Transformácia jednoduchého značkovacieho jazyka do HTML (domáce zadanie OS ZS 2021/22)

November 19, 2021

1 Zadanie

Vytvorte skript `markdown.sh`, ktorý bude na štandardnom vstupe očakávať vstup v značkovacom jazyku, ktorý je čiastočnou implementáciou jazyka markdown. Na výstupe bude produkovať HTML podľa doleuvedených pokynov.

Použitie bude teda napríklad

```
./markdown.sh < input.md > output.html
```

a výstup potom bude v jazyku HTML. Výstup si potom môžete otvoriť v nejakom internetovom prehliadači, napríklad

```
firefox output.html
```

2 Notácia

V doleuvedenom značí znak ␣ medzeru.

3 Požadovaný výstup

3.1 Hlavička

Na začiatku skript vypíše konštantný text

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-type" content="text/html;charset=UTF-8" />
</head>
<body>
```

3.2 Odstavce

Riadok, ktorý obsahuje iba medzery transformujte na riadok

```
<p>
```

3.3 Veľký nadpis

Ak riadok začína znakmi `#␣`, transformujte ho podľa pravidla

`#␣`*text* → `<h1>`*text*`</h1>`

Napríklad riadok

```
# Nadpis
```

sa transformuje na

```
<h1>Nadpis</h1>
```

3.4 Menší nadpis

Ak riadok začína znakmi `##␣`, transformujte ho podľa pravidla

`##␣`*text* → `<h2>`*text*`</h2>`

Napríklad riadok

```
## Podnadpis
```

sa transformuje na

```
<h2>Podnadpis</h2>
```

3.5 Zvýrazňovanie

Kdekoľvek na riadku sa nachádza reťazec typu `__`*text*`__`, [1] nahraďte ho podľa pravidla

`__`*text*`__` → `<strong>`*text*`</strong>`

Kdekoľvek na riadku sa nachádza reťazec typu `_`*text*`_`, nahraďte ho podľa pravidla

`_`*text*`_` → `<em>`*text*`</em>`

Pritom *text* nesmie obsahovať podčiarkovníky. Napríklad riadky

```
Toto je _zvýraznený text_ a toto _iný zvýraznený text_.
Silne zvýraznený text vyzerá __takto__.
```

sa transformujú na riadky

```
Toto je <em>zvýraznený text</em> a toto <em>iný zvýraznený text</em>.
Silne zvýraznený text vyzerá <strong>takto</strong>.
```

Dávajte si pozor na to, že riadok môže obsahovať viacero sekvencií horeuvedeného typu.

[1] Znak _ je podčiarkovník.

3.6 Pätička

Na konci skript vypíše konštantný text

```
</body>
</html>
```

4 Pomôcky a návody

Napíšem, ako som zadanie implementoval ja.

- Použil som idióm `while` – `read`

```
while read LINE
do
...
done
```

  Vo vnútri cyklu som potom mal k dispozícii postupne jednotlivé riadky *stdin* v premennej `LINE`.

- Testovanie, či obsah premennej `LINE` sedí s regulárnym výrazom `regex` som robil podľa vzoru

```
if echo "$LINE" | grep 'regex' > /dev/null
then
...
fi
```

  Tento idióm funguje takto:

  1. Konštrukt `if` ***príkaz*** testuje, či exit status *príkaz* je rovný 0.
  2. Exit status rúrovej sekvencie je exit status posledného príkazu v rúrovej sekvencii.
  3. Príkaz `grep 'regex'` má exit status 0 práve vtedy, keď nájde ten *regex* na svojom vstupe

4. sekvencia `> /dev/null` slúži na zrušenie normálneho výstupu príkazu `grep` – chceme iba testovať, nechceme aby niečo vypisoval.

- Používal som idióm

```
LINE=$(echo "$LINE" | sed 's@regex@replace@')
```

pre transformáciu obsahu premennej `LINE` pomocou `sed`

- Pripomínam, že vo `while` je možné používať `continue`.
- V `sed` som používal `@` miesto `/` na ohraničenie regulárneho výrazu, pretože HTML obsahuje `/`.
- V `sed` som používal spätnú referenciu cez `\1` v časti `replace`.
- V `sed` som používal flag `g`, to jest `'s@regex@replace@g'`, ak bolo treba.

5 Príklad vstupu a výstupu

Je v tomto repozitári; `example.md` je vstup a `example.html` je výstup.