Systemy operacyjne

wykład 5 - BASH - zmienne tablicowe, pętla FOR, argumenty wywołania skryptu

dr Marcin Ziółkowski

Instytut Matematyki i Informatyki
Akademia im. Jana Długosza w Częstochowie

14 kwietnia 2016 r.

ZMIENNE TABLICOWE

Jak każdy język programowania BASH ma wbudowaną możliwość operowania na tablicach (listach). Tablice te są tablicami dynamicznymi, co oznacza, że możemy na przykład zmieniać ich rozmiar, wstawiać nowe elementy czy usuwać wybrane elementy. W BASHU zwykle określamy tablicę poprzez podanie jej elementów, natomiast poprzez indeks możemy odwołać się do dowolnego elementu tej tablicy lub nawet całej tablicy, jeśli użyjemy jako indeksu znaku * lub **@**:

```
a=(1 2 3 4 5)
echo ${a[0]}
echo ${a[1]}
echo ${a[2]}
echo ${a[*]}
echo ${a[@]}
```

ZMIENNE TABLICOWE

Tablicę możemy też tworzyć poprzez dopisywanie kolejnych elementów. Dopisywanie kolejnych elementów jest dość intuicyjne i wykonuje się je poprzez operację podstawienia, natomiast usuwanie elementów wykonujemy, używając instrukcji **unset**. Oto przykład:

```
a[0]=1
a[1]=b
a[2]=2
a[3]=c
echo ${a[*]}
a[4]=7
a[5]=9
echo ${a[*]}
unset a[4]
echo ${a[*]}
```

ZMIENNE TABLICOWE

Oczywiście możemy odwoływać się do wybranych elementów tablicy i wykonywać na nich operacje. Jeśli operacje te wymagają przejrzenia wszystkich elementów tablicy, możemy do tego celu wykorzystać pętlę WHILE lub UNTIL. Wygodne jest tu również odwołanie się do długości tablicy, które realizujemy poleceniem

`${#a[*]}`.

Oto przykład skryptu obliczającego sumę elementów tablicy:

SKRYPT - SUMA ELEMENTÓW TABLICY

```
a=(1 2 3 4 5 6 7 8)
i=0
suma=0
while [ $i -lt ${#a[@]} ]
do
  echo ${a[i]}
  suma=$[suma+${a[i]}]
  i=$[i+1]
done
echo "Suma elementów tablicy wynosi $suma"
```

PĘTLA FOR - SCHEMAT

Bezpośrednio z tablicami (listami) powiązana jest instrukcja iteracyjna FOR, której schemat jest następujący:

SCHEMAT PĘTLI FOR

```
for zmienna in LISTA
do
  instrukcje
done
```

Oto prosty przykład zastosowania tej instrukcji:

```
for i in 1 2 3 4 5 6 7 8
do
  echo $i
done
```

PĘTLA FOR

Oczywiście pętli iteracyjnej FOR możemy użyć też w odniesieniu do listy plików z danej lokalizacji, co oczywiście jest bardzo praktycznym jej zastosowaniem. Oto przykład skryptu, który wypisuje nazwy wszystkich plików w bieżącym katalogu, które mają rozszerzenie **.pdf**.

SKRYPT - WYPISYWANIE PLIKÓW

```
for i in *.pdf
do
  echo $i
done
```

ARGUMENTY SKRYPTU

Wszystkie do tej pory omawiane skrypty były wywoływane bez dodatkowych argumentów. Można jednak w interpreterze BASH wywoływać skrypty z ustaloną liczbą argumentów. W tym celu w interpreterze poleceń dla argumentów skryptu są zarezerowane specjalne stałe:

- `$0`
  jest stałą oznaczającą nazwę skryptu
- `$1, $2, $3, $4...`
  są stałymi oznaczającymi kolejne argumenty skryptu
- `$@`
  jest stałą oznaczającą listę wszystkich argumentów skryptu
- `$#`
  jest stałą oznaczającą liczbę wszystkich argumentów skryptu

ARGUMENTY SKRYPTU

Oto przykład prostego skryptu, który dodaje dwie liczby będące dwoma argumentami skryptu. Oczywiście w danym przypadku wywołanie skryptu wygląda następująco: **bash skrypt liczba1 liczba2**.

SKRYPT - DODAWANIE

```
echo "$1 + $2 = $[$1 + $2]"
```

A oto skrypt, za pomocą którego dodamy więcej liczb:

SKRYPT - DODAWANIE WIELU LICZB

```
suma=0
for i in $@
do
  suma=$[suma+i]
done
echo "suma wynosi $suma"
```

INSTRUKCJA SHIFT I PĘTLA WHILE

Instrukcja **SHIFT** służy do zdejmowania(usuwania) argumentów skryptu. Argumenty zdejmowane są w kolejności od pierwszego do ostatniego. Instrukcja shift pozwala na pisanie łatwych skryptów, w których zamiast domyślnej pętli FOR używamy pętli WHILE. Oto przykład pierwszego skryptu napisanego z jej wykorzystaniem.

SKRYPT - WYPISYWANIE ARGUMENTÓW

```
while [ $# -gt 0 ]
do
    echo $1
    shift
done
```

INSTRUKCJA SHIFT I PĘTLA WHILE

A oto skrypt, za pomocą którego dodajemy liczby wprowadzane jako argumenty (tym razem z pętlą while):

SKRYPT - DODAWANIE WIELU LICZB

```
suma=0
i=1
while [ $i -le $# ]
do
   suma=$[suma+$1]
   shift
done
echo "suma wynosi $suma"
```

ARGUMENTY SKRYPTU - C.D

Oto przykład skryptu, który wypisuje ciąg liczbowy od liczby oznaczającej wartość pierwszego argumentu skryptu do liczby oznaczającej drugi argument skryptu. Tym razem wykorzystamy pętlę WHILE.

SKRYPT - WYPISYWANIE LICZB

```
i=$1
while [ $i -le $2 ]
do
  echo $i
  i=$[i+1]
done
```

PĘTLA FOR A ARGUMENTY SKRYPTU - INNA WERSJA

Istnieje jeszcze jedna specjalna wersja pętli for używana w połączeniu z argumentami skryptu. Jej składnia jest następująca:

INNA WERSJA PĘTLI FOR

```
for zmienna
do
  instrukcje
done
```

Zmienna zastępuje tu listę wszystkich argumentów skryptu.

PĘTLA FOR A ARGUMENTY SKRYPTU - INNA WERSJA

Oto prosty przykład zastosowania - skrypt obliczający iloczyn wprowadzonych jako argumenty skryptu liczb.

SKRYPT - MNOŻENIE LICZB

```
iloczyn=1
for zmienna
do
  iloczyn=$[iloczyn*$zmienna]
done
echo "Iloczyn liczb wynosi $iloczyn"
```

BASH - INNE MOŻLIWOŚCI

Na koniec tego wykładu warto podkreślić, że podczas omawiania interpretera BASH zwróciliśmy uwagę tylko na najważniejsze aspekty programowania. BASH jak każdy język programowania posiada inne użyteczne konstrukcje i możliwości m.in:

- instrukcje CASE oraz SELECT pozwalające na tworzenie rozbudowanych instrukcji warunkowych lub tworzenia MENU
- możliwość tworzenia własnych FUNKCJI, również funkcji rekurencyjnych
- możliwość wykorzystania wbudowanego generatora liczb pseudolosowych - funkcja RANDOM
- możliwość tworzenia okien dialogowych - wykorzystanie instrukcji DIALOG