox, 114
marginesy, 62
\markboth, 55
\markright, 55
\mbox, 114
mechanizm numerowania, 148
\medskip, 82
minuskuły, 194
\multicolumn, 136

N
nagłówek, 56, 60
nawiasy, 150
\newcommand, 21, 96, 98
\newenvironment, 21, 97, 98
\normalcolor, 184
\normalsize, 81
\notag, 159
notki na marginesie, 94
nowe polecenie, 96

O
obracanie, 186
\oddsidemargin, 47, 56
odstęp
    międzywyrazowy, 193
    pionowy, 79
    poziomy, 77
odsyłacze, 102
operatory, 151

P
\pageref, 103
pakiet, 21
    amsmath, 20, 103, 148, 156, 157
    amssymb, 148
    amsthm, 22, 148
    array, 133
    babel, 29
    chappg, 60
    fancyhdr, 46, 56
    geometry, 46
    graphicx, 185
    lastpage, 60
    polski, 28
    tabularx, 133
    theorem, 22
\paperwidth, 47
\parbox, 27, 116, 118, 133, 168
pierwiastki, 149
podkreślenia, 150
podrozdział, 48
podziękowania, 48
\poptabs, 130, 131
preambuła, 21, 22, 27
przypisy, 99
\pushtabs, 130, 131

R
\raggedleft, 83
\raggedright, 83

\raisebox, 115
\reflectbox, 188
\renewcommand, 21, 96, 100
\renewenvironment, 97
\resizebox, 188
RGB, 181
\rotate, 186
\rotatebox, 186, 189
rozdział, 28, 48

S
\scalebox, 187
\selectfont, 38, 41
\setcounter, 55, 99
\setlength, 31, 46, 80, 91, 113, 158
skalowanie, 187
skład
    w chorągiewkę, 192
    wyjustowany, 193
skorowidz, 52
\sloppy, 93
\smallskip, 82
spis
    literatury, 20, 48, 50
    rysunków, 50
    tabel, 50, 133
    treści, 49
stopka, 56, 60
streszczenie, 28, 48
strona kodowa, 28
styl strony, 54
symbol Newtona, 156
system kodowania, 27
szerokość tekstu, 62

Ś
środowisko
    abstract, 49
    align, 158
    alignat, 166
    aligned, 161
    array, 132, 154
    cases, 167
    center, 82, 133
    description, 105
    displaymath, 145, 148, 172
    enumerate, 105, 106
    eqnarray, 156
    eqnarray*, 156
    equation, 96, 101, 103, 148
    equation*, 148
    figure, 104
    flushleft, 83, 85
    flushright, 83
    gather, 160
    gathered, 161
    itemize, 105, 106
    list, 107
    lstlisting, 119
    minipage, 104, 115, 117

minipage , 99
multline, 165
nowe, 97
proof, 173, 176
sloppy, 93
split, 164
subequation, 172
tabbing, 125, 128, 129
table, 133, 135
tabular, 132, 136
tabular*, 132

T
\textcolor, 183
\textheight, 47
tgadventor, 39
tgbonum, 39
tgchorus, 39
tgcursor, 39
tgheros, 39
tgpagella, 39
tgschola, 39
tgtermes, 39
\thefootnote, 101
\thispagestyle, 55
\topmargin, 47
\topskip, 47
tryb
matematyczny, 145
tekstowy, 145
typograf, 191
typografia, 191
tytuł, 47, 48

U
ułamki, 147
\usepackage, 21, 22
utf8, 27, 28

V
\vfill, 81
\vline, 138
\vspace, 81
\vspace*, 82

W
windvi, 16
wysokość tekstu, 62
wyznaczniki, 154, 168
wzory
matematyczne, 146
wieloliniowe, 155
numerowane, 170

Z
znaki specjalne, 24

Bibliografia

[1] Robert Bringhurst, Elementarz stylu w typografii, Design Plus, Kraków 2007.

[2] Robert Chwałowski, Typografia typowej książki, Helion, 2002.

[3] Michael Goossens, Frank Mittelbach, Alexander Samarin, The LaTeX Companion – Second Edition, Addison-Wesley, 2005.

[4] Jobst Hoffmann, The Listings Package, `www.ctan.org/pkg/listings` .

[5] Andrzej Icha, LaTeX2e dla matematyków, Wydawnictwo Akademii Pomorskiej w Słupsku, Słupsk 2007.

[6] Donald E. Knuth, TeX. Przewodnik użytkownika, WNT, Warszawa 2005.

[7] Helmut Kopka, Patrick W. Daly, A Guide to LaTeX2e, Addison-Wesley, second edition 1997.

[8] Leslie Lamport, LaTeX. Podręcznik i przewodnik użytkownika, WNT, Warszawa 2004.

[9] Wojciech Myszka, Ewaryst Rafajłowicz, LaTeX zaawansowane narzędzia, Akademicka Oficyna Wydawnicza PLJ, Warszawa 1996.

[10] Piet van Oostrum, Page layout in LaTeX, dostępny na `www.ctan.org`

[11] Tomasz Przechlewski, Praca magisterska i dyplomowa z programem LaTeX, Wolters Kluwer Polska Sp. z o.o, 2011.

[12] Hideo Umeki, The Geometry Package, `www.ctan.org/pkg/geometry`

[13] Using the `amsthm` Package, AMS, www.ctan.org/pkg/amsthm

[14] Herbert Voß, Math mode – v. 2.47, `www.tex.ac.uk/tex-archive/info/math/.../mathmode/Mathmode.pdf`