Skriptovací jazyk BASH

Domácí příprava

Nastudujte syntaxi a základní příkazy skriptovacího jazyka BASH. Můžete použít i jiné zdroje.

Náplň cvičení

Cílem cvičení je seznámit se se skriptovacím jazykem *BASH* (Bourne Again Shell), pokročilejšími funkcemi příkazové řádky a dalšími nástroji UNIXového operačního systému.

Bash (nebo jiné skriptovací jazyky) používáme většinou na rychlou tvorbu jednoduchých programů, které automatizují různé akce a na které se nám nevyplatí vytvářet program ve vyšším jazyku. Ačkoliv se základní možnosti jazyka mohou zdát omezené, jeho síla spočívá v možnosti snadno spouštět jiné programy (binární nebo skripty), využívat tak jejich funkce a výstupy a kombinovat je do celků řešících i komplexní úlohy.

Zadání úlohy

Vytvořte skript v jazyce BASH (s příponou `.sh`), který:

- bude číst řádky ze standardního vstupu (až do signalizace konce souboru)
- pro každou řádku začínající řetězcem “PATH ” bude postupovat následovně:
  - Zbytek řádku za řetězcem “PATH ” bude považovat za cestu v souborovém systému
  - Cesty v souborovém systému na vstupu mohou být jak relativní, tak absolutní. Výstup skriptu bude obsahovat cesty tak, jak byly na vstupu.

    - Podle typu souboru na dané cestě skript vypíše jednu z následujících hlášek:
        - `FILE 'cesta/k/souboru' pocet_řádků_souboru '1._řádek_souboru'` pro existující běžný soubor (pokud je počet řádků 0, `1._řádek_souboru` bude prázdný řetězec)
        - `DIR 'cesta/k/adresari'` pokud se jedná o adresář
        - `LINK 'cesta/k/symlinku' 'cesta/k/cilovemu/souboru'` pokud se jedná o symbolický odkaz (včetně neplatných odkazů). Cestu k cílovému souboru můžete zjistit příkazem `readlink` bez přepínače.
        - `ERROR 'cesta/k/souboru'` pokud cesta neexistuje (nebo se nejedná o jeden z předchozích případů)
    - Hlášku `ERROR` vypisujte na standardní chybový výstup, ostatní na standardní výstup.
- všechny ostatní řádky ignoruje
- bude mít 2 nepovinné přepínače: `-h` a `-z`
    - přepínač `-h` způsobí vypsání stručné nápovědy ke skriptu a ukončení skriptu s návratovou hodnotou `0` ještě před čtením řádek ze standardního vstupu. Přepínače následující za `-h` budou ignorovány.

      > Opravdu stručnou! neopisujte tam prosím zadání úlohy!

    - přepínač `-z` na konci zabalí všechny soubory pro které skript vypsal řádku `FILE` (tedy symlinky ne) do archivu s názvem ‘`output.tgz`’. K vytvoření archivu použijte příkaz

      ```
      tar czf output.tgz file1 file2...
      ```

    - pořadí přepínačů může být libovolné
    - při zadání libovolného jiného přepínače skript skončí chybou (kromě případu, kdy je jiný přepínač za `-h` – viz výše)
- bude mít návratovou hodnotu
    - `0` pokud nedojde k žádné chybě a nebyla vypsána žádná řádka začínající `ERROR`
    - `1` pokud výstup obsahuje alespoň jeden řádek začínající `ERROR`

    - `2` pokud nastane jiná chyba (špatné argumenty, chyba při vytváření archivu, chyba při čtení souboru apod.). Při zjištění chyby této kategorie skript ihned skončí.

Dodržujte předepsané formátování výstupu (včetně apostrofů) kvůli automatickému vyhodnocování.

Při psaní skriptu se řiďte běžnými programátorskými zásadami, aby byl skript efektivní, jednoduchý (nekomplikovaný), přehledný a snadno pochopitelný.

Skript nahrajte do odevzdávacího systému, který automaticky zkontroluje jeho základní funkčnost. Výsledné body budou připsány cvičícím po následné manuální kontrole. (To znamená, že nám prosím nepište, pokud skript projde všemi testy a vy v BRUTE stále vidíte u úlohy 0 bodů!)

Ladění úlohy

Automatický vyhodnocovací skript v odevzdávacím systému nekontroluje striktně standardní chybový výstup, ale při detekci problému vám zobrazí vše, co bylo na chybový výstup posláno. Toho se dá využít k ladění skriptu:

- Příkaz

  ```
  set -x
  ```

  způsobí, že se na chybový výstup vypíše každý následující příkaz, který bash vykoná.

- Můžete si tam vypisovat i vaše poznámky. Např:

  ```
  echo "Ctu radku '$line'" >&2
  ```

- Specificky k této úloze: při ladění pozor, abyste nevypsali řádky typu

  ```
  ERROR 'nejaky-retezec-v-apostrofech'
  ```

  Ty by se pletly s očekávaným výstupem skriptu na `stderr`.

- Při ladění na svém počítači využívejte přesměrování vstupu nebo roury, abyste nemuseli opisovat vstup ručně:

```
echo PATH soubor.txt | ./script.sh
./script.sh < vstup.txt
```

- Zkuste používat nástroj ShellCheck, který vám pomůže vyvarovat se typických chyb. Můžete používat buď online verzi, nebo si ho nainstalovat lokálně a případně si ho integrovat do svého editoru.

Materiály

- Základy jazyka BASH

Domácí příprava na další cvičení

Nastudujte nástroje pro zpracování textu a použití regulárních výrazů.