LaTeX okiem praktykującego

Andrzej M. Borzyszkowski
Instytut Podstaw Informatyki PAN
ul. Abrahama 18, 81-825 Sopot
a.borzyszkowski@ipipan.gda.pl
www.ipipan.gda.pl/~andrzej

Streszczenie

Przedstawione tu zostaną podstawy LaTeX'a wymieszane z osobistymi nieraz uwagami na temat różnych zestawów makr. Część z zestawów jest używana od lat przez autora, inne rozwiązują wieloletnie problemy.

Basic info on LaTeX will be presented mixed with personal opinions on some LaTeX packages. Some packages have been used by the author for years, other solve old problems.

Podstawy

Ideologia System TeX służy do składu tekstu. Oznacza to, że tekst źródłowy pełen jest instrukcji formatujących. System LaTeX, nadbudowany nad TeX-em, kładzie nacisk raczej na strukturę logiczną dokumentu, a instrukcje formatujące schowane są w klasach i pakietach czytanych podczas kompilacji dokumentu. Rozdzielenie struktury logicznej i sposobu przedstawienia tejże struktury powoduje, że w miarę łatwo można zmienić wygląd dokumentu.

Przykładem niech będzie LaTeX-owe `\section`, parametrem tego makra jest tytuł rozdziału. Użycie tego makra niesie wiele skutków:

- tekst tytułu zostanie złożony odpowiednim fontem,
- zostaną zastosowane odstępy przed i po tytule rozdziału,
- rozdziałowi zostanie nadany kolejny numer,
- numer ten być może będzie użyty w numeracji podrozdziałów czy twierdzeń wewnątrz rozdziału,
- tytuł rozdziału powędruje do spisu treści,
- i być może wystąpią jeszcze inne zjawiska.

Co więcej, definicja tego wszystkiego zawarta jest w użytych klasach i pakietach. Zmiana kilku wierszy w pliku źródłowych pozwoli na przesłanie artykułu do innego wydawnictwa stosującego własne zasady składu.

Główne składniki tekstu LaTeX-owego Tekst źródłowy tego artykułu mógłby[1] zaczynać się od słów

```
\documentclass[twocolumn]{article}
\usepackage{polski}
```

Każdy dokument ma przypisaną swoją klasę. Podstawowe klasy LaTeX-a to `article`, `report`, `book` oraz `letter`. Ta ostatnia klasa różni się mocno od pozostałych – służy do składu listów, a nie tekstów naukowych. Jest również mniej przydatna dla Polaków, powodem jest duża rozbieżność pomiędzy polskimi a anglosaskimi zwyczajami typograficznymi.

Pozostałe klasy są w miarę podobne, wszystkie służą do składu tektów naukowych, zaczynając od najskromniejszych, a kończąc na książkach. W codziennej pracy naukowca najczęściej używaną klasą jest `article`. Można też stosować klasy niestandardowe, np. klasę `llncs`[2] przygotowaną przez wydawnictwo Springera do składu bardzo popularnych sprawozdań konferencyjnych z dziedziny informatyki (dosłownie setki tomów po kilkadziesiąt artykułów rocznie). Klasy posiadają swoje opcje, w przykładzie mamy opcję składu dwułamowego.

Dokument LaTeX-owy prawdopodobnie będzie używać kilku pakietów. Są to biblioteki makr przydatne w danym zastosowaniu. Pakiet `polski` jest pożyteczny przy pisaniu tekstu po polsku – polskie słowa kluczowe, odpowiednia strona kodowa używana przez TeX-a, ... Odrębnym problemem jest samo kodowanie polskich liter w pliku źródłowym. Jednym ze stosowanych rozwiązań jest użycie pakietu `inputenc` z opcją wyjaśniającą użyte kodowanie. Innym jest użycie specjalnego wiersza na początku pliku, tutaj

```
%&platex --translate-file=il2-pl
```

[1] gdyby nie to, że pisany jest w specjalnym formacie wymaganym przez komitet programowy naszej konferencji.

[2] ftp.springer.de/pub/tex/latex/llncs/latex2e/

To rozwiązanie jest swojego rodzaju nowością, nie jest dostępne we wszystkich implementacjach. Dopiero takie rozwiązanie zapewnia jednolite przekodowanie polskich liter we wszystkich plikach pomocniczych, o których dalej. W przykładzie mamy też inne zastosowanie: wymuszenie formatu TEX-owego. Utrudni to kompilację w systemie nie posiadającym `platex`'a, ułatwi używanie edytora atomatycznie wywołującego TEX-a na swoim buforze.

Kolejną różnicą pomiędzy LATEX-em a TEX-em jest pojęcie środowiska. O ile drobne fragmenty tekstu bez problemu są obsługiwane przez TEX-owy mechanizm grupowania za pomocą nawiasów, o tyle dłuższy tekst wygodniej grupować za pomocą jawnych instrukcji początku i końca. W szczególności cały tekst przeznaczony do złożenia zawarty jest w środowisku `document`

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

LATEX sprawdza poprawność ustawienia takich „nawiasów".

Pomiędzy deklaracją klasy a początkiem głównego środowiska, czyli w preambule, nie powinno być w ogóle nic, co generuje powstanie tekstu. Jest to miejsce wyłącznie na deklaracje pakietów, definicje makr i środowisk oraz użycie pewnych specjalnych makr nie skutkujących powstaniem tekstu.

Mechanizmy LATEX-a LATEX ma inny niż TEX mechanizm definiowania nowych makr i nowych środowisk. Zamiast TEX-owego

```
\def\makro#1#2{#2.#1}
```

mamy

```
\newcommand\makro[2]{#2.#1}
```

Jak widać z podanego przykładu mechanizm LATEX-a jest słabszy niż oryginalny. Czego nie widać, to wygodnego sprawdzianu, czy makro `\makro` jest już zdefiniowane. LATEX robi to automatycznie, powtórne definiowanie makra już zdefiniowanego wymaga jawnego użycia instrukcji `\renewcommand` zwiększając dyscyplinę programowania. Oszukiwać się nie da, LATEX narzeka również na nieuzasadnione użycie `\renewcommand`. Definicja nowego środowiska, `\newenvironment`, wymaga podania *dwóch* parametrów, definicji początku oraz definicji końca środowiska. Środowisko też może posiadać swoje parametry, niestety mogą one być użyte wyłącznie w pierwszej części, na początku. Ale nikt nie broni zapamiętać parametru na początku, `\sbox{arg}{#1}`, i użyć go na końcu, `\usebox{arg}`, gdzie pudełko musi być zawczasu przygotowane, `\newsavebox{arg}`.

Makra i środowiska zdefiniowane w LATEX-u posiadają często możliwość podania argumentu opcjonalnego. Argument taki podawany jest nawiasach kwadratowych, przykładem niech będzie

```
\parbox[t]{2cm}{tekst}
```

służący do składu tekstu w akapicie o podanej szerokości, opcjonalny parametr podaje sposób pionowego wyrównania otrzymanego pudełka względem reszty tekstu. Inny przykład wystąpił przed chwilą, `\newcommand` dopuszcza argument opcjonalny z domyślną wartością 0. Użytkownik też może definiować makra i środowiska z parametrem opcjonalnym:

```
\newcommand\tuple[2][n]{(#2_1,\ldots,#2_{#1})}
```

Pierwszy parametr do tak zdefiniowanego makra czy środowiska powinien być podawany w nawiasach kwadratowych, w razie jego braku zostanie przyjęta wartość domyślna podana w drugiej parze nawiasów kwadratowych w definicji. Jak metodami LATEX-a zdefiniować makro z kilkoma parametrami opcjonalnymi, jak spowodować by parametr opcjonalny był np. drugim, albo miał domyślnie wartość innego – nie wiem.

Standardowe klasy LATEX-a posiadają mechanizm automatycznego numerowania nie tylko stron ale i rozdziałów, twierdzeń, tabel itd. Użytkownik może zechcieć zmienić wartość któregoś z liczników: `\setcounter{section}{5}`, może też zechcieć posiadać swój własny licznik: `\newcounter{licznik}`. Stan licznika można drukować w różnych formatach, `\arabic`, `\roman`, `\Alph`, a strona tytułowa zawiera często przypisy „numerowane" śmiesznymi symbolami, †, ‡, §, ¶, otrzymane są one przy użyciu `\fnsymbol` (tylko dla wartości od 1 do 9).

Podkreślić należy, że LATEX pozwala na korzystanie z całego dobrodziejstwa gołego TEX-a. Na przykład, można używać TEX-owego `\def`, można korzystać z pakietów przygotowanych do współpracy z formatem plain. Twórcy LATEX 2ε zachęcają jednak do używania mechanizmów LATEX-a, nie TEX-a.

Implementacje Z powodów historycznych wspomnieć należy o oryginalnym LATEX-u, znanym obecnie jako LATEX 2.09. Obecnie zaleca się używanie LATEX 2ε. Wersja ta o wiele łatwiej poddaje się polonizacji i posiada całkowicie odmiennie zaimplemetowane metody wyboru fontów. Również mechanizm pakietów wspomniany powyżej należy do LATEX 2ε. Z drugiej strony, przeciętny użytkownik może bez problemu posługiwać się klasycznym podręcznikiem Lamporta, twórcy LATEX-a, [Lam86] – zasady LATEX-a nie zmieniły się. Podręcznik [GMS93] jest znacznie obszerniejszy niż poprzedni i obejmuje wiele znanych pakietów. Dotyczy on LATEX 2ε i już zakłada znajomość LATEX-a, można go uznać za uzupełnienie [Lam86]. Skutecznym wprowadzeniem, i bardziej dostępnym, jest [OPHS].

Zestawy makr składające się na LATEX-a znajdują się w katalogu `texmf/tex/latex/base`, głównie w pliku `latex.ltx`, ale również w plikach dla klas, np. `book.cls` i ich opcji, np. `bk12.clo`. Wspomnijmy już tutaj o pewnym szczególe implementacji LaTeX-a. Otóż wiele makr LaTeX-owych zawiera w swojej nazwie znak `@`. Normalnie nie ma on prawa pojawiać się w nazwie makr i właśnie dlatego używany jest w implementacji. Zwykły użytkownik nie ma szans na przypadkową zmianę standardowych definicji. Użytkownik świadomy ma szansę na taką zmianę, jeśli fragment kodu, w którym występują takie specjalne nazwy, otoczony jest instrukcjami

```
\makeatletter
\makeatother
```

Można więc znaleźć interesujący nas fragment kodu źródłowego, skopiować go do naszego dokumentu i odpowiednio zmodyfikować. Można też zmodyfikowany kod zapisać w pliku `mojstyl.sty` i włączyć go za pomocą `\usepackage`, wówczas również `@` jest dopuszczalne w nazwach.

Elementy LaTeX-a ogólnego użytku

Układ strony Standardowe klasy LaTeX-a mają wstępnie zdefiniowane wymiary strony. I nie są to tylko dwa wymiary, jest ich cała gama. Oryginalne definicje są mało przydatne w praktyce, użytkownik standardowych klas musi zadbać o własne definicje. Np.

```
\setlength\textheight{23.5cm}
\setlength\textwidth{16cm}
```

Warto/trzeba zadbać też o `\headheight`, `\headsep`, `\footskip` i inne. Jedną z możliwości uniknięcia tej żmudnej pracy jest użycie jakiegoś dostępnego stylu definiującego wymiary, np. `a4wide`. Również korzystając z klasy przygotowanej przez jakieś wydawnictwo nie musimy na ogół zajmować się tymi szczegółami – makra wydawnictw przede wszystkim zajmują się wymiarami strony. W tym ostatnim przypadku warto sobie uzmysłowić możliwość małego oszustwa: jeśli mamy problemy ze zmieszczeniem się w limicie stron, można spróbować:

```
\addtolength\textheight{2mm}
```

Jest duża szansa, że nikt nie zauważy, a może to rozwiązać problem dodatkowej pozycji bibliograficznej dołączonej w ostatniej chwili. Również rozkład materiału ruchomego, patrz dalej, może się radykalnie zmienić przy zaskakująco małych zmianach wymiaru.

Każdy dokument LaTeX-owy musi posiadać swój styl paginacji, `\pagestyle`, domyślną wartością jest `plain` – zwykłe liczby pośrodku stopki. Żywą paginę w nagłówkach zapewnia styl `headings`, sama pagina zdefiniowana jest w użytej klasie. Użytkownik ma pewne możliwości osobistego zdefiniowania kształtu paginy:

```
\pagestyle{myheadings}
\markright{\latex{}\hrulefill\today\hrulefill}
```

Jednak próby uzyskania niestandardowej paginy są skazane na niepowodzenie, np. nie można mieć bieżącego tytułu w nagłówku, a numeru strony w stopce. Wymyślniejszą paginę można otrzymać używając pakietu `fancyhdr`.

LaTeX generalnie rozróżnia pomiędzy składem przeznaczonym do druku jednostronnego a składem dwustronnym. W tym ostatnim występuje pojęcie strony prawej i lewej czyli nieparzystej i parzystej. Np. używając opcji `twoside` klasy `article` lub klasy `book` należałoby w powyższym przykładzie użyć `\markboth` z dwoma parametrami.

Pakietem użytecznym do definiowania wymiarów strony jest `geometry`.

Zmiana kroju pisma To akurat zjawisko stoi trochę w poprzek ogólnej ideologii LaTeX-a nie ingerowania bezpośrednio w wygląd zewnętrzny. Jednak nieraz zachodzi potrzeba lokalnej zmiany używanego kroju pisma. Oryginalny LaTeX proponował dwie metody. W celu wytłuszczenia niewielkiego tekstu należało użyć makra `\bf`, być może tworząc z tej okazji grupę. Dłuższy tekst, należało zawrzeć w środowisku o nazwie `bf`. Mały tekst należy obecnie wytłuszczać jako `\textbf{tekst}` a wyróżniać jako `\emph{tekst}`. Jednak poprzednia metoda też działa. Należy pamiętać o wyższości `\emph` nad `\textit`. W pierwszym przypadku tekst będzie złożony kursywą w otoczeniu kroju prostego, ale jeśli zdarzy się nam użyć tego wyróżnika w środowisku składanym kursywą, wówczas wyróżni się on właśnie prostotą. Drugie makro powoduje bezwarunkowy skład kursywą, bez względu na sąsiedztwo. `\emph` dba również o poprawkę na pochyłość kursywy, w LaTeX-u 2.09 należało pamiętać by pisać `{\em ..\/}` chyba, że zaraz potem był znak przestankowy.

Przy tej okazji można powtórzyć apel jeszcze Knuth'a o rozróżnienie pomiędzy `\mathit{fifka}` fifka a `fifka` $fifka$ w środowisku matematycznym. Wielu autorów nie przestrzega tej zasady. W środowisku matematycznym w zasadzie nie zmienia się kroju pisma. Jeśli tekst miałby być wytłuszczony, to *na zewnątrz* tego środowiska powinna znaleźć się deklaracja `\mathbold`. Za zmianę kroju trudno uznać użycie `\mathcal`, służy to raczej do otrzymania pojedynczych liter kaligraficznych (tylko wielkie litery) dobrych na wymyślne oznaczenia. Podobnie `\mathbf`, choć tu nie ma tego ograniczenia.

Specjalnym krojem pisma jest pismo maszynowe, `tt`. Używa się go często w kontekście dosłownego cytowania kodu, np. LaTeX-owego. Wówczas `\texttt` nie wystarcza, pojedyncze słowa można skutecznie obsłużyć konstrukcją `\verb|..|` gdzie zamiast pałki można użyć dowolnego znaku[3], byle do pary. Dłuższy tekst powinien być zawarty w środowisku `verbatim`.

Innym efektem jest zmiana rozmiaru pisma. Niewielki tekst można zawrzeć w grupie razem z instrukcją np. `\large`, większy tekst w środowisku o tej nazwie. Przy tej okazji uwidacznia się wyższość LaTeX 2_ε, kombinacja `\bf\large` da pismo wytłuszczone i powiększone, identycznie jak użycie przeciwnej kolejności. W LaTeX-u 2.09 tylko jedna z kolejności daje pożądany efekt. Deklaracja rozmiaru zmienia nie tylko sam rozmiar czcionek, ale również różne odstępy. Liczy się deklaracja rozmiaru aktualna w momencie zakończenia akapitu.

Rozmiary typu `\large` są względne. Rozmiary bezwględne są zdefiniowane w definicji klasy lub opcji do klasy. Np. standardowa klasa `article` używa domyślnie fontu 10 pt, klasa ta wywołana z opcją `11pt` spowoduje, że normalnym rozmiarem będzie 11 pt, a `\small` będzie powodować skład dziesięciopunktowy. Taka implementacja jest zgodna z duchem LaTeX-a i powoduje łatwość przejścia pomiędzy różnymi możliwymi stylami.

Wyróżnienia Środowiska `itemize` i `enumerate` służą do wyróżniania wyliczeń. W zasadzie pierwszego nie należałoby nazywać wyliczeniem, bo nie ma w nim numeracji, w html'u odpowiednikiem jest `<ul>`, unordered list. Każda z pozycji w wyliczance poprzedzana jest makrem `\item`, być może z opcją symbolu podanego jawnie, np. `\item[--]`. Standardowe klasy LaTeX-a zawierają definicje domyślnych symboli wyróżniających pozycje, odrębnych dla każdego z czterech poziomów zagnieżdżeń. Jeśli etykiety pozycji mają być dłuższe, należy użyć środowiska `description`.

Sprawa doboru tych symboli, szczególnie w wersji nienumerowanej, a również odstępów pomiędzy pozycjami, od dawna budzi wątpliwości użytkowników. Polska wersja klas, zapowiadana właśnie na ten Bachotek, będzie miała na pewno inne symbole. Użytkownicy chcący zmian już teraz mogą zdefiniować sobie własne etykiety:

```
\renewcommand\labelitemi{--}
```

i podobnie dla dalszych poziomów zagnieżdżeń. Bardziej radykalne zmiany wymagają nowego środowiska. Posiłkować się można środowiskiem `list` o bagatej liczbie możliwych odstępów i możliwościach definiowania własnych symboli dla pozycji:

```
\newcounter{amb}
\newenvironment{ambenum}[1][\S]
{\begin{list}{#1 \arabic{amb}.}
              {\usecounter{amb}\itemsep 0pt}}
\end{list}}
```

[3] z wyjątkiem `*`, bo ona daje wersję gwiazdkowaną makra.

Znacznie prostszą metodą wyróżnienia tekstu są środowiska `center` i `quote`, być może z dodatkową deklaracją zmiany kroju pisma na początku. Zmianę wiersza można wymuszać za pomocą `\\`.

Do wyróżniania fragmentów matematyki LaTeX proponuje środowiska `equation` oraz `displaymath`. W pierwszym otrzymujemy numerowany wzór, opcja klasy `fleqn` określa, że wzory będą wyrównane do lewej (z szerokim marginesem), zaś `leqno`, że numeracja pojawi się po lewej. Należy zwracać uwagę, by środowiska tego nie poprzedzać wolną linijką, następuje wówczas niepotrzebne dodawanie pionowych odstępów. W drugim przypadku wzór będzie nienumerowany i zawsze wyśrodkowany. W praktyce granice środowiska `displaymath` zapisywane są albo za pomocą podwójnych dolarów, albo `\[..\]`. Pierwszej metody, choć jest popularna, należy unikać. Wiele edytorów jest w stanie dokonać analizy ustawienia nawiasów, ale muszą one tworzyć parę. Poza tym wprowadzając podwójny znak dolara łatwo o pomyłkę. Wspomnijmy, że tekst matematyczny bez wyróżniania otrzymuje się za pomocą pojedynczych dolarów, `\(..\)` albo środowiska `math`.

Pudełka Są podstawowym pojęciem dla TeX-a. LaTeX posiada własne operacje na pudełkach, twórcy LaTeX 2_ε zachęcają do ich używania zamiast prymitywów TeX-owych. O nieczęstych operacjach już wspomnieliśmy – do tymczasowego przechowania materiału. Ale w codziennej praktyce spotyka się raczej zwykłe pudełka, `\parbox` zawiera cały akapit, `\mbox` gwarantuje, że fragment tekstu nie będzie dzielony pomiędzy wierszami, `\makebox[2cm][c]` gwarantuje stały wymiar, oba pudełka mają wersje z ramką, a `\raisebox` służy do pionowego przemieszczania materiału, np.

```
\newcommand\quotient[2]%
  {\raisebox{.3ex}{$#1$}/\raisebox{-.6ex}{$#2$}}
```

pozwala wygodnie złożyć $X/_\equiv$. Przy większych przesunięciach można mieć problem z pionowymi odstępami, można wówczas rozważyć objęcia przesunięciami również `\strut`'a. Jest to pionowe pudełko o zerowej szerokości, a więc niewidoczne, i wysokości równej `\baselineskip`. Wówczas odstęp pomiędzy wierszami będzie większy.

Definiując makra warto czasem celowo włożyć argument do pudełka z matematyką, `\mbox{$#1$}`, gwarantując, że będzie on potraktowany matematycznie bez względu na okoliczności zewnętrzne. Metoda ta ma jednak wadę, tekst w takim pudełku nie podlega zmianom rozmiaru liter w zależności od okoliczności. LaTeX 2_ε proponuje zamiast tego `\ensuremath{#1}`. Do dziś jednak widuje się wystąpienia indeksów dolnych i górnych wielkości głównego tekstu. Prawdopodobnie autorzy używali makr z pudełkami w starym stylu.

Oczywiście, zwykły tekst w środowisku matematycznym powinien znaleźć się w pudełku.

Tabele Tabele w LaTeX-u składają się z deklaracji wzorca dla kolumn i z samej zawartości komórek. Przykładowe środowisko do tabel zaczyna się następująco

```
\begin{tabular}[t]{||lc@{:\quad}r|p{2cm}||}
```

gdzie litery `l`, `r` oraz `c` oznaczają zwykłe kolumny wyrównywane do lewej, prawej czy centrowane, `p{2cm}` oznacza, że tekst będzie złożony tak jakby był argumentem do `\parbox[t]{2cm}`, `@{:\quad}` deklaruje zawartość dodatkowej kolumny, pozbawionej domyślnych odstępów od sąsiednich kolumn, a pionowe kreski służą do kreślenia pionowych linii. Parametr opcjonalny określa sposób pionowego wyrównania całej tabeli, domyślnie jest centrowana.

Zawartość tabeli składa się z wierszy oddzielonych za pomocą `\\`[4], być może z dodatkowym odstępem pionowym `\\[.5cm]`, każdy wiersz składa się z komórek oddzielanych znakiem `&`. Komórki mogą być puste, może być ich w wierszu mniej niż przewiduje wzorzec, ale nie więcej. Mniejsza liczba komórek w wierszu spowoduje, że reszta wiersza będzie pusta. W prostym przypadku daje to pożądany efekt. Jednak w podanym wyżej przykładzie należałoby jawnie użyć pustych komórek tak, by wymusić pojawienie się pozostałych elementów wzorca: pionowych kresek czy dwukropka. W szczególności, tylko w najprostszych tabelach można bezpiecznie używać podwójnego znaku `\\` do utworzenia pustego wiersza. Wierszem tabeli może być też linia pozioma, `\hline`. Jeśli zachodzi konieczność połączenia kilku komórek w którymś z wierszy, do dyspozycji mamy

```
\multicolumn{2}{||c@{:}}{dwie komórki}
```

Należy podać sposób wyrównania takiej wielokomórkowej zawartości i pamiętać o dodaniu dodatkowej kolumny czy pionowych kresek, jeśli takie występują we wzorcu po prawej stronie zastępowanych kolumn lub na lewym skraju tabeli.

Jednym z zastosowań wzorca `@` jest wyrównanie tabeli do lewego marginesu, np.

```
\begin{tabular}{@{}llll}
```

Bez tego lewa kolumna tabeli miałaby niezerowy odstęp po lewej, akurat dobry przy nawiasach czy pionowych liniach, ale niepożądany, jeśli ich brak.

Wzorzec `p` odpowiada za skład komórki w wąskim pudełku. W tabelach pudełko może być tak wąskie, że zajdzie potrzeba przeniesienia już w pierwszym wyrazie, rzecz w TeX-u niesłychana. Aby oszukać TeX-a należy mu podłożyć fałszywy „pierwszy wyraz”: `\hspace{0pt}`. Przy pomocy pakietu `array` można to zrobić systematycznie.

Inna sytuacja oszukiwania TeX-a związana jest wyrównywaniem zawartości kolumny np. względem kropki dziesiętnej. W przykładzie

```
\begin{tabular}{|r@{\hspace{2em}}|}
999\\
  9\makebox[0pt][l]{.99}\\
```

części ułamkowe liczb udają, że ich nie ma, więc wyrównanie komórek do prawej wyrównuje części całkowite liczb. Z drugiej strony wzorzec kolumn przewiduje wolne miejsce z prawej strony kolumny liczb w sam raz na dwie cyfry.

Odpowiednikiem tabel w środowisku matematycznym jest `array`, jedyną różnicą jest fakt, iż zawartość komórek jest składana w trybie matematycznym.

Pożytecznym parametrem dotyczącym tabel jest `\arraystretch`. Wbrew nazwie dotyczy również tabel ze zwykłym tekstem. Parametr ten posiada domyślnie wartość 1, jego zmiana dokonywana za pomocą `\renewcommand` powoduje rozsunięcie się wierszy tabeli. Jednak wartość poniżej 1 niczego nie zmieni. Z odstępem pomiędzy wierszami tabeli związany jest jeszcze jeden problem. Otóż czasami jedna z komórek ma zawartość złożoną większymi czcionkami, np. `\Large`. Ale w takim kontekście odstęp pomiędzy wierszami jest zwiększany tylko do niezbędnego minimum, tak by uniknąć nachodzenie się liter na siebie. Jeśli uważamy, że nie jest to ładnie, należy zawartość komórki wzbogacić o makro `\strut`. Wówczas odstęp pomiędzy wierszami tabeli będzie odpowiedni dla tej najwyższej komórki.

Należy pamiętać, że środowisko `tabular` nie powoduje żadnych skutków związanych z układem materiału na stronie, wynik produkowany przez to środowisko może być traktowany tak jak pojedyncza litera. Np. może być częścią komórki innej tabeli. Jeśli intencją było wyróżnienie tabeli, należy o to zadbać osobno.

Obiekty ruchome Są to obiekty na tyle duże, że ryzykowne byłoby pozwolić na pozostawienie wolnego tylko dlatego, że obiekt się nie mieści w pozostającym na stronie miejscu. Standardowe klasy LaTeX-a przewidują dwa takie środowiska, `figure` i `table`, inne klasy czy pakiety mogą definiować dalsze.

W czasie kompilacji dokumentu LaTeX stara się umieścić obiekt ruchomy na bieżącej stronie. Jeśli nie ma już miejsca, to obiekt dokładany jest do listy zaległości, która będzie obsłużona po zakończeniu strony. Jeśli takie zaległości powstaną, to dalsze obiekty, nawet niewielkie, będą dokładane do listy

[4] oddzielonych, ale nie zakończonych, tzn. ostatni wiersz nie może się kończyć podwójnym w-tył-ciachem.

– kolejność musi być zachowana. Oba środowiska tworzą swoje własne listy zaległości, tak więc tabela może wyprzedzić rysunek zamieszczony po niej, i na odwrót.

Dokładne miejsce umieszczenia obiektu ruchomego zależy od klasy. Standardowe klasy celują na górę strony, ale np. klasa `llncs` stara się umieścić rysunek w jego naturalnym miejscu. Środowiska `figure` i `table` dopuszczają parametr opcjonalny mówiący o naszych preferencjach położenia rysunku, można zażądać umieszczenia w miejscu wystąpienia, na górze lub dole strony, lub odłożyć rysunek na osobną stronę na później. Jeśli takie właśnie były nasze preferencje, parametrem będzie `[htbp]`. Pamiętać należy, że pierwsze użycie opcji `[p]` spowoduje, że dalsze rysunki czy tabele też zastaną odłożone na później – kolejka obowiązuje. Już pierwszy rysunek w kolejce może wypełnić osobną stronę, może się jednak zdarzyć, że i kilka będzie za mało. Klasa `book` przewiduje, że odłożone rysunki i tabele są drukowane najpóźniej na końcu rozdziału, w pozostałych klasach mogą czekać końca dokumentu. `\clearpage`, w przeciwieństwie do `\newpage`, powoduje natychmiastowe zakończenie bieżącej strony *oraz* natychmiastowy wydruk wszelkich zaległości.

Skład dwułamowy dodatkowo komplikuje sprawę. Pojedynczy łam może być za wąski dla większego, z definicji, obiektu. Dlatego istnieją wersje gwiazdkowane tych środowisk[5]. Ich użycie powoduje, że rysunek zostanie odłożony na listę obiektów oczekujących i będzie złożony na całej szerokości następnej strony. Dzieje się tak nawet, jeśli wymiary pionowe rysunku nie uzasadniają odkładania na później. Co gorsza, osobna strona z rysunkami szerokimi nie może zawierać rysunków jednołamowych nawet, gdyby kolejność była zachowana. A już przeplatanie tych dwóch klas rysunków ma zdecydowanie niekorzystne konsekwencje.

Obiekty ruchome mogą powodować duże frustracje autora. Niewielkie zmniany w gotowym prawie tekście mogą spowodować radykalną zmianę wyglądu całości. W szczególności, zmniejszenie tekstu źródłowego może skutkować wydłużeniem składu – rysunek, który dotychczas ładnie się komponował z tekstem, ląduje nagle na osobnej stronie. LaTeX posiada kilkanaście parametrów do sterowania układem rysunków: maksymalna liczba rysunków na stronie czy części strony, minimalna ilość tekstu, który może się znaleźć na stronie odległości pomiędzy rysunkami a tekstem. Jednak wieloletnia praktyka pokazuje, że trudno za ich pomocą uzyskać pożądane rezultaty. Autor musi się liczyć z koniecznością ręcznego przestawiania rysunków w tekście. A to z kolei będzie łatwiejsze, jeśli obiekty ruchome nie będą częścią głównego tekstu, a będą włączane za pomocą `\input`.

Przy okazji omawiania obiektów ruchomych wspomnijmy o środowisku `minipage`. Jest szczególnie przydatne gdy chcemy dokonać rozkładu materiału w dużym obiekcie ruchomym. Np., powtarzając dwukrotnie konstrukcję

```
\begin{minipage}[t]{0.45\textwidth}\centering
  ........
\caption{...}\end{minipage}
```

możemy umieścić w środowisku `figure` dwa rysunki. `\caption` użyte tak jak w przykładzie wygeneruje osobny podpis pod każdym z rysunków. Bardziej złożone zachowanie, osobne podpisy pod rysunkami i wspólny pod całością, zapewnia pakiet `subfigure`.

Diagramy i grafika Jest to raczej słaby punkt LaTeX-a – służy on przecież do składania tekstów, a nie do tworzenia rysunków. Dodatkowym powodem słabej obsługi grafiki jest brak założenia, że ostateczny wydruk będzie wykonywany za pomocą postscriptu.

LaTeX posiada wbudowane środowisko `picture`. Jego składnia odbiega zaskakująco od zwyczajów LaTeX-owych, parametry podaje się w nawiasach okrągłych. Jednym z nich jest para liczb – wymiary rysunku. Drugim, opcjonalnym, położenie początku układu współrzędnych. Jednostką miary jest `\unitlength`, jego zmiana musi być dokonana na zewnątrz środowiska, domyślnie ma wartość 1 pt. Jedyną instrukcją, którą można używać wewnątrz środowiska jest `\put(x,y){..}`[6] do położenia w zadanym położeniu dowolnych obiektów LaTeX-owych, choćby innych `picture`. W środowisku tym jednak pudełka są inaczej zdefiniowane, np. `\makebox` ma jako parametr *parę* liczb określających jego wymiary (w jednostce `\unitlength`), a parametr opcjonalny wskazuje nie tylko na wyrównanie poziome, ale i pionowe. Tymi metodami możemy precyzyjnie rozmieścić na płaszczyźnie tekst. Obiektami graficznymi są proste, okręgi i „owale”. Są one zaimplementowane w postaci znaków w specjalnym foncie, skutkiem tego proste dostępne są tylko w kilku predefiniowanych nachyleniach, a okręgi są przybliżane okręgami dostępnymi w foncie. Do rysowania krzywych służą krzywe Beziera drugiego stopnia, `\qbezier`. Są one składane z ogromniej liczby małych kropek. Ich implementacja pozostawia wiele do życzenia. Jednym z powodów są ogromne wymagania obliczeniowe, innym jest wyraźna różnica gęstości w środku i na koń-

[5] pakiet `multicol` nie dopuszcza w ogóle zwykłych wersji środowisk ruchomych.

[6] istnieje też wersja wielokrotna, nigdy jej nie użyłem.

cach rysowanej krzywej, tak było przynajmniej we wcześniejszych implementacjach.

Używając środowiska `picture` można osiągnąć wiele, np. można narysować diagramy przemienne albo struktury drzewiaste. Jest to jednak żmudne zajęcie, wymaga obliczania w pamięci[7] współrzędnych obiektów, a utworzone dokumenty będzie trudno zmodyfikować.

Prawidłowym rozwiązaniem jest stosowanie narzędzi odpowiednich do zadania. Np. prawdziwą grafikę można tworzyć przez zewnętrzny program i eksportować w formacie postscriptowym rozumianym przez LaTeX-a. Alternatywę daje pakiet `pstricks` – tutaj operuje się na postscripcie z poziomu LaTeX-a, w szczególności łatwo mieszać rozwiązania czysto postscriptowe z TeX-owymi. Istnieje też wiele pakietów wyspecjalizowanych w kierunku tworzenia diagramów, np. `xypic`.

Typowy artykuł naukowy

Ponieważ LaTeX został stworzony głównie w celu ułatwienia pisania artykułów naukowych, zawiera on kilka narzędzi temu pomocnych:

- strona tytułowa,
- podział artykułu,
- twierdzenia, itp.,
- odwołania do numeracji,
- środowisko do bibliografii,
- spis treści itp.,
- i inne.

Tytuł całości Tytuł artykułu powstaje wskutek użycia `\maketitle`. Jednak informacja niezbędna to utworzenia tytułu zostaje zgromadzona wcześniej, formalnie jako parametr do odpowiednich makr: `\author`, `\title`, i inne. Najczęściej umieszcza się je w preambule, choć nie jest to obowiązkowe. Standardowe klasy LaTeX-a przewidują, że tytuł składa się z tytułu sensu stricto, autora oraz daty. Klasy rozprowadzane przez wydawców czasopism zawierają na ogół więcej danych, co najmniej afiliację autora z jakąś formą adresu. Do pewnego stopnia można oczywiście oszukiwać, nikt nie zabrania zdefiniować autora następująco

```
\author{Andrzej M.\ Borzyszkowski\\[1ex]
  Instytut Podstaw Informatyki PAN}
```

Nie jest to jednak eleganckie rozwiązanie – trzeba zadbać o szczegóły formatowania, w tym przykładzie zmiana wiersza i odstęp pionowy, w miejscu do tygo nieprzeznaczonym. Rozwiązanie to stanie się niedopuszczalnym, jeśli informacja o autorze ma być użyta jeszcze w inny sposób, np. w bieżącej paginie czy w indeksie autorów w dziele zbiorowym. Rozwiązaniem zawsze skutecznym, ale i rozpaczliwym jest wykorzystanie środowiska `titlepage`. Wówczas jesteśmy po prostu wyłącznymi autorami strony tytułowej. Jest to odrębna strona, nawet jeśli używamy klasy `article`, w której tytuł domyślnie umieszczony jest razem z tekstem.

Nieoczywiste jest postępowanie w przypadku większej liczby autorów. Standardowe klasy LaTeX-a przewidują tylko jednego, być może w postaci kilku nazwisk oddzielonych `\and`. Jest to skuteczne rozwiązanie tylko, jeśli nie ma wątpliwości co do afiliacji różnych autorów. Inne klasy, np. `ltugproc` używana w tej chwili, przewidują powtarzanie całego bloku informacji. Też nie najlepiej – część autorów może być z tej samej instytucji. Najlepiej problem rozwiązuje chyba klasa `llncs`, piszemy tam

```
\author{Raz\inst{1}, Dwa\inst{1} i~Trzy\inst{2}}
\institute{Jeden\and Drugi}
```

i nazwiska autorów i adresy zastaną powiązane odpowiednimi numerami.

Wydaje się, że sposób tworzenia tytułu jest słabym punktem standardowych klas LaTeX-a. Ambitniejszy użytkownik powinien bez skrupułów brać się za modyfikację odpowiadających za to makr.

Instrukcje podziału Podział artykułu dokonywany jest za pomocą instrukcji `\part`, dalej `\section`, `\subsection`, `\paragraph`, a w klasie `book` mamy też dodatkowo `\chapter`. W podstawowych klasach LaTeX-a części te są numerowane, tytuł poprzedzany jest numerem *bez kropki*. Brak kropki razi niezwykle polskich użytkowników LaTeX-a. Jednym ze sposobów zaradzenia jest poczekanie na w pełni polskie klasy dokumentów, innym sposobem jest zmiana wewnątrz-LaTeX-owego makra odpowiedzialnego za wyświetlanie licznika wszelkich rozdziałów i podrozdziałów, np.

```
\def\@seccntformat#1{\csname the#1\endcsname.\ }
```

Makra podziału dokumentu są zdefiniowane przy użyciu `\@startsection`, z wyjątkiem najgrubszych. Użytkownik, którego rażą zbyt duże odstępy w tytułach lub zbyt duży rozmiar samych tytułów również może zaingerować w ich definicje.

Elementy podziału artykułu są numerowane na wyższych poziomach, np. rozdział, i nienumerowane na poziomach niższych. Ustalenie poziomu numeracji przewidziono dla zwykłych użytkowników: ustawienie licznika `secnumdepth` na 5 spowoduje numerowanie nawet akapitów. Brak wszelkiej numeracji wymaga ustawienia licznika na -2 – najgrubszy poziom podziału w klasie `book` ma numer -1. Z kolei licznik `tocdepth` wskazuje, które elementy podziału wejdą do spisu treści. Domyślne wartości tych poziomów są sobie równe, ale nie jest to konieczne.

[7] autora, nie komputera!

Czasami chcemy mieć podział wyróżniony wizualnie w tekście, ale bez całego bagażu numeracji, spisu treści i paginy. Służą temu wersje gwiazdkowe instrukcji podziału. Np. LaTeX-owe środowisko `bibliography` zawiera w swej definicji fragment

```
\section*{\refname
  \@mkboth{\MakeUppercase\refname}...
```

Bibliografia utworzy odrębny rozdział, nienumerowany, a żywa pagina została w tym przypadku jawnie zdefiniowana, być może inaczej niż w użytej klasie. Przy okazji widzimy wysiłek twórców LaTeX 2_ε zmierzający do ułatwienia lokalizacji, wystarczy zdefiniować makro `\refname` używając lokalnego dialektu.

Jeszcze jedną metodą rozróżnienia pomiędzy treścią tytułu w tekście a treścią w paginie i w spisie treści jest użycie argumentu opcjonalnego. Wówczas normalny argument instrukcji podziału jest umieszczany w głównym tekście, a opcjonalny służy pozostałym celom.

Wspomnieć tu należy jeszcze o prostej instrukcji `\appendix`. Od momentu jej wystąpienia, rozdziały „numerowane” będą kolejnymi literami alfabetu. W ten sposób możemy uważać te rozdziały za „załączniki” wykraczające poza główny tekst.

Twierdzenia Nie ma artykułu naukowego bez twierdzeń. Są one zazwyczaj numerowane i odpowiednio wyróżniane w tekście. Z formalnego punktu widzenia są to środowiska, LaTeX posiada jednak specjalne narzędzie do definiowania tych środowisk. I tak

```
\newtheorem{thm}{Twierdzenie}[section]
```

definiuje środowisko `thm` którego użycie spowoduje poprzedzenie tekstu słowem `Twierdzenie` wraz z numerem złożonych tłustym drukiem oraz złożenie samego tekstu kursywą. Licznik twierdzeń będzie zerowany z początkiem każdego rozdziału. Z kolei deklaracja

```
\newtheorem{lem}[thm]{Lemat}
```

definiuje środowisko `lem` do lematów numerowanych wspólnie z twierdzeniami. Każde z tak zdefiniowanych środowisk dopuszcza parametr opcjonalny, jest to dodatkowy tekst, który również będzie wytłuszczony, najczęściej nazwa twierdzenia.

Nie zawsze chcemy by tekst „twierdzenia” był składany kursywą. Wówczas można zastosować metodę okrężną

```
\newtheorem{defin}{Definition}
\newenvironment{dfn}{\begin{defin}\rm}
                    {\end{defin}}
```

Taka sztuczka powoduje jednak, że pozbawiamy się możliwości stosowania parametru opcjonalnego do środowiska tego „twierdzenia” – nie jest ono przecież zdefiniowane przy użyciu `\newtheorem`. Klasa `llncs` posiada narzędzia pozwalające bezpośrednio zdefiniować takie środowisko składane pismem prostym. Pakiet `theorem` posiada jeszcze więcej możliwości definiowania środowisk twierdzeniopodobnych.

Klasy LaTeX-owe przygotowane przez wydawnictwa, choćby klasa `llncs`, zawierają często już definicje pewnych standardowych środowisk dla twierdzeń. Jest to i wygodne i męczące. Wygodnie jest mieć już część pracy za sobą. Ale zmieniając klasy, np. przygotowując dotychczasowy raport techniczny do druku, musimy stale pilnować, które ze środowisk są definiowane przez nas, a które przychodzą z zewnątrz. Cierpi na tym nieco przenaszalność tekstu źródłowego.

Wzajemne odwołania W tekście naukowym częściej niż w innym chcemy precyzyjnie odwołać się do określonego fragmentu tekstu. LaTeX posiada dwa narzędzia do tego właśnie celu. Pierwszym z nich jest makro `\label`, jego parametrem jest wymyślona przez nas nazwa. Nazwa może być w miarę dowolna, może zawierać spacje, należy jednak z ostrożnością podchodzić do polskich liter. Nie w każdej implementacji skutek będzie równy spodziewanemu. W miejscu pojawienia się etykiety LaTeX bada wartości dwóch liczników, licznika stron i licznika „bieżącego”. Ten ostatni to np. numer podrozdziału, albo numer twierdzenia, albo numer pozycji w wyliczance, albo … Jeśli użytkownik posiada własny licznik, to może go powiększać metodą `\stepcounter` lub `\refstepcounter`, w drugim przypadku jego wartość stanie się owymż licznikiem bieżącym. Wartości tych liczników zostają zapisane w pliku pomocniczym `.aux`, produkowanym przez LaTeX-a głównie w tym celu. Autor może odwołać się do zaznaczonego miejsca używając makra `\ref` lub `\pageref` z parametrem równym etykiecie i otrzymać wartość zapamiętanego licznika bieżącego lub numer strony. Wartości te są odczytywane przez LaTeX-a z pliku pomocniczego. Bystry czytelnik rozumie już teraz, że w wersji ostatecznej artykułu należy dokonać dwukrotnej[8] kompilacji, tak by za drugim razem LaTeX posiadał wiarygodne informacje o położeniu etykiet. Nie jest to zresztą trudne, LaTeX informuje o zmianie położenia etykiet w stosunku do poprzedniego, użytego podczas składu. Jest to jeden z niewielu naprawdę pożytecznych komunikatów wypluwanych podczas kompilacji.

Jedno z częstych nieporozumień dotyczących odwołań dotyczy środowisk ruchomych. Samo użycie środowiska `figure` lub `table` *nie* powoduje wy-

[8] co najmniej, mogą być inne powody wymagającej większej liczby przebiegów; czytelnik zechce sprawdzić, czy *te słowa* pojawiły się na stronie 46.

generowania żadnego numeru. Tak więc licznik bieżący będzie dotyczył np. numeru rozdziału. Licznik bieżący zostanie zmieniony dopiero poprzez użycie podpisu `\caption`. Tak więc etykieta musi znaleźć się wewnątrz środowiska ruchomego, ale *za* podpisem, a najlepiej wewnątrz niego. Widzieliśmy już wcześniej, że obiekt ruchomy może posiadać kilka podobiektów, każdy ze swoim numerowanym podpisem.

Z użyciem etykietowania związane jest zjawisko uboczne, czasami o nieoczekiwanym skutku. Otóż `\label{..}` nie produkuje tekstu, ale jest pudełkiem o zerowej szerokości. Może więc spowodować, że powstaną *dwie* spacje w nieoczekiwanym miejscu, może mieć to dalsze skutki jeśli występuje na początku niektórych środowisk. Warto o tym pamiętać i przyswoić sobie jeden z dwóch stylów:

```
\begin{theorem}\label{etykieta}
```

```
\begin{theorem}                    \label{etykieta}%
```

I jeszcze trywialna uwaga na koniec. `\ref` generuje numer, który łączy się logicznie z poprzedzającym słowem. Będzie nieładnie, jeśli wypadnie na początku wiersza, warto nabrać nawyku łączenia go z poprzednim słowem tyldą.

Drugie narzędzie LaTeX-a służące do odwołań dotyczy wyłącznie pozycji bibliograficznych, o nim dalej.

Bibliografie Środowisko `thebibliography` jest kolejnym środowiskiem do wyliczeń. Poszczególne pozycje poprzedzane są makrem `\bibitem` z obowiązkowym parametrem będącym etykietą danej pozycji bibliograficznej, odwołania do niej produkowane są poprzez `\cite`. Jak w każdej wyliczance, można użyć opcjonalnie własnych etykiet pojawiających się w tekście lub pozostawić LaTeX-owi ponumerowanie pozycji. Standardowe klasy LaTeX-a przewidują, że cytowania generują etykiety pozycji bibliograficznej w nawiasach kwadratowych. Nie każdemu musi się to podobać, istnieją pakiety, które zapewniają więcej możliwości.

Środowisko `thebibliography` wymaga podania jako argumentu najszerszej etykiety w bibliografii. Nie musi to być etykieta naprawdę występująca, w przypadku użycia liczb rozsądnym parametrem jest `99`, w przypadku etykiet tekstowych, najczęściej trzy litery z nazwiska plus dwie cyfry roku, możemy podać jako parametr `XXX99`.

Jak już wspomniano, samo użycie tego środowiska generuje nowy rozdział, jego tytuł przechowywany jest w makrze `\refname`. Makro to ma zdefiniowaną wartość w standardowych klasach LaTeX-a, styl `polski` definiuje odpowiednie polskie słowo. Każdy jednak użytkownik ma prawo posiadać własny tytuł dla bibliografii, szczególnie, jeśli miał powód by użyć tego środowiska więcej niż raz.

Samo tworzenie bibliografii jest zajęciem żmudnym, szczególnie jeśli chce się zachować pewną jednorodność stylu. Z drugiej strony, autor pracujący w danej dziedzinie będzie prawdopodobnie wielokrotnie cytował te same prace i będzie miał chęć kopiować biliografie z różnych nieprzystających źródeł. Zadanie tworzenia bibliografii ułatwia BibTeX, program generujący plik bibliograficzny z bazy danych zawierającej informacje o pozycjach bibliograficznych. Generowany plik zawiera wyłącznie pozycje cytowane w danym artykule i może być wygenerowany w różnych stylach. Więcej o tym w poprzednim numerze Biuletynu GUST'u, [Bor99].

Spisy i inne skorowidze Trzy podstawowe spisy standardowego LaTeX-a to

```
\tableofcontents \listoffigures \listoftables
```

Ich użycie spowoduje, że LaTeX produkuje kolejne pliki pomocnicze, z których w kolejnym przebiegu odczyta potrzebne informacje. Jest to jeszcze jeden powód konieczności wielu przebiegów kompilacji. Tu uwaga: LaTeX *nie* ostrzega, że któryś ze spisów został złożony na podstawie nieaktualnych danych. Sam spis treści też jest rozdziałem, jednak w wersji gwiazdkowanej, tzn. sam nie wchodzi do spisu treści, podobnie jak bibliografia. Jeśli chcemy do spisu na siłę coś wcisnąć, choćby bibliografię, to mamy do dyspozycji

```
\addcontentsline{toc}{section}{Literatura}
```

do umieszczenia w miejscu, które potem spis treści wskaże.

Sprawa tworzenia indeksu, czyli skorowidza, jest bardziej skomplikowana, szczególnie w kontekście języka fleksyjnego, jakim jest język polski. LaTeX posiada środowisko `theindex` będące specjalnym rodzajem wyliczenia, właściwym do złożenia skorowidza. Sam LaTeX jednak nie wystarcza, by utworzyć skorowidz. Dane do skorowidza będą gromadzone w pliku pomocniczym `.idx`, o ile w preambule użyta będzie instrukcja `\makeindex`. Do pliku tego będzie zapisywana kolejna pozycja przy każdym wystąpieniu makra `\index` – zapisany będzie parametr tego makra i numer strony. Samo makro nie produkuje tekstu (ale produkuje niewidoczne pudełko, uwaga na podwójne spacje). Rozsądnym rozwiązaniem jest przygotowania własnego makra, np.

```
\newcommand\newterm[2]{\emph{#1}\index{#2}}
```

i jego późniejsze systematycznie używanie:

```
Podstawową rolę w~matematyce pełni pojęcie
\newterm{funkcji}{funkcja}.
```

Teraz w pliku `.idx` znajdzie się wiersz z numerem strony, na której zaznaczyliśmy słowo `funkcja`. Po zakończeniu kompilacji dokumentu należy wywołać zewnętrzny program `makeindex`, który to program wygeneruje plik o nazwie zakończonej na `.ind` zawierający kod LaTeX-owy tworzący skorowidz. Plik ten *nie jest* automatycznie włączany do dokumentu tylko z okazji swojego istnienia, należy go włączyć jawnie, np. `\printindex`.

Program `makeindex` zajmuje się m. in. sortowaniem pozycji. Oczywiście sortowanie zgodne z polskim porządkiem liter nie może przyjść samo z siebie. Koledzy z GUST'u dokonali odpowiednich polonizacji[9], niestety, nie mam doświadczeń w tej sprawie.

Pakiety

W tym rozdziale chciałbym przedstawić kilka pakietów, które uważam za wygodne, nawet jeśli trochę się nimi rozczarowałem, ale są niezbędne do pewnych celów. Niektóre z tych pakietów nie wymagają od użytkownika niczego, oprócz ich zadeklarowania, inne są dużymi systemami.

tools Zestaw pakietów autorstwa Franka Mittelbacha i Davida Carlisle.

array, longtable Podstawowe klasy LaTeX-a nie pozwalają na bardziej wymyślną obsługę tabel. Wydaje się, że LaTeX ograniczył nawet możliwości dostępne w zwykłym TeX-u. Jednym z pakietów poprawiających tę sytuację jest pakiet `array`.

Jednym z problemów rozwiązywanych przez ten pakiet jest opcja jednolitego potraktowanie wybranej kolumny, np. zmiany fontu w całej kolumnie. Pakiet pozwala na trochę więcej, mianowicie między wzorcami kolumn można umieścić dowolny tekst wraz ze wskazówką do której z kolumn ma on zostać „doklejony". I tak fragment wzorca `>{\bf}l<{ :}` oznacza, że do zawartości każdej komórki tej kolumny zostanie dodana z przodu instrukcja `\bf`, wówczas zawartość tej kolumny zostanie złożona pismem wytłuszczonym. Należy pamiętać, że zawartość każdej komórki stanowi grupę, nie ma więc potrzeby osobnego grupowania. W podanym przykładzie, na końcu zawartości każdej komórki dodany będzie dwukropek. Innym zastosowaniem tej możliwości jest deklaracja kolumny jako matematyki, `>{$}c<{$}`, w środowisku `tabular`. Rozsądnym jest też dodanie w ciemno zerowego pudełka do wąskich akapitów

```
\begin{tabular}{...>{\hspace{0pt}}p{10mm}...}
```

Innym szczegółem jest obecność wzorców produkujących pudełko z akapitem wyrównane względem środka, `m`, lub podstawy, `b`.

Pakiet ten powoduje również inną obsługę linii pojawiających się w tabeli. W przypadku pojedynczych cienkich linii różnica jest mało wyraźna. Zaczyna być jednak wyraźniejsza, w przypadku stosowania linii pogrubionych (za grubość odpowiada parametr `arrayrulewidth`).

Jeszcze innym pakietem usuwającym ograniczenia LaTeX-a w dziedzinie tabel jest `longtable`. Ma on zastosowanie, gdy tabela jest zbyt długa by zmieściła się na jednej stronie. Oczywiście każdą tabelę można zawsze podzielić ręcznie, pakiet ten robi to automatycznie. Co więcej, kosztem większej liczby przebiegów, wyrównuje szerokości kolumn na wielu stronach. Wcześniejszym pakietem do obsługi długich tabel był `supertabular`, nie wyrównywał on jednak kolumn na różnych stronach.

multicol Jest to zestaw makr umożliwiających skład wielołamowy, w dodatku można łączyć takie fragmenty na jednej stronie. Jest to duży krok do przodu w stosunku do standardowego LaTeX-a dopuszczającego jedynie skład dwułamowy. W standardowym LaTeX-u każde użycie `\twocolumn[tekst]` i odwrotnie rozpoczyna nową stronę, jedyną możliwość mieszania daje właśnie przekazanie części tekstu do składu jednołamowego na początku strony dwułamowej.

Niestety, cena płacona za taką giętkość nie jest mała. Praktyka pokazuje, że rozkład materiału w pionie nie jest najlepszy. Co więcej, w standardowym zastosowaniu tego pakietu, nie daje on wiele więcej możliwości niż standardowy LaTeX. W przypadku czasopism naukowych, nie gazet codziennych, skład wielołamowy oznacza skład dwułamowy, a tekst składany jest w ten sposób w całości, z wyłączeniem być może tytułu. Wówczas jedynym plusem użycia tego pakietu jest wyrównanie łamów na ostatniej stronie.

theorem Wspomnieliśmy o ograniczeniach standardowych środowisk dla twierdzeń. Zdefiniowane za pomocą dostępnego w LaTeX-u narzędzia nie spełniają często wymagań wydawców – a to numer powinien poprzedzać słowo „twierdzenie", a to kursywa jest niepożądana itd. Pakiet ten umożliwia jawne definiowanie fontów dla nagłówka i tekstu twierdzenia, posiada też kilka stylów dla kształtu nagłówka odmiennych od standardowego.

indentfirst Pakiet ma właściwie dwa wiersze, jedyną jego rolą jest „europejskie" ustawienie zmiennej odpowiedzialnej za wcięcie pierwszego akapitu w rozdziale.

xspace Pakiet usuwa pewną niedogodność przy wprowadzaniu tekstu źródłowego – w tekście

```
\TeX nad \TeX-ami
```

pierwsze wystąpienie TeX-a zostanie przyklejone do następnego słowa. Użytkownik musi pamiętać by

[9] `www.ia.pw.edu.pl/~wujek/tex/idx/`

umieszczać pustą grupę i nie może tego załatwić łatwym makrem. Chyba, że z pakietem,

```
\newcommand\tex{\TeX\xspace}
```

a `\xspace` wyprodukuje spację w miarę potrzeby.

graphics Zestaw pakietów graficznych autorstwa Davida Carlisle.

graphics, graphicx Pakiety te korzystają z faktu, że ostateczny wydruk będzie dokonywane za pomocą postscriptu – np. interpreter postscriptu będzie odpowiedzialny za obrócenie pudełka, `\rotatebox`, a XƎLATEX zajmie się jedynie obliczeniem jego wymiarów prostokątnych. Innym zadaniem jest bezpośrednie włączanie grafiki dostępnej w formacie `eps`.

color Pakiet ten umożliwia zmianę koloru pisma podobnie do zmiany rozmiaru czy kroju pisma. Mamy do dyspozycji kolejno przełącznik, pudełko z kolorowym tekstem i pudełko z zadanym tłem,

```
\color{red}
\textcolor{green}{zielono mi}
\colorbox{blue}{Lucy in the sky}
```

Istnieją też bardziej zaawansowane metody definiowania kolorów.

Oprócz kolorowania slajdów czy dodawania bajerów do listów pakiet ten ma jeszcze jedno zastosowanie. Otóż pisząc tekst w kilka osób można stosować kolory w celu wyróżnienia autorstwa czy stopnia zaawansowania poszczególnych fragmentów tekstu. Na ekranie nie ma problemu z rozróżnianiem kolorów, łatwo jest też wyłączyć cały mechanizm zmieniając definicję powyższych makr na jałową.

Również pakiet `pstricks` posiada definicje kolorów. Dokumentacja stwierdza pewną niezgodność tych pakietów.

lscape Umożliwia złożenie poszczególnych stron w środowisku `landscape`.

pstricks Pakiet ten jest ogromnym systemem, da się w nim zrealizować przykłady z książki [GRM97]: krzywe, wypełnienia obszarów, tekst składany wdłuż krzywej, diagramy. Przykładowo, rysunek w artykule [Bor99] zawierał fragmenty

```
\begin{pspicture}(-4,-1)(4,1)
\rput(2,0.5){\ovalnode{latex}{\LaTeX}}
\rput(2,-1.5){\rnode{aux}{\psframebox{.aux}}}
\nccurve[angleA=330,angleB=30]{->}{latex}{aux}
\ncline{oo->}{bbl}{latex}
```

Elementy zostały rozłożone na wyimaginowanej macierzy, proste i krzywe o zadanych cechach (kąty, rodzaj końców) zostały wykreślone odwołując się do etykiet tych elementów.

W LATEX-u nawet prosty skład linii prostej z gotowych fragmentów jest tylko protezą – składnia dla prostych w pakiecie jest o wiele bardziej przejrzysta bo i są to prawdziwe proste. Pakiet posiada też krzywe Beziera, oczywiście realizowane postscriptowo. W przeciwieństwie do standardowych klas LATEX-a, są to krzywe *trzeciego* stopnia. Wydaje mi się, że są one mniej intuicyjne, choć o większych możliwościach, niż krzywe stopnia drugiego. Drugim ich minusem jest trudność ewentualnego przeniesienia gotowego rysunku z LATEX-a używającego `qbezier`. Jednak zalety podejścia postscriptowego powinny powodować całkowite zaniechanie prób składania krzywych metodą kropek.

Pakiet dobrze współdziała ze standardowym LATEX-em, w szczególności można bez problemu dodać do istniejącego rysunku w środowisku `picture` obiekt postscriptowy, jeśli zaistnieje nagła konieczność. Trzeba jednak pamiętać, że jednostka miary *nie jest* dziedziczona z LATEX-a, trzeba ją ustawić, najlepiej jako `\psset{unit=\unitlength}`.

carlisle Zestaw innych pakietów Davida Carlisle.

fix2col Pakiet ten usuwa pewne „cechy” obecne w składzie dwułamowym w wykonaniu LATEX-a. Na pewno nie zaszkodzi, a może coś poprawić.

enumerate Pakiet umożliwia łatwą definicję etykiet w wyliczeniach – całe środowiska z wyliczaniem, nie tylko poszczególne pozycje, dopuszczają podanie etykiety jako argumentu opcjonalnego.

verbatim, url, typehtml Standardowe środowisko `verbatim` służy do składu tekstu dosłownie, jest przydatne szczególnie gdy cytuje się kod źródłowy LATEX-a. Czasami jednak ta dosłowność przeszkadza. Powiedzmy, że chcemy włączyć do tekstu LATEX-owego plik z tekstem źródłowym, w TEX-u czy w innym języku. Wówczas bezcelowe jest używanie instrukcji `\input` w środowisku `verbatim` – zamiast włączenia pliku otrzymamy tę instrukcję dosłownie. Należy raczej przerobić plik źródłowy dopisując na jego początku i końcu instrukcje środowiska. Sposób ten jest męczący, jeśli plik źródłowy jest zmienny, a zawsze jest nieelegancki. Pakiet `verbatim` posiada dodatkowe środowisko, `verbatiminput`, wywoływane z parametrem będącym nazwą pliku. Jego użycie spowoduje włączenie pliku i jego dosłowny skład.

Podobnie bezowocna będzie próba zmiany wielkości fontu wewnątrz środowiska `verbatim`, skądinąd rozsądna. Ale można zdefiniować sobie własne środowisko

```
\newenvironment{smallverb}
  {\small\verbatim}{\endverbatim}
```

Dlaczego definicja ta nie działa bez pakietu – nie wiem.

Pakiet ten posiada też produkt uboczny, środowisko `comment` pozwalające wyłączyć większy fragment tekstu.

Pakiet `url` rozwiązuje bardzo specyficzny, ale chyba narastający, problem. Mianowicie, wraz z rozpowszechnieniem internetu, coraz częściej cytuje się długie adresy internetowe. TEX jednak nie znajdzie w takim napisie miejsca do dzielenia. Problemem jest też możliwość wystąpienia w takim adresie znaków specjalnych: procentu (starsi użytkownicy internetu pamiętają może jeszcze) lub w-tył-ciacha. Używając tego pakietu można zapisać `bardzo.długi.adres@internetowy/lub\ścieżkę$zawierające_znaki%specjalne`. Napis taki zostanie potraktowany dosłownie, ale z możliwością dzielenia. Podany tu przykład jest skrajny, TEX-owi nie wystarcza sama możliwość dzielenia wyrazów, potrzebuje jeszcze możliwości rozciągnięcia spacji, a tych tu brak. Może się okazać, że niezbędne będzie użycie `\sloppy`, zezwalające na niedomiar w akapicie.

A propos internetu, coraz częściej kodem źródłowym jest html. Jeśli chcemy umieścić w dokumencie LATEX-owym list otrzymany w formacie html, jego skład dosłowny jest niecelowy. Można próbować posiłkować się pakietem `typehtml`, prosty kod html'owy zostanie złożony całkiem zręcznie. Obawiam się jednak, że zmiany w stylu programowania w html'u są tak szybkie, że pakiet nie nadąży.

caption Pakiet `caption` ułatwia obsługę podpisów pod rysunkami i tabelami i zawiera dodatkowe możliwości, np. umożliwia centrowanie *ostatniego* wiersza podpisu. Można również zmienić font użyty do podpisu.

geometry Pakiet ten pozwala na wygodne definiowanie wymiarów strony. Użytkownik podaje pewne wymiary, a pakiet automatycznie ustawia LATEX-owe parametry sensownie. Np.

```
\usepackage[body={16cm,24cm},nohead]{geometry}
\geometry{footskip={30pt},a4paper,dvips}
```

Ostatnie dwa parametry mogłyby równie dobrze znaleźć się w pliku konfiguracyjnym `geometry.cfg` na stałe, przecież zawsze będziemy używać papieru A4 i używać dvips'a.

fancyhdr Pakiet ten służy głównie do deklarowania wymyślniejszej niż zazwyczaj paginy.

seminar Specjalna klasa dokumentów ułatwiająca przygotowanie prezentacji za pomocą techniki klasycznej, tzn. slajdów. Slajdy są domyślnie składane poziomo, co na marginesie mówiąc warto polecić wszystkim prezenterom. Klasa ta zachowuje cechy standardowych klas, tak, że można bezpiecznie kopiować fragmenty z gotowego artykułu, i jednocześnie daje środowisko `slide` służące do składu slajdów. Kilka możliwości to zmiana powiększenia stosowanego dla slajdów (domyślne jest oczywiście całkiem duże), przygotowanie nakładek, wydruku tylko slajdów lub tylko reszty czy wydruku wskazanych slajdów.

Różne symbole LATEX, a właściwie TEX, nie daje łatwych metod zdefiniowania potrzebnego akurat symbolu. Np. ktoś wymyślił sobie używanie odwróconego symbolu ampersanda i już mamy duży problem. Jednym z rozwiązań może być intensywne używanie MetaFontu czy innych narzędzi graficznych. Innym rozwiązanie jest kopiowanie cudzych rozwiązań, kilka pakietów temu właśnie służy: `amsfonts`, `stmaryrd`, `ec`, `euler` i kilka innych.

Pamiętać należy, że instalacja takiego pakietu jest rzeczą skomplikowaną. Musimy udostępnić LATEX-owi nie tylko zestawy makr odwołujące się do nowych symboli, ale i same symbole.

Bibliografia

[Lam86] Leslie Lamport. *LATEX: A Document Preparation System.* Addison-Wesley, 1986. Istnieje tłumaczenie polskie: Wyd. Ariel, tłum.: Piotr Wyrostek, 1992.

[GMS93] Michael Goossens, Frank Mittelbach i Alexander Samarin. *The LATEX Companion.* Addison-Wesley, 1993.

[GRM97] Michael Goossens, Sebastian Rahtz i Frank Mittelbach. *The LATEX Graphics Companion.* Addison-Wesley, 1997.

[OPHS] Tobias Oetiker, Hubert Partl, Irene Hyna i Elisabeth Schlegl. *Nie za krótkie wprowadzenie do systemu LATEX 2ε.* CTAN i TEXLive, 1998.

[Bor99] Andrzej M. Borzyszkowski. BIBTEX – narzędzie do przygotowania bibliografii. GUST, zeszyt 13 (1999), str. 44–55.

◇ Andrzej M. Borzyszkowski
`a.borzyszkowski@ipipan.gda.pl`