Opakování operací a relací při zlomu řádku

Josef Tkadlec

Dojde-li ke zlomu řádku za operací či relací, pak by se příslušný znak měl na začátku řádku zopakovat. Je tedy zapotřebí předefinovat příslušné příkazy nebo znaky, aby byla tato podmínka splněna. Řešení zveřejnil ve Zpravodaji K. Horák [1]. Mně se nepodařilo některá makra v LaTeXu zprovoznit, takže jsem došel — za vydatného přispění P. Olšáka a J. Kubena, kterým tímto děkuji — k makrům jiným.

Odlišnost od řešení uvedeného K. Horákem je dvojí. Co se týče operací a relací zapisovaných příkazem, Horákova makra automaticky ukládají původní definici příkazu do příkazu předdefinovaného názvu a umožňují ji získat tím, že se předefinovanému příkazu předřadí speciální příkaz (`\original`). Níže uvedené makro je jednodušší, zato vyžaduje, aby se explicitně zadal příkaz, do kterého se původní definice uloží.

Co se týče operací a relací zapisovaných znakem, je základní přístup stejný. Zatímco však v Horákově článku byla pro každý požadovaný znak uvedena sada příkazů, zde uvádíme jedno sjednocující makro, které tuto úlohu zjednodušuje.

Můžeme tedy použít následující makra:

```
\def\OpakujPrikaz #1#2{\let #2=#1
  \def #1{#2\nobreak\discretionary{}{\hbox{$#2$}}{}}}
\def\OpakujZnak #1#2{\mathchardef #2=\mathcode‘#1
  \activedef #1{#2\nobreak\discretionary{}{\hbox{$#2$}}{}}
  \uccode‘\~=0 \mathcode‘#1="8000 }
\def\activedef #1{\uccode‘\~=‘#1 \uppercase{\def~}}
```

U prvních dvou příkazů označuje prvni parametr příkaz resp. znak popisující operaci nebo relaci, pro kterou chceme dosáhnout opakování, druhý parametr označuje příkaz, do kterého se uloží původní definice. Uložení původní definice je nutné ke správnému fungování maker, není tedy možné použít třeba stejný příkaz pro různé operace či relace. Použití maker ukazují následující příklady (pro ukládání původních definic byly zvoleny názvy příkazů odvozené od původního názvu nebo popisu znaku přidáním koncovky ORI):

```
\OpakujPrikaz {\leq}{\leqORI}
\OpakujZnak   <{\lessORI}
```

Je zapotřebí implicitně zakázat zlom řádku po operacích a relacích a je asi vhodné taková dělení výrazněji penalizovat. Použití může být následující:

```
\binoppenalty   =10000
\relpenalty     =10000
```

```
\exhyphenpenalty=1000
```

Nevýhodou je, že zlom řádku po předefinovaných operacích a relacích je penalizován stejně — zlom řádku po operacích by měl být penalizován více.

Známé problémy

- Makra nerozlišují, zda použitá operace nebo relace je binární (chceme ji při zlomu řádku zopakovat) nebo unární (zlom řádku je nepřípustný). Je tedy zapotřebí zakázat dělení předefinované operace nebo relace v místech, kde se vyskytuje jako unární symbol. To lze zajistit uzavřením do složených závorek — například symbol $-\infty$ po předefinování znaku „−“ píšeme `{-\infty}` (není to nutné v displaymath modu, kde nevynucený zlom řádku nehrozí).
- Příkazy `\neq` a `\ne` jsou definovány jako složenina `\not=` a v případě jejich předefinování výše uvedeným makrem by se při zlomu řádku na novém řádku objevilo rovnítko. Toto lze odstranit tím, že si nejprve předefinujeme tyto příkazy přirozenějším způsobem:

  ```
  \def\neq {\mathrel{\not=}}  \let\ne=\neq
  ```

- Při použití balíčku `amsmath` v LaTeXu je nutné případné předefinování znaků „−“ a „=“ provést až za `\begin{document}` nebo v parametru příkazu `\AtBeginDocument{}`. Je-li předefinován znak „−“, hlásí použití příkazu `\DeclareMathOperator` chybu — té se zbavíme odstraněním volaného vnitřního makra:

  ```
  \makeatletter
  \let\newmcodes@\relax
  \makeatother
  ```

Reference

[1] Horák, K.: *Sazba matematiky v českých textech.* Zpravodaj CSTUG **11** (2001), č. 1–3, str. 136–148.

Josef Tkadlec, `tkadlec@fel.cvut.cz`