Uniwersytet w Białymstoku

Wydział Matematyczno-Fizyczny

Instytut Matematyki

Barbara Duchnowska

Automatyczne tworzenie spisu symboli w LaTeX-u

Praca dyplomowa napisana
pod kierunkiem
dr. Mariusza Żynela

Białystok 2006

Składam serdeczne podziękowania
dr. Mariuszowi Żynelowi
za pomoc w tworzeniu pracy.

Barbara Duchnowska

Spis treści

Wstęp

Tworzenie indeksu nie jest prostą sprawą. Teoretycznie ma on pomóc Czytelnkowi w znalezieniu w książce tego, co jest mu potrzebne. Niestety pomimo swojej użyteczności bardzo często index jest pomijany w tworzeniu książki.

LaTeX jest zestawem instrukcji (poleceń, makr, makrodefinicji) umożliwiającym autorom redagowanie i wydrukowanie ich prac na poziomie druku profesjonalnego przy użyciu powszechnie dostępnych komputerów.

Aby wydać swe dzieło autor dostarcza je do wydawnictwa, gdzie redaktor decyduje o układzie graficznym dokumentu. Redaktor-człowiek odgaduje intencje autora ustala, które fragmenty tekstu są tytułami rozdziałów, podrozdziałów, cytatami, wzorami matematycznymi itd. LaTeX jest właśnie takim redaktorem a TeX - zecerem. Z tym, że LaTeX jako program potrzebuje pomocy autora. Jedną z jego zalet jest to, że wystarczy znać jego fragmenty, aby móc efektywnie pracować.

Pierwszym problemem jest wybór koncepcji, czyli pomysłu na skorowidz. Indeks powinien znacznie ułatwiać czytelnikowi dostęp do treści dzieła, i to w sposób najprostszy. Trudną sprawą jest układ logiczny haseł w indeksie i ich dobór. Następnie problem może stworzyć "wychwycenie" wszystkich haseł i stron, na których one występują oraz sortowanie. Praca taka jest bardzo żmudna. O ile pierwszy z etapów jest trudny i pracochłonny to doświadczony redaktor może sprawnie to wykonać. Drgi etap za człowieka może wykonać maszyna, czyli komputer wraz z oprogramowaniem. Właśnie do rozwiązania tego problemu nadaje się idealnie TeX .

Cały zapis tekstu pod LaTeX-em, łącznie ze wzorami, rysunkami i wykresami, prowadzony jest w kodzie ASCII. Pozwala to na przesyłanie tekstów pocztą elektroniczną lub na dyskietce, dając równocześnie gwarancję, że po stronie odbiorcy tekst zostanie odczytany dokładnie tak, jak życzył sobie nadawca.

Korzystanie z LaTeX-a , bądź TeX-a polega na dostarczaniu temu systemowi pewnych słów, równań, rysunków, tabel itp., aby otrzymać plik zawierający skład gotowy do druku.

Oprogramowanie jest zrobione z myślą o matematyce ale nie wykluczone jest używanie go w innych dziedzinach nauki, które wymagaja użycia znaków i symboli dla przedstawienia danego materiału. Tego samego oprogamowania mogliby używać fizycy i chemicy.

Rozdział 1

Wprowadzenie do tematu

1.1 System LaTeX

TeX jako język programowania powstał w Stanach Zjednoczonych na Uniwersytecie Stanforda. Jego autorem jest profesor Donald E.Knuth. Został ukończony w 1986 roku w 10 lat po rozpoczęciu pracy.

TeX to wyspecjalizowany język programowania pozwalający na złożenie w sposób automatyczny dokumentu o dowolnej skali trudności. Co za tym idzie TeX to skomplikowany język, o specyfcznej składni i dużej liczbie poleceń. Unikalny algorytm, którym posługuje się TeX przy składaniu akapitów, powoduje, że nie ma programu oferującego pod tym względem lepszych możliwości.

TeX został zaprojektowany w taki sposób, że może być przystosowany do składania tekstów w dowolnych językach, nawet tak egzotycznych dla nas, użytkowników alfabetu łacińskiego, jak drukowanie z góry na dół (chińska albo japońska wersja językowa) czy z prawa na lewo (hebrajska i arabska).

Wreszcie TeX jest oprogramowaniem otwartym, przez co rozumieć należy jego zdolność do współpracy z innymi programami.

System przygotowania dokumetów LaTeX jest specjalną wersją systemu TeX. Dodaje on do TeX-a kolekcję komend upraszczających składanie tekstu i pozwalających użytkownikowi skoncentrować się na tworzonym tekście, a nie na komendach. LaTeX jest bardzo rozbudowanym zestawem makr, zawierającym wiele mechanizmów opisu struktury logicznej dokumentów. Jest to obecnie najbardziej rozpowszechniony format i dostarczany w każdej dystrybucji stanowi tym samym ważny sładnik systemu TeX.

Twórcą LaTeX-a jest Leslie Lamport. Zaprojektowany przez niego zestaw makr został „zamrożony" jako tzw. LaTeX 2.09 i praktycznie nie jest już używany. Obecnie używany LaTeX to intensywnie rozwijany przez tzw. grupę LaTeX Team (kierowaną przez Franka Mittelbacha), określany jako LaTeX2epsilon (LaTeX2e).

W 1992 roku powstała Polska Grupa Użytkowników Systemu TeX (GUST). Obecnie GUST zrzesza ponad 250 członków indywidualnych i instytucjonal-

nych.

Polecenia LaTeX-a występują w dwóch następujących odmianach:

1. Instrukcje rozpoczynające się znakiem *backslash*, po których występuje ciąg liter. Końcem instrukcji jest odstęp lub inny znak nie bedący literą.
2. Instrukcje składające się ze znaku *backslash* oraz dokładnie jednego znaku nie bedacego literą.

W nazwie instrukcji LaTeX odróżnia litery małe i duże, natomias używanie polskich liter w nazwach instrukcji jest niemożliwe. Niektóre instrukcje posiadają argumenty, które pisze się w nawiasach sześciennych.

Zalety programu LaTeX :

- logiczne i jednolita budowa komend ułatwiających ich przyswajanie;
- automatyczne formatowanie stron, dobór wielkości czcionki, składanie wierszy, centrowanie itp.;
- bogactwo komend generujących symbole matematyczne ze wszystkich ich dziedzin;
- logiczny i zgodny z logiką zapis formuł matematycznych;
- szeroki zestaw środków pozwalający generować tabele, rysunki, zestawy objaśnień;
- samonumerujące odsyłacze do numerów wzorów, tabel, rysunków, podrozdziałów oraz numerów stron;
- samonumerujące odsyłacze do cytowanej literatury oraz możliwość tworzenia własnej bazy danych bibligraficznych, która ponadto pozwala na kształtowanie cytowań i spisu literatury według różnych konwencji;
- prawie automatyczne tworzenie spisu treści, spisu tabel i rysunków oraz indeksów tematycznych;
- ogólna dostępność zarówno TeX-a jak i LaTeX-a;
- dostępne są przygotowane przez fachowców gotowe układy graficzne.

I wreszcie zaleta ostatnia: LaTeX uwalnia nas od konieczności uczenia się TeX-a, który jest znacznie bardziej rozbudowany.

Wad systemu LaTeX jest niewiele.W początowym etapie pracy z LaTeX-em brak możliwości natychmiastowego zobaczenia tekstu, a zwłaszcza wzorów matematycznych jest nieco kłopotliwe. Po kilkunastu próbach przekonamy się jednak, że pilne ogladanie każdego poprawnie zapisanego wzoru nie jest konieczne, bo LaTeX potafi nadać wzorom satysfakcjonującą postać graficzną. Rysunki niestety trzeba szczegółwo opisywać za pomocą komend stawiających kreski, kółka itp.

1.2 Tworzenie indeksów

Przy tworzeniu i sortowaniu skortowidza LaTeX wykorzystuje program *Makeindex* Przygotowanie indeksu wygląda następująco:

- tworzymy dokument LaTeX-owy, w którym etykietujemy słowa wybrane do skorowidzu komędą `\index`,
- kompilujemy i otrzymujemy nieposortowany plik `.idx`, zawierający hasła występujące kolejno w tekscie,
- program *Makeindex* sortuje wejściowy plik `.idx` tworząc posortowany plik `.ind` będący złożonym skorowidzem,
- kolejna kompilacja daje w efekcie publikację z indeksem.

Na poczatku dodajemy styl `makeindex` jako parametr komendy, następnie w preambule komendę `\makeindex` tak, aby znajdowała sie ona pomiędzy komendami `\documentstyle` a `\begin`. Należy w tekście umieścić również komedę `\printindex` w miejscu gdzie chcemy aby indeks się pojawił. Obok wybranego słowa, które chcemy aby znalazło się w spisie należy wpisać komendę `\index`.

Podczas przetwarzania pliku źródłowego przez LaTeX-a każda instrukcja `\index` powoduje zapisanie odpowiedniej pozycji skorowidza oraz numeru strony, którego ta pozycja dotyczy do pliku pomocniczego. Z kolei plik `.idx` musi być przetworzony za pomocą programu *Makeindex*. W rezultacie program ten tworzy posortowany indeks i zapisuje go do pliku o nazwie identycznej z nazwą głównego pliku źródłowego i rozszerzeniu `.ind`. Jeżeli jeszcze raz przetworzymy plik źródłowy to tym razem taki skorowidz zostanie dołączony do dokumentu w miejscu gdzie znajduje sie polecenie `\printindex`.

Rozdział 2

Problem indeksowania symboli

2.1 Warianty spisu symboli w literaturze

Bardzo użytecznym elementem książek szczególnie tych dotyczących konkretnej dziedziny nauki sa spisy symboli. W indeksie możemy szybko odnaleźć to czego szukamy i miejsce w książce gdzie dany symbol został zdefiniowany.

Z dwudziestu wybranyh książek z dziedziny matematyki tylko w sześciu pozycjach znajdował się spis symboli. Najczęściej był on umieszczony na końcu książki przed spisem pojęć. Tylko w jednej z pozycji, a mianowicie w " *Introduction to linear algebra* " *Johnsona, Riessa, Arnolda* spis symboli znajdował się na początku ksiąki. Autorzy wypisali symbole występujące w książce według kolejności w jakiej występują, uwzględniając tylko te strony gdzie dany symbol jest zdefiniowany. Taką samą metodę jak poprzednicy zastosowali *S. Balcerzyk, T. Józefiak* w swojej książce " *Pierścienie przemienne* " z tą różnicą, że spis znajduje się na końcu książki.

S. Łojasiewicz w *"Wstępie do geometrii analitycznej zespolonej"* umieścił spis symboli jak większość autorów na końcu książki. Indeks jest posegregowany według kolejności stron na których występują symbole. Tylko w tej pozycji spis jest podzielony dodatkowo na rozdziały.

J. Słupecki, K. Hołkowska, K. Piróg-Rzepecka w swoim dziele " *Logika matematyki* " posegregowali spis symboli według stron na których wystepuje symbol, z tym że są to numery wszystkich stron gdzie dany znak występuje. Kolejność wyznacza numer strony gdzie symbol pojawił sie po raz pierwszy.

Alfabetyczny spis symboli w swojej książce zastosował *J. Grancarzewicz* " *Geometria różniczkowa* "Indeks zawiera dany symbol, jego opis i numer srony na której on występuje.

Zazwyczaj spis symboli wyglądał jednakowo, czyli strona podzielona jest na dwie kolumny gdzie w każdej z nich umieszcone są: symbol, opis symbolu oraz strona na której on występuje. We wspomnianej wcześniej książce *S. Łojasiewicza* uwzględnione są tylko symbol i strona. W pozycji " *Logika matematyki* " spis symboli jest utworzony z jednej kolumny i zawiera również

opis.

Z wybranych pięciu pozycji z dziedziny ekonomi tylko w jednej znajdował się spis symboli, a mianowicie w *",Podstawy zarządzania finansami," Eugene F. Brighama* Ṫak jak jest to najpopularniejsze znajduje się on na końcu książki, jest uporządkowany alfabetycznie. Wszyskie symbole pisane są dużą literą.

2.2 Sortowanie symboli

2.2.1 Reguły sortowania po Polsku

Reguły sortowania po polsku ujęte są w normie PN-80/N-01223, która mówi, iż porządek w indeksie powinnien być zgodny z porządkiem w alfabecie polskim z dodatkiem liter $q, v, x,$. Obce znaki diakrytyczne nie powinny wpływać na porzadek. Porządek w polskich skorowidzach, to tak zwany porządek słowowy, czyli spacje i znaki interpunkcyjne zostaną umieszczone przed literami alfabetu. Bogusław Lichoński stworzył program *PLindex* przeznaczony do sortowania w języku polskim z zachowaniem wszystkich liter występujących w alfabecie.

2.2.2 Sortowanie w innych językach

W jezyku angielskim litery są sortowane według alfabetu angielskiego. W odróżnieniu do języka polskiego duża litera występuje przed małą. Do sortowania w tym języku został stworzony program *Makeindex*, który w przypadku innych języków jest praktycznie bezużyteczny. Opracowano systemy oparte na TeX-u do specyficznych języków i alfabetów. Np.: JTeX dla japońskiego, sbTeX-xet dla hebrajskiego, cyrylica. Istnieje klika wersji do różnojęzycznych alfabetów łacińskich, np.: sbTeX , emTeX .

Do sortowania w w obu językach polskim i angielskim został stworzony *plmindex* przez Włodzimierza Macewicza

2.3 Opisy do symboli

Rozdział 3

Tworzenie spisu symboli

3.1 Implementacja

Implementację naszego pakietu zaczynamy od zdefiniowania opcji.

```
\newif\if@twocolumnindex
\newif\if@showsymidx

\DeclareOption{onecolumn}{\@twocolumnindexfalse}
\DeclareOption{twocolumn}{\@twocolumnindextrue}
\DeclareOption{showsymidx}{\@showsymidxtrue}
\DeclareOption{hidesymidx}{\@showsymidxfalse}

\ExecuteOptions{onecolumn, hidesymidx}
\ProcessOptions\relax
```

Wprowadzamy najpierw dwa pomocnicze makra warunkowe poleceniem `\newif`. Będzie w nich pamiętane jakie opcje zostały wybrane. Opcje deklaruje się przy pomocy makra `\DeclareOption`:

onecolumn – spis symboli będzie składany w jednej kolumnie tekstu,

twocolumn – spis symboli będzie składany w dwóch kolumnach tekstu,

showsymidx – w momencie wyboru symbolu do zapisania w indeksie symbol ten będzie wypisywany na marginesie; w ten sposób widać na podglądzie złożonego tekstu gdzie zostało użyte makro `\mathsymbol` opisane dalej,

hidesymidx – symbole nie będą wypisywane na marginesie.

Aby móc zrealizować opcję wypisywania spisu symboli w dwóch kolumnach potrzebujemy dodatkowy pakiet `multicol`. Podłączamy go przy pomocy makra `\RequirePackage` tak jak poniżej:

```
\RequirePackage{multicol}
```

Wprowadzamy teraz dwa parametry sterujące formatowaniem spisu symboli:

```
\newdimen\symbolwidth
\setlength{\symbolwidth}{0.25\hsize}

\newdimen\descwidth
\setlength{\descwidth}{0.65\hsize}

\newdimen\symboldescsep
\setlength{\symboldescsep}{12pt}
```

Makro `\symbolwidth` odpowiada za szerokość przestrzeni zarezerwowanej na wypisanie symbolu w spisie. Ustawione ono jest domyślnie na 1/4 szerokości wiersza tekstu na stronie. Makro `\descwidth` odpowiada za szerokość opisu symbolu i domyślnie przyjmuje wartość 65/100 szerokości wiersza tekstu. Parametr `\symboldescsep` decyduje o odstępie pomiędzy symbolem a jego opisem w indeksie. Te trzy wielkości mogą być modyfikowane z poziomu tekstu w celu zmiany formatowania.

Niżej zdefiniowane makro `\mathsymbol` jest podstawowym makrem naszego pakietu.

```
\newcommand{\mathsymbol}[2][]{%
    \ensuremath{#2}
    \addtocontents{sym}{%
        \string\indexsymline%
        {\noexpand\protect\noexpand #2}%
        {#1}%
        {\thepage}%
        {\@ifundefined{chapter}{\thesection}{\thechapter}}}
    \if@showsymidx
        \marginpar{\scriptsize\raggedright\ensuremath{#2}}
    \fi
}
```

Przy pomocy tego makra użytkownik mówi, które symbole mają trafić do spisu. Makro `\mathsymbol` pobiera dwa parametry: opis symbolu oraz sam symbol. Pierwszy argument jest opcjonalny i może być pominięty. Wówczas w spisie pojawi się tylko sam symbol bez opisu.

Makro działa w ten sposób, że najpierw wypisywany jest symbol, czyli drugi argument, w trybie matematycznym. Następnie przy pomocy makra `\addtocontents` (zob. podroz. D), na plik o rozszerzeniu `.sym`, zapisywane jest wywołanie makra `\indexsymline` z czterema argumentami. Te argumenty to kolejno: symbol matematyczny, jego opis, numer bieżącej strony

oraz numer bieżącego rozdziału lub sekcji w zależności od tego czy makro `\chapter` jest zdefiniowane w używanej klasie dokumentu, czy też nie. Makro `\indexsymline` odpowiada za sformatowanie pojedynczego wpisu w indeksie symboli.

Na koniec, o ile została użyta odpowiednia opcja pakietu, symbol wypisywany jest na marginesie.

Kolejnym ważnym makrem w naszym pakiecie jest `\indexofsymbols`. Na potrzeby tłumaczenia pakietu na inne języki niż angielski definiujemy pomocnicze makro `\indexsymname` zawierające nagłówek spisu symboli. W pakietach językowych może ono być odpowiednio zredefiniowane.

```
\newcommand\indexsymname{Index of Symbols}

\newcommand{\indexofsymbols}{%
    \@ifundefined{chapter}{%
        \section*{\indexsymname}%
    }{%
        \chapter*{\indexsymname}%
    }
    \@mkboth{\MakeUppercase\indexsymname}
            {\MakeUppercase\indexsymname}%
    \begingroup
        \parindent0pt
        \if@twocolumnindex
          \begin{multicols}{2}
              \@starttoc{sym}
          \end{multicols}
        \else
          \@starttoc{sym}
        \fi
    \endgroup
}
```

Działanie makra `\indexofsymbols` zaczyna się od rozpoznania, czy w danej klasie dokumentu mamy rozdziały. Jeśli nie, to rozpoczynamy nową sekcję w tytule, której umieszczamy nazwę spisu symboli. Sekcja nie będzie numerowana. Jeśli natomiast w dokumencie jest podział na rozdziały, to zamiast sekcji rozpoczynamy nowy rozdział, również bez numeru.

Następnie nazwę spisu symboli umieszczamy w nagłówkach stron (działanie makra `\@mkboth` opisane jest w podroz. D). Zastosowanie makra `\MakeUppercase` powoduje wypisanie nazwy wielkimi literami.

W kolejnym kroku, wypisywany jest spis symboli. Ponieważ, każdy wiersz spisu symboli jest osobnym paragrafem tekstu, to likwidujemy wcięcia paragrafów, tak aby zaczynały się one od lewego marginesu. Robimy to lokalnie w grupie, aby poza spisem symboli wcięcie pozostało bez zmian. W zależ-

ności od użytej opcji pakietu spis symboli zostanie wypisany w dwóch kolumnach, bądź w jednej kolumnie. Wypisanie spisu następuje poprzez wywołanie makra `\@starttoc` (zob. podroz. D). Przeczytany zostanie plik o rozszerzeniu `.sym`, który zawiera praktycznie cały indeks w postaci wywołań makra `\indexsymline` z odpowiednimi argumentami zawierającymi symbole, ich opisy itd. Innymi słowy, w tym miejscu zostanie zawołana seria makr `\indexsymline`.

```
\def\indexsymstyle#1{%
  \@ifundefined{is@#1}%
    {\undefinedindexsymstyle{#1}}
    {\expandafter
     \let\expandafter
     \indexsymline\csname is@#1\endcsname}%
}
```

Przy pomocy powyższego makra ustalamy styl formatowania indeksu symboli. Za formatowanie tego indeksu odpowiedzialne są makra postaci `\is@nazwa`, gdzie `nazwa` to nazwa konkretnego stylu. Właśnie ta nazwa jest argumentem makra `\indexsymstyle`. Jeśli odpowiednie makro formatujące jest zdefiniowane w pakiecie, to makro `\indexsymstyle` staje się synonimem makra `is@nazwa`. W przeciwnym razie zgłaszany jest błąd informujący, że żądany styl nie jest zdefiniowany.

W naszy pakiecie zdefinowaliśmy siedem różnych styli indeksów symboli. Ich implementację można zobaczyć w Dodatku A na stronie 21. Warto tylko wspomnieć, że zastosowaliśmy makra `\parbox` w celu uzyskania podziału indeksu symboli na kolumny ustalonej wcześniej szerokości. Makro definiujące styl *complex* zasługuje na dokładniejszy opis.

```
\newcommand{\complex@header}[1]{%
  \def\tmp@num{#1}
  \ifx\tmp@num\current@num
  \else
    \gdef\current@num{#1}
    \par
    \medskip
    {\bfseries
        \@ifundefined{chapter}{Section}{Chapter}
        \current@num}
    \par
    \smallskip
  \fi
}

\newcommand{\is@complex}[4]{%
```

```
  \complex@header{#4}
  \is@plain{#1}{#2}{#3}{}
}
```

Pomocnicze makro `\complex@header` ma za zadanie umieścić nagłówek rozdziału lub sekcji wraz z numerem. Jako jedyny argument pobiera ten właśnie numer. Zostaje on zapamiętany w globalnej zmiennej `\current@num` i w sytuacji kiedy się zmienia zostaje wypisany nagłówek rozdziału lub sekcji. Z tego makra korzystamy w trzech kolejnych definicjach. Tutaj przytaczamy tylko pierwszą z nich, czyli makro `\is@complex`. Po wywołaniu `\complex@header`, niezależnie od numeru rozdziału, czy sekcji, wypisywany jest symbol, jego opis i numer strony wystąpienia symbolu zgodnie z formatowaniem *plain*. Pozostałe dwie wersje styli powstają w analogiczny sposób.

3.2 Instrukcja użytkowania pakietu

Pakiet `symbols.sty` włącza się do dokumentu w standardowy sposób za pomocą `\usepackage`, na przykład:

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

Pakiet możemy podłączyć z opcjami: `onecolumn` lub `twocolumn` oraz `showsymidx` lub `hidesymidx`. Dwie pierwsze decydują o ilości kolumn w których będzie wypisywany indeks. Dwie pozostałe przełączają wypisywanie symboli na marginesie w miejscu dodania symbolu do spisu.

Aby dodać symbol do spisu należy użyć makro `\mathsymbol`. Jako opcjonalny argument pobiera ono opis symbolu do umieszczenia w indeksie. Wywołanie na przykład:

```
\mathsymbol[zbiór liczb rzeczywistych]{\mathbb{R}}
```

spowoduje umieszczenie w indeksie symbolu $\mathbb{R}$ wraz z podanym opisem. Oczywiście opis może być pominięty. Zakłada się, że symbol wymaga trybu matematycznego, dlatego możemy go podać bez deklaracji tego trybu tak jak w powyższym przykładzie. Makro `\mathsymbol` może wystąpić jako fragment bardziej skomplikowanej formuły, nawet jeżeli ta formuła jest ujęta w środowisku *equation* lub pochodnym.

Jeśli użyliśmy opicji `showsymidx` to w miejscu wystąpienia `\mathsymbol`, na marginesie, zostanie wypisany dodawany do indeksu symbol.

Makro `\mathsymbol` możemy używać z tym samym symbolem wielokrotnie tak aby w indeksie pojawiły się wszystkie numery stron użycia symbolu. Będziemy jednak musieli skorzystać ze skryptu `ismult.pl`.

Aby w danym miejscu dokumentu umieścić indeks symboli używamy makro `\indexofsymbols`. W dokumentach gdzie zdefiniowany jest rozdział, indeks symboli umieszczany jest jako nowy rozdział bez numeru. W pozostałych dokumentach indeks umieszczany jest jako sekcja bez numeru.

Pakiet dostarcza kilku styli formatowania indeksu. Aby zmienić styl należy użyć makro `\indexsymstyle` i jako obowiązkowy argument podać nazwę stylu. Możliwe są następujące style: *plain, rich, inverted, richinverted, dense, richdense, complex, complexinverted, complexdense*. Ich dokładny opis znajduje się w następnym podrozdziale.

3.3 Formatowanie spisu symboli

Poniżej przedstawimy siedem predefiniowanych styli indeksu symboli.

$\mathrm{Sub}_k(V)$	k-subspaces of V	1
$\mathbf{p}(H, B)$	k-pencil of k-subspaces	1
$\mathbf{P}_k(V)$	space of pencils	1
$\mathbf{A}_{k,m}(V, W)$	spine space	1
$\mathrm{Trc}\, D$	trace of D	3
$\mathrm{Ctr}\, D$	cotrace of D	3
$\simeq^{\tau}$	affine polygonal path	5

Rysunek 3.1: Formatowanie w stylu *plain*.

Styl *plain* (rys. 3.1), który jest stylem domyślnym naszego pakietu, formatuje wpis w indeksie symboli, w ten sposób, że najpierw wypisywany jest symbol, zostawiany jest odstęp poziomy wielkości `\symboldescsep`, wypisywany jest opis symbolu i na końcu numer strony wystąpienia symbolu wyrównany do prawego marginesu.

$\mathrm{Sub}_k(V)$	k-subspaces of V	1 (1)
$\mathbf{p}(H, B)$	k-pencil of k-subspaces	1 (1)
$\mathbf{P}_k(V)$	space of pencils	1 (1)
$\mathbf{A}_{k,m}(V, W)$	spine space	1 (1)
$\mathrm{Trc}\, D$	trace of D	3 (2)
$\mathrm{Ctr}\, D$	cotrace of D	3 (2)
$\simeq^{\tau}$	affine polygonal path	5 (2)

Rysunek 3.2: Formatowanie w stylu *rich*.

Styl *rich* (rys. 3.2) różni się od stylu *plain* tym, że na końcu w nawiasach podany jest rozdział lub sekcja, w której symbol wystąpił.

k-subspaces of V	$\mathrm{Sub}_k(V)$	1
k-pencil of k-subspaces	$\mathbf{p}(H, B)$	1
space of pencils	$\mathbf{P}_k(V)$	1
spine space	$\mathbf{A}_{k,m}(V, W)$	1
trace of D	$\mathrm{Trc}\, D$	3
cotrace of D	$\mathrm{Ctr}\, D$	3
affine polygonal path	$\simeq^\tau$	5

Rysunek 3.3: Formatowanie w stylu *inverted*.

W stylu *inverted* (rys. 3.3), w stosunku do stylu *plain*, zamieniono kolejność wypisywania symbolu i opisu. Analogicznie ma się styl *richinverted* (rys. 3.4) do stylu *rich*.

k-subspaces of V	$\mathrm{Sub}_k(V)$	1 (1)
k-pencil of k-subspaces	$\mathbf{p}(H, B)$	1 (1)
space of pencils	$\mathbf{P}_k(V)$	1 (1)
spine space	$\mathbf{A}_{k,m}(V, W)$	1 (1)
trace of D	$\mathrm{Trc}\, D$	3 (2)
cotrace of D	$\mathrm{Ctr}\, D$	3 (2)
affine polygonal path	$\simeq^\tau$	5 (2)

Rysunek 3.4: Formatowanie w stylu *richinverted*.

$\mathrm{Sub}_k(V)$ — k-subspaces of V, 1
$\mathbf{p}(H, B)$ — k-pencil of k-subspaces, 1
$\mathbf{P}_k(V)$ — space of pencils, 1
$\mathbf{A}_{k,m}(V, W)$ — spine space, 1
$\mathrm{Trc}\, D$ — trace of D, 3
$\mathrm{Ctr}\, D$ — cotrace of D, 3
$\simeq^\tau$ — affine polygonal path, 5

Rysunek 3.5: Formatowanie w stylu *dense*.

W stylu „zagęszczonym” *dense* (rys. 3.5) opis symbolu wypisywany jest zaraz po kresce za symbolem. Numer strony nie jest wyrównany do prawego marginesu, lecz wypisany po przecinku za opisem.

$\mathrm{Sub}_k(V)$ — k-subspaces of V, 1 (1)
$\mathbf{p}(H, B)$ — k-pencil of k-subspaces, 1 (1)
$\mathbf{P}_k(V)$ — space of pencils, 1 (1)
$\mathbf{A}_{k,m}(V, W)$ — spine space, 1 (1)
$\mathrm{Trc}\, D$ — trace of D, 3 (2)
$\mathrm{Ctr}\, D$ — cotrace of D, 3 (2)
$\simeq^\tau$ — affine polygonal path, 5 (2)

Rysunek 3.6: Formatowanie w stylu *richdense*.

Kolejna hybryda to styl *richdense* (rys. 3.6), a mianowicie połączenie stylu *dense* ze stylem *rich*.

Chapter 1

$\mathrm{Sub}_k(V)$	k-subspaces of V	1
$\mathbf{p}(H, B)$	k-pencil of k-subspaces	1
$\mathbf{P}_k(V)$	space of pencils	1
$\mathbf{A}_{k,m}(V, W)$	spine space	1

Chapter 2

$\mathrm{Trc}\, D$	trace of D	3
$\mathrm{Ctr}\, D$	cotrace of D	3
$\simeq^{\tau}$	affine polygonal path	5

Rysunek 3.7: Formatowanie w stylu *complex*.

W przypadku obszerniejszych wydawnictw, szczególnie w pracach zbiorowych, warto podzielić spis symboli ze względu na rozdziały. Zapewnia to styl *complex* (rys. 3.7). Jest on o tyle uniwersalny, że w przypadku, gdy klasa składanego dokumentu nie definiuje rozdziału, to podział robiony jest na sekcje.

Chapter 1

k-subspaces of V	$\mathrm{Sub}_k(V)$	1
k-pencil of k-subspaces	$\mathbf{p}(H, B)$	1
space of pencils	$\mathbf{P}_k(V)$	1
spine space	$\mathbf{A}_{k,m}(V, W)$	1

Chapter 2

trace of D	$\mathrm{Trc}\, D$	3
cotrace of D	$\mathrm{Ctr}\, D$	3
affine polygonal path	$\simeq^{\tau}$	5

Rysunek 3.8: Formatowanie w stylu *complexinverted*.

Dalej mamy dwa połączenia stylu *complex* ze stylami *inverted* oraz *dense*. Powstaje w ten sposób odpowiednio *complexinverted* (rys. 3.8) oraz *complexdense* (rys. 3.9).

Chapter 1

$\mathrm{Sub}_k(V)$ — k-subspaces of V, 1
$\mathbf{p}(H, B)$ — k-pencil of k-subspaces, 1
$\mathbf{P}_k(V)$ — space of pencils, 1
$\mathbf{A}_{k,m}(V, W)$ — spine space, 1

Chapter 2

$\mathrm{Trc}\, D$ — trace of D, 3
$\mathrm{Ctr}\, D$ — cotrace of D, 3
$\simeq^{\tau}$ — affine polygonal path, 5

Rysunek 3.9: Formatowanie w stylu *complexdense*.

Nie ma sensu łączyć stylu *complex* ze stylem *rich*, gdyż oba z nich oddają tę samą informację o rozdziale, tylko w inny sposób.

Przedstawione przykłady nie wyczerpują możliwości. Otóż, z jednej strony możliwe jest połączenie wymienionych styli z opcją **twocolumn**, z drugiej, możemy zastosować jeden z dodatkowych programów, tak aby uzyskać dodatkowe informacje w spisie lub inną kolejność występowania wpisów. Wspomniane programy wraz z przykładami ich zastosowania zostaną przedstawione w następnym rozdziale.

Rozdział 4

Dodatkowe funkcje pakietu

Tak jak wcześniej zostało napisane, spotyka się spisy symboli w książkach, takie że wielokrotne wystąpienie danego symbolu odnotowane jest poprzez podanie numerów stron rozdzielonych przecinkami. W naszym pakiecie dobrze by było mieć możliwość stworzenia takich spisów symboli. Po dłuższej analizie problemu doszliśmy do wniosku, że nie da się uzyskać opisywanej funkcjonalności na poziomie samego LaTeX-a. Musimy tutaj skorzystać z narzędzi zewnętrznych.

Inna funkcjonalność, która przydałaby się w naszym pakiecie to sortowanie spisu symboli według opisów a nie numerów stron wystąpienia symbolu. Tej funkcji również nie uda się zrealizować przy pomocy samego LaTeX-a.

Aby zrealizować wyszukiwanie powtórzeń symboli i sortowanie spisu wystarczy zmodyfikować odpowiednio plik pomocniczy `.sym` tworzony w trakcie przetwarzania LaTeX-em pracy matematycznej. Zdecydowaliśmy się, że zrobimy to przy pomocy Perl-a. Wybór Perl-a był dość oczywisty z uwagi na jego ogromne możliwości w przetwarzaniu różnych tekstów. Początkowo próbowaliśmy skanować plik `.sym` przy pomocy regularnych wyrażeń. Natknęliśmy się jednak na duże trudności związane z zagnieżdżonym grupowaniem poprzez nawiasy klamrowe ({}). Mianowicie nawiasy te grupują argumenty makr a jednocześnie mogą się pojawić wewnątrz tych argumentów. W związku z tym wycofaliśmy się z regularnych wyrażeń na rzecz gotowego parsera LaTeX::TOM. Jest to jedyny parser języka LaTeX jaki udało nam się znaleźć w bogatej bibliotece Perl-a. Całkowicie jednak wystarczył do realizacji naszych zadań.

4.1 Parser LaTeX-a w Perlu

Skrót TOM w nazwie modułu LaTeX::TOM oznacza TeX Object Model. Moduł ten został zaprojektowany do przetwarzania plików LaTeX-owych z nastawieniem na ekstrakcję zwykłego tekstu i jego modyfikację.

Moduł LaTeX::TOM dostarcza parser, który skanuje i interpretuje, chociaż

nie w pełni, dokumenty LaTeX-owe. Parser ten jako wynik skanowania zwraca reprezentacje dokumentu w postaci drzewa. To drzewo jest obiektem o nazwie LaTeX::TOM::Tree. Węzłami (ang. nodes) tego drzewa są obiekty o nazwie LaTeX::TOM::Node. Rozpoznawanych jest pięć konstrukcji logicznych LaTeX-a i stąd pięć możliwych typów węzłów. Są to:

TEXT – Ten typ węzła reprezentuje fragmenty zwykłego tekstu zawartego w dokumencie. Zawiera on wzory matematyczne oraz wszystko, co nie zostanie rozpoznane jako makro.

```
\label{etykieta}
```

Zostanie ono zidentyfikowane jako węzeł COMMAND, który będzie posiadał jednoelementowe poddrzewo posiadające węzeł typu TEXT, zawierający napis `etykieta`.

ENVIRONMENT – Otoczenia (środowiska) reprezentowane są jako węzły typu ENVIRONMENT. Zawierają one metadane na temat danego otoczenia oraz poddrzewo reprezentujące to, co zawarte jest w tym otoczeniu. Na przykład środowisko:

```
\begin{equation}
   r \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}
\end{equation}
```

zostanie zeskanowane jako węzeł ENVIRONMENT z nazwą `equation`. Związane z nim poddrzewo zawierać będzie rezultat skanowania powyższej formuły matematycznej.

GROUP – Obiekt tego typu jest poddrzewem przedstawiającym efekt skanowania tego co zostało umieszczone w nawiasach klamrowych ({}). W nawiasach tych umieszcza się fragmenty kodu LaTeX-owego, aby semantycznie odizolować je od reszty kodu. Jest to zachowywane przez parser LaTeX::TOM.

COMMENT – Węzeł COMMENT jest bardzo podobny do węzła typu TEXT. Różnica polega na tym, że zawiera on fragmenty tekstu poprzedzone znakiem %.

Moduł LaTeX::TOM dostarcza trzy obiekty, które teraz opiszemy. Skupimy się wyłącznie na metodach i własnościach, które wykorzystujemy w pracy.

LaTeX::TOM::Parser

Poza konstruktorem tego obiektu dostępne są dwie metody:

`parseFile(filename)` – Metoda ta czyta plik o nazwie `filename` i skanuje go zwracając obiekt LaTeX::TOM:Tree.

`parse(string)` – Ta metoda skanuje napis `string` i zwraca jako wynik obiekt LaTeX::TOM::Tree.

LaTeX::TOM::Tree

Opisywany tu obiekt reprezentuje strukturę dokumentu w postaci drzewa. Między innymi posiada następujące metody:

`print` – Wypisuje pełną strukturę drzewa w postaci tesktowej.

`toLaTeX` – Zwraca napis w języku LaTeX odpowiadający dla danego drzewa. Szczególnie przydatna metoda, gdy chcemy zapisać zmiany dokonane w dokumencie wejściowym.

`getCommandNodesByName(name)` – Zwraca tablicę adresów wszystkich węzłów typu COMMAND o nazwie `name`.

`getEnvironmentsByName(name)` – Zwraca tablicę adresów wszystkich węzłów typu ENVIRONMENT o nazwie `name`.

`getNodesByCondition(expression)` – Zwraca tablicę adresów wszystkich węzłów, które spełniają podane wyrażenie `expression`. Jest to bardzo wygodna metoda do przeszukania całego drzewa dokumentu i wyłapania interesujących nas fragmentów. Wyrażenie `expression` jest dowolnym, logicznym wyrażeniem w Perlu, co świadczy o dużych możliwościach tej metody. W wyrażeniu to odwołuje się do przetwarzanego węzła za pomocą zmiennej `$node`.

LaTeX::TOM::Node

Obiekt ten odzwierciedla poszczególne konstrukcje dokumentu LaTeX-owego zwane węzłami.

`getNodeType` – Zwraca typ węzła, czyli jedno z TEXT, COMMAND, ENVIRONMENT, GROUP, albo COMMENT.

`getNodeText` – Ta metoda może być użyta w kontekście TEXT albo COMMENT i zwraca tekst zawarty w odpowiednim węźle,

`setNodeText(text)` – Ustawia wartość węzłów tekstowych TEXT i COMMENT na podaną wartość `text`.

`getEnvironmentClass` – Dla otoczeń, czyli węzłów typu ENVIRONMENT zwraca nazwę otoczenia.

`getCommandName` – Zwraca nazwę makra. Ma oczywiście sens wyłącznie dla węzłów typu COMMAND.

`getChildTree` – Zwraca obiekt LaTeX::TOM::Tree będący drzewem weźłów „poniżej" danego węzła. Stosuje się wyłącznie do węzłów typu COMMAND, ENVIRONMENT lub GROUP.

`getFirstChild` – Zwraca obiekt LaTeX::TOM::Node będący pierwszym węzłem poddrzewa obiektów poniżej danego węzła. Metoda ta może być użyta dla węzłów typu COMMAND, ENVIRONMENT lub GROUP.

`getPreviousSibling` – Zwraca adres węzła znajdującego się bezpośrednio przed danym węzłem, na tym samym poziomie drzewa.

`getNextSibling` – Zwraca adres węzła znajdującego się bezpośrednio za danym węzłem, na tym samym poziomie drzewa.

`getParent` – Zwraca adres węzła bezpośrednio nad danym węzłem.

4.2 Wielokrotne wystąpienia symboli

Chapter 1		
$\mathrm{Sub}_k(V)$	k-subspaces of V	1,2
$\mathbf{p}(H, B)$	k-pencil of k-subspaces	1,3,7
$\mathbf{P}_k(V)$	space of pencils	1,5,8
$\mathbf{A}_{k,m}(V, W)$	spine space	1,6
Chapter 2		
$\mathrm{Trc}\, D$	trace of D	10,19,20
$\mathrm{Ctr}\, D$	cotrace of D	10,21
$\simeq^{\tau}$	affine polygonal path	11,14
$\mathbf{p}(H, B)$	k-pencil of k-subspaces	12
$\mathbf{A}_{k,m}(V, W)$	spine space	13

Rysunek 4.1: Formatowanie w stylu *complex* po wcześniejszym zaaplikowaniu programu `ismult.pl`.

Ten sam spis w stylu *rich* wyglądałby tak jak na rys. 4.2.

$\mathrm{Sub}_k(V)$	k-subspaces of V	1,2 (1)
$\mathbf{p}(H, B)$	k-pencil of k-subspaces	1,3,7,12 (1,2)
$\mathbf{P}_k(V)$	space of pencils	1,5,8 (1)
$\mathbf{A}_{k,m}(V, W)$	spine space	1,6,13 (1,2)
$\mathrm{Trc}\, D$	trace of D	10,19,20 (2)
$\mathrm{Ctr}\, D$	cotrace of D	10,21 (2)
$\simeq^{\tau}$	affine polygonal path	11,14 (2)

Rysunek 4.2: Formatowanie w stylu *rich* po wcześniejszym zaaplikowaniu programu `ismult.pl`.

4.3 Sortowanie spisu symboli

Chapter 1

k-pencil of k-subspaces	$\mathbf{p}(H, B)$	1,3,7
k-subspaces of V	$\mathrm{Sub}_k(V)$	1,2
space of pencils	$\mathbf{P}_k(V)$	1,5,8
spine space	$\mathbf{A}_{k,m}(V, W)$	1,6

Chapter 2

affine polygonal path	$\simeq^{\tau}$	11,14
cotrace of D	$\mathrm{Ctr}\, D$	10,21
k-pencil of k-subspaces	$\mathbf{p}(H, B)$	12
spine space	$\mathbf{A}_{k,m}(V, W)$	13
trace of D	$\mathrm{Trc}\, D$	10,19,20

Rysunek 4.3: Formatowanie w stylu *complexinverted* po wcześniejszym zaaplikowaniu programu `issort.pl`.

Dodatek A

Pakiet symbols.sty

```
%%
%%  File: symbols.sty
%%
%%  Author: Barbara Duchnowska
%%
%%  Title: Index of symbols
%%
%%  Create date: Mar 03, 2003
%%
%%  Last modified: Oct 13, 2006
%%
%%  Copyright (c) 2003 - 2006 Barbara Duchnowska and Mariusz Zynel.
%%
%%  This software is FREE. You can use and/or redistribute it for any
%%  purpose in either, modified, or unmodified form, under the terms of the
%%  GNU General Public License as published by the Free Software Foundation.
%%
%%  The above copyright notice and this permission notice shall be included
%%  in all copies or substantial portions of this software.
%%
%%  THIS SOFTWARE IS PROVIDED AS IS AND COME WITH NO WARRANTY OF ANY KIND,
%%  EITHER EXPRESSED OR IMPLIED. IN NO EVENT WILL THE COPYRIGHT HOLDER BE
%%  LIABLE FOR ANY DAMAGES RESULTING FROM THE USE OF THIS SOFTWARE.

\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{symbols}[2006/10/03 v0.1 Index of symbols (BD/MZ)]
\typeout{Index of symbols v0.1 <2006/10/03> (BD/MZ)}

\newif\if@twocolumnindex
\newif\if@showsymidx

\DeclareOption{onecolumn}{\@twocolumnindexfalse}
\DeclareOption{twocolumn}{\@twocolumnindextrue}
\DeclareOption{showsymidx}{\@showsymidxtrue}
\DeclareOption{hidesymidx}{\@showsymidxfalse}

\ExecuteOptions{onecolumn, hidesymidx}
\ProcessOptions\relax

\RequirePackage{multicol}


\newdimen\symbolwidth
\setlength{\symbolwidth}{0.25\hsize}

\newdimen\descwidth
```

```
\setlength{\descwidth}{0.65\hsize}

\newdimen\symboldescsep
\setlength{\symboldescsep}{12pt}

\newcommand{\mathsymbol}[2][]{%
    \ensuremath{#2}
    \addtocontents{sym}{%
        \string\indexsymline%
        {\noexpand\protect\noexpand #2}%
        {#1}%
        {\thepage}%
        {\@ifundefined{chapter}{\thesection}{\thechapter}}}
    \if@showsymidx
        \marginpar{\scriptsize\raggedright \ensuremath{#2}}
    \fi
}

\newcommand\indexsymname{Index of Symbols}

\newcommand{\indexofsymbols}{%
    \@ifundefined{chapter}{%
        \section*{\indexsymname}%
    }{%
        \chapter*{\indexsymname}%
    }
    \@mkboth{\MakeUppercase\indexsymname}
            {\MakeUppercase\indexsymname}%
    \begingroup
        \parindent0pt
        \if@twocolumnindex
          \begin{multicols}{2}
              \@starttoc{sym}
          \end{multicols}
        \else
          \@starttoc{sym}
        \fi
    \endgroup
}


\def\indexsymstyle#1{%
  \@ifundefined{is@#1}%
    {\undefinedindexsymstyle{#1}}
    {\expandafter\let\expandafter\indexsymline\csname is@#1\endcsname}%
}


\newcommand{\is@plain}[4]{%
  \parbox{\symbolwidth}{\makebox{$#1$}}\hskip\symboldescsep
  \parbox[t]{\descwidth}{#2}
  \hfill #3\par
}

\newcommand{\is@rich}[4]{%
  \parbox{\symbolwidth}{\makebox{$#1$}}\hskip\symboldescsep
  \parbox[t]{\descwidth}{#2}
  \hfill #3 (#4)\par
}

\newcommand{\is@inverted}[4]{%
  \parbox[t]{\descwidth}{#2}\hskip\symboldescsep
  \makebox{$#1$}
  \hfill #3\par
}
```

```
\newcommand{\is@richinverted}[4]{%
  \parbox[t]{\descwidth}{#2}\hskip\symboldescsep
  \makebox{$#1$}
  \hfill #3 (#4)\par
}

\newcommand{\is@dense}[4]{%
  \makebox{$#1$} --- #2, #3\par
}

\newcommand{\is@richdense}[4]{%
  \makebox{$#1$} --- #2, #3 (#4)\par
}

\newcommand{\complex@header}[1]{%
  \def\tmp@num{#1}
  \ifx\tmp@num\current@num
  \else
    \gdef\current@num{#1}
    \par
    \medskip
    {\bfseries \@ifundefined{chapter}{Section}{Chapter} \current@num}
    \par
    \smallskip
  \fi
}

\newcommand{\is@complex}[4]{%
  \complex@header{#4}
  \is@plain{#1}{#2}{#3}{}
}

\newcommand{\is@complexinverted}[4]{%
  \complex@header{#4}
  \is@inverted{#1}{#2}{#3}{}
}

\newcommand{\is@complexdense}[4]{%
  \complex@header{#4}
  \is@dense{#1}{#2}{#3}{}
}


\indexsymstyle{plain}

\newcommand*{\symbols@err}{\PackageError{Symbols}}

\def\undefinedindexsymstyle#1{\symbols@err{%
    Undefined index style: \protect#1
  }{%
    Change the index style.
  }}%

\endinput
%%
%% End of file `symbols.sty'.
```

Dodatek B

Skrypt ismult.pl

```
#!/opt/cfw/bin/perl
#
# Synopsis: Handle multiple symbol appearance in the index
#
# Last modified: Oct 18, 2006
#
# Copyright (c) 2006 Barbara Duchnowska & Mariusz Zynel
#
# This software is FREE. You can use and/or redistribute it for any purpose
# in either, modified, or unmodified form, provided that the above copyright
# notice and this permission are included in all copies or substantial
# portions of this software.
#
# THIS SOFTWARE IS PROVIDED AS IS AND COME WITH NO WARRANTY OF ANY KIND,
# EITHER EXPRESSED OR IMPLIED. IN NO EVENT WILL THE COPYRIGHT HOLDER BE
# LIABLE FOR ANY DAMAGES RESULTING FROM THE USE OF THIS SOFTWARE.

use strict;
use LaTeX::TOM;

(my $self = $0) =~ s/.*\///;

my (@tmpsym, @sym);

sub usage {
    print "usage: $self filename[.sym]\n";
    exit 1;
}

sub error {
    my ($msg) = @_;
    print "ERROR: $self: $msg\n";
    exit 1;
}

sub find_symbol {
    my ($symbol) = @_;
    for (my $i = 0; $i < scalar @tmpsym; $i++) {
        my $rec = $tmpsym[$i];
        if ($rec->{symbol} eq $symbol) {
            return $i;
        }
    }
    return -1;
}
```

```
if ($#ARGV != 0) {
    usage();
}

my $symfile = @ARGV[0];

if ($symfile !~ m/\.sym$/) {
    $symfile .= ".sym";
}

(my $latexfile = $symfile) =~ s/\.sym/.tex/;

my $parser = new Parser;
my $document = $parser->parseFile($latexfile);
my $indexsymstyles = $document->getCommandNodesByName("indexsymstyle");
my $indexsymstyle = @$indexsymstyles[0]->getChildTree->toLaTeX();

my $index = $parser->parseFile($symfile);
my $indexsymlines = $index->getCommandNodesByName("indexsymline");

my $lastsec;

foreach my $cmd (@$indexsymlines) {
    my $symbol = $cmd->getChildTree->toLaTeX();
    my $group = $cmd->getPreviousSibling->getPreviousSibling;
    my $desc = $group->getChildTree->toLaTeX();
    $group = $group->getPreviousSibling->getPreviousSibling;
    my $page = $group->getChildTree->toLaTeX();
    $group = $group->getPreviousSibling->getPreviousSibling;
    my $sec = $group->getChildTree->toLaTeX();
    if ($indexsymstyle eq "complex" && $lastsec != $sec) {
        push(@sym, @tmpsym);
        undef @tmpsym;
    }
    my $i = find_symbol($symbol);
    if (0 <= $i) {
        if ($tmpsym[$i]->{page} !~ /$page$/) {
            $tmpsym[$i]->{page} .= ", $page";
        }
        if ($tmpsym[$i]->{sec} !~ /$sec$/) {
            $tmpsym[$i]->{sec} .= ", $sec";
        }
    } else {
        push(@tmpsym, { "symbol" => $symbol,
                        "desc" => $desc,
                        "page" => $page,
                        "sec" => $sec });
        $lastsec = $sec;
    }
}

push(@sym, @tmpsym);

open(SYMFILE, "> $symfile")
            || error("Cannot open file for writing: $symfile: $!");

foreach my $sym (@sym) {
    printf(SYMFILE "\\indexsymline {%s}{%s}{%s}{%s}\n",
                        $sym->{symbol},
                        $sym->{desc},
                        $sym->{page},
                        $sym->{sec});
}

close(SYMFILE);
```

Dodatek C

Skrypt issort.pl

```
#!/opt/cfw/bin/perl
#
# Synopsis: Sort index of symbols
#
# Last modified: Oct 18, 2006
#
# Copyright (c) 2006 Barbara Duchnowska & Mariusz Zynel
#
# This software is FREE. You can use and/or redistribute it for any purpose
# in either, modified, or unmodified form, provided that the above copyright
# notice and this permission are included in all copies or substantial
# portions of this software.
#
# THIS SOFTWARE IS PROVIDED AS IS AND COME WITH NO WARRANTY OF ANY KIND,
# EITHER EXPRESSED OR IMPLIED. IN NO EVENT WILL THE COPYRIGHT HOLDER BE
# LIABLE FOR ANY DAMAGES RESULTING FROM THE USE OF THIS SOFTWARE.

use strict;
use LaTeX::TOM;

(my $self = $0) =~ s/.*\///;

my (@tmpsym, @sym);

sub usage {
    print "usage: $self filename[.sym]\n";
    exit 1;
}

sub error {
    my ($msg) = @_;
    print "ERROR: $self: $msg\n";
    exit 1;
}

sub by_desc {
    (my $x = $a->{desc}) =~ s/\$//g;
    (my $y = $b->{desc}) =~ s/\$//g;
    return $x cmp $y;
}


my $symfile = @ARGV[0];

if ($symfile !~ m/\.sym$/) {
    $symfile .= ".sym";
```

```
}

(my $latexfile = $symfile) =~ s/\.sym/.tex/;

my $parser = new Parser;
my $document = $parser->parseFile($latexfile);
my $indexsymstyles = $document->getCommandNodesByName("indexsymstyle");
my $indexsymstyle = @$indexsymstyles[0]->getChildTree->toLaTeX();

my $parser = new Parser;
my $index = $parser->parseFile($symfile);
my $indexsymlines = $index->getCommandNodesByName("indexsymline");

my $lastsec;

foreach my $cmd (@$indexsymlines) {
    my $symbol = $cmd->getChildTree->toLaTeX();
    my $group = $cmd->getPreviousSibling->getPreviousSibling;
    my $desc = $group->getChildTree->toLaTeX();
    $group = $group->getPreviousSibling->getPreviousSibling;
    my $page = $group->getChildTree->toLaTeX();
    $group = $group->getPreviousSibling->getPreviousSibling;
    my $sec = $group->getChildTree->toLaTeX();
    if ($indexsymstyle eq "complex" && $lastsec != $sec) {
        push(@sym, sort by_desc @tmpsym);
        undef @tmpsym;
    }
    push(@tmpsym, { "symbol" => $symbol,
                    "desc" => $desc,
                    "page" => $page,
                    "sec" => $sec });
    $lastsec = $sec;
}

push(@sym, sort by_desc @tmpsym);

open(SYMFILE, "> $symfile")
                || error("Cannot open file for writing: $symfile: $!");

foreach my $sym (@sym) {
    printf(SYMFILE "\\indexsymline {%s}{%s}{%s}{%s}\n",
                $sym->{symbol},
                $sym->{desc},
                $sym->{page},
                $sym->{sec});
}

close(SYMFILE);
```

Dodatek D

Zastosowane makra

- `\addtocontens`
  Komenda dodaje tekst(lub komendy formatujace) bezpośrednio do pliku generujacego spis treści, rysunków lub tablic.

  `\addtocontens{plik}{tekst}`

  Parametry komendy: *plik* to rozszerzenie nazwy pliku , na który ma być pisana informacja; *tekst*- informacja do zapisania.

- `\parbox`
  LaTeX tworzy stronę z pudełek które odpowiednio skleja. Polecenie `\parbox` składa tekst w pudełkach, w razie potrzeby dzieląc go na linijki.

  `\parbox[t]{szerokość}{tekst}`

  Wartość *t* określa jak ma znajdować się pudełko wzgledem otaczajacego go tekstu. Argument *szerokość* to wymiar określający wielkość pudełka.

- `\par`
  Polecenie to jest używane wtedy gdy chcemy zmienić stopień pisma lub interlinię.

- `\def`
  Służy do definiowania nowych poleceń (`\def{nowanazwa}`)

- `\hfil`
  Służy do automatycznego sterowania spacjowaniem. Powoduje ono umieszczenie całej dostepnej w wierszu wolnej przestrzeni w miejscu, w którym się pojawia.

- `\hfill`
  Działa podobnie jak `\hfil` tylko "rozpycha się" silniej w dostepnej wolnej przestrzeni.

- `\ensuremath`
  Uzywany jest przy pisaniu symboli matematycznych, upewnia sie ze tryb matematyczny jest włączony.

- `\string`
  Powoduje ze makro wystepujace za nim jako argument zostanie jedynie wypisane a nie wykonane.

- `\makeuppercase`
  Powoduje ze przekazany w argumencie tekst zostanie przekształcony na duże litery.

- `\bfseries`
  Służy do pisania pogrubioną czcionką.

- `\medskip, \smollskip`
  Oba makra dotyczą rozmiaru czcionki. Pierwsze: średni rozmiar, drugie: mały rozmiar liter.

- `\@mkboth`
  Ustala co ma być wypisane w nagłówkach strony. Posiada dwa argumenty. Pierwszy argument odpowiada za strony nieparzyste a drugi za strony parzyste.

- `\@startoc`
  Utworzenie pomocniczego pliku o nazwę za pomocą makra `\addtocontents`. Jeśli plik już istnieje to jest on wstawiany do dokumentu w miejscu wywłania `\@startoc`.

Bibliografia

[1] Lamport L., LaTeX *System przygotowania dokumentów*, ARIEL, 1992.

[2] Lichoński B., TeX *na indeksie*, Biuletyn GUST 1994 (3).

[3] Oetoker T., Partl H., Hyna I., Schlegl E., *Nie za krótkie wprowadzenie do systemu* LaTeX2e, 1998.

[4] Doob M., *Łagodne wprowadzenie do* TeX-a,2002.

[5] Borkowski M., LaTeX - *profesjonalny skład publikacji*, 1992.

[6] Rafajłowicz E., Myszko w.,LaTeX - *zaawansowane narzędzia*, PLJ, 1996.

[7] Rafajłowicz E., Myszko w., LaTeX - *podręcznik użytkownika*, PLJ, 1992.