**PROCESY V OS LINUX**

Skôr ako zaradíme túto tému do vyučovania je dôležité vedieť, aké má mať študent vedomosti a zručnosti. V prvom rade sú to základné pojmy, ako sú súbor, adresár, základné príkazy Linuxu (najmä cd, pwd a ls). Študent má disponovať základnou zručnosťou pri práci so systémom. Vie sa pohybovať po adresárovej (súborovej) štruktúre systému a vie zadávať základné príkazy do tzv. príkazového riadku. Ovládať základy tvorby skriptov nie je podmienkou. Základy skriptovania je, podľa nás, vhodné zaradiť práve do tejto témy. Treba si uvedomiť, že ak chceme pracovať s systéme, je potrebné ovládať základné programové konštrukcie ako sú sekvencie príkazov, vetvenia a cykly a mať základnú zručnosť v ich programovaní v ľubovoľnom programovacom jazyku, alebo prostredí.

**Príklady**

V tejto časti prezentujeme nielen príkazy a príklady, ale aj postup ako majú za sebou nasledovať. Okomentované príkazy a príklady tvoria výklad pre študenta a cieľom je, aby metodicky viedli učiteľa. Všetky spustené programy (procesy) sa nachádzajú v operačnom systéme Linux v priečinku /proc. Výpis urobíme pomocou príkazu ls /proc.

Každý proces v systéme má svoje meno (číslo). Je to jedinečné číslo. To znamená, že žiadny iný proces v systéme nie je označený takým istým číslom. Pomocou príkazu ps (angl. process status) vypíšeme stav procesov, ktoré sú v systéme. Keďže sa nachádzame v určitom adresári, zistíme stav, najmä čísla procesov, ktoré spúšťame. Z výpisu sme zistili, že príkaz ps vytvoril proces ps, ktorý je v systéme označený číslom 3132. Na výpise vidíme ešte proces bash, ktorý je označený číslom 32694.

Ak chceme získať zoznam všetkých procesov, ktoré sú v systéme napíšeme príkaz ps ax. Príkazy v systéme sú v skutočnosti vykonávateľné súbory. Po ich spustení rovnako aj vykonávaný príkaz v systéme vytvorí proces a vidíme ho vo výpise procesov. Preto príkaz ps, vytvorí proces ps.

Podobne vytvoríme proces aj spustením ľubovoľného programu. Program je vykonávateľný súbor, ktorého cieľom je po spustení riešiť určitú úlohu alebo vykonať daný výpočet. Kód programu môže byť napísaný v rôznych programovacích jazykoch. V našom prípade, vytvoríme program vo forme skriptu. Skript tvorí množina za sebou idúcich príkazov operačného systému.

Na obrázku 1 je uvedený kód pre skript využívajúci príkazy operačného systému Linux v príkazovom interpreteri BASH. BASH (z angl. Bourne Again Shell) je program, ktorý umožní vykonávať používateľom zadané príkazy operačného systému Linux.

1 #!/bin/bash

2

3 for x in $(seq 1 10000)

4 do

5 echo $x >> VystupnySubor

6 done

*Obrázok 1: Jednoduchý príklad skriptu využívajúceho cyklus s pevným počtom opakovaní*

Na obrázku 1 je jednoduchý skript, v ktorom sme použili len jeden cyklus s pevným počtom (10000) opakovaní. Program v cykle zapisuje čísla od 1 do 10000 do riadku tak, že čísla ukladá do špecifikovaného súboru VystupnySubor. Podstatnými časťami kódu na obrázku 1 sú príkazy v riadkoch 3 a 5. V riadku 3 je uvedená syntax príkazu cyklu FOR. Príkaz FOR slúži na vytvorenie cyklu s pevným počtom opakovaní. V našom prípade premenná x postupne (sekvenčne) nadobúda hodnoty 1 až 10000, čo je definované vo výraze $(seq 1 10000), pričom prvé číslo označuje počiatočnú hodnotu premennej x a druhé číslo konečnú hodnotu tejto premennej. Pre každú hodnotu x sa vykonajú všetky príkazy uvedené medzi riadkami 4 a 6. V našom príklade sa na riadku 5 nachádza príkaz echo $x >> VystupnySubor, ktorý postupne vypisuje (príkaz echo) hodnoty premennej x (vyberáme ich pomocou $x). Pretože chceme, aby sa premenné zapisovali do súboru, použijeme znaky presmerovania a meno výstupného súboru (>> VystupnySubor). Skript, obsahujúci kód z obrázku 1, sme uložili pod názvom NazovSkriptu. Skript (program) spustíme rovnako ako každý program, t.j. pomocou dvojice znakov ./ a jeho názvu.

Teraz chceme zobraziť naše procesy, ktoré bežia v systéme, preto použijeme príkaz ps počas behu nášho skriptu. Na obrázku 2 vidíme príkaz pre spustenie tohto skriptu na pozadí a zároveň spustenie príkazu ps.

./NazovSkriptu & ps

*Obrázok 2: Spustenie skriptu NazovSkriptu na pozadí a vypísanie práve bežiacich procesov*

Keďže beh nášho skriptu trvá len niekoľko sekúnd, pri vypísaní aktuálne bežiacich procesov môžeme vidieť aj novovzniknutý proces s názvom NazovSkriptu. Po ukončení práce skriptu môžeme znova spustiť príkaz ps, pričom vidíme, že proces s názvom NazovSkriptu už viac nie je medzi aktívnymi procesmi.

Po skončení aplikácie (programu) sa vytvorený proces ukončí a je vymazaný z /proc a tiež uvoľní pridelenú časť pamäte. Ďalším cieľom resp. úlohou učiteľa je, aby žiak vedel meniť parametre procesov (pozri výkonový štandard).

Preto rozšírime náš skript (obrázok 1) o prácu s parametrami, ktoré môžu meniť beh skriptu. Parameter je hodnota, ktorú zadá používateľ pri spúšťaní skriptu a táto hodnota bude v skripte použitá tak, že sa priradí do premennej. Ak zadáme jeden parameter, jeho hodnota sa uloží do premennej s názvom $1. Ak zadáme za menom skriptu viac parametrov, budú sa postupne priraďovať do premenných $1, $2, $3, ... . Aby sme študentov naučili používať parametre skriptu, jednoducho upravíme skript z obrázku 1 tak, aby program nepočítal striktne do 10 000, ale po také číslo, aké mu používateľ zadá pri spúšťaní. Zmena v tele skriptu je uvedená na obrázku 3. Kde systém za premennú $1 dosadí prvú zadanú hodnotu pri spúšťaní skriptu.

1 #!/bin/bash

2

3 for x in $(seq 1 $1)

4 do

5 echo $x >> VystupnySubor

6 done

*Obrázok 3: Úprava skriptu NazovSkriptu, ktorý nazveme NazovUpravenehoSkriptu*

Okrem premenných, ktoré zadal používateľ ($1, $2, $3, ...) sa v skriptoch bežne využívajú aj špeciálne premenné, ktoré nie je potrebné pri spúšťaní zadávať. Medzi najvyužívanejšie premenné tohto typu patrí najmä $0 a $#. Premenná $0 vždy obsahuje názov spúšťaného skriptu, v ktorom je použitá. Premenná, označená ako $# obsahuje počet parametrov zadaných používateľom pri spustení skriptu. Obe špeciálne premenné využívame v príklade na obrázku 4.

1 #!/bin/bash

2

3 echo Nazov tohto suboru je $0

4 echo Pocet parametrov je $#

5 echo Parameter 1 ma hodnotu $1

6 echo Parameter 2 ma hodnotu $2

*Obrázok 4: Skript využívajúci štandardné aj špeciálne parametre*