TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN HÀ NỘI BÀI THỰC HÀNH LINUX
KHOA TOÁN – CƠ – TIN HỌC **Bài 05**

----------oOo----------

Phần 1: Thực hành

1. Phạm vi biến trong shell

Trong shell có hai loại biến: Biến cục bộ và biến toàn cục

- Biến cục bộ: Chỉ có thể truy cập bởi shell hiện thời
- Biến toàn cục: Có thể truy cập bởi shell hiện thời và tất cả các tiến trình con của shell này.

Ví dụ1:

Cho nội dung của tệp *myvar.sh* như sau

```
#!bin/bash
echo “MYVAR is: $MYVAR”
MYVAR=“hi there”
echo “MYVAR is: $MYVAR"
```

Kết quả:

MYVAR is:
MYVAR is: hi there

Ví dụ 2:

```
student@linux$ MYVAR=”Hello”
student@linux$ bash myvar.sh
```

Kết quả:
MYVAR is:
MYVAR is: hi there

Dùng lệnh *export* để đặt một biến là biến toàn cục, nó có thể được kế thừa bởi một chương trình khác bao gồm một shell script

Cú pháp: export var1, var2, …, varN

Ví dụ 3:

```
student@linux$ MYVAR=”Hello”
student@linux$ export MYVAR student@linux$
gedit myvar1.sh
```

Gõ nội dung của tệp tin *myvar1.sh* như sau:

```
!#/bin/bash
echo "MYVAR is: $MYVAR"
MYVAR="hi there"
echo "MYVAR is: $MYVAR"
```

sau đó chạy file vừa tạo

student@linux$ bash myvar1.sh

Kết quả:

MYVAR is:hello

MYVAR is: hi there

Để xóa giá trị vừa thiết lập cho biến sử dụng lệnh **unset.**

Ví dụ: *unset MYVAR*

2. Một số biến đặc biệt trong lập trình shell

$0	Tên file của script hiện tại
$n	Những biến tương ứng với các tham số mà một script được gọi. Ở đây n là một số thập phân dương ứng với vị trí của tham số (tham số đầu tiên là $1, tham số thứ 2 là $2,…)
$#	Số tham số tương ứng với một script
$* hoặc $@	Tất cả các đối số được trích dẫn, nếu một script gọi đến hai đối số, thì $* tương ứng với $1 $2
$?	Trạng thái exit của câu lệnh cuối cùng thực hiện

3. Lệnh exit và exit status

Các lệnh exit thường dùng để kết thúc một script, giống như trong chương trình C. Mỗi lệnh exit trả về một exit status. Trong Linux, khi một lệnh hoặc một script thực hiện, nó trả về hai giá trị. Nếu lệnh thực thi thành công sẽ trả về 0, ngược lại sẽ trả về khác 0 (thông thường là một mã lỗi).

Tương tự như vậy, một hàm trong một script sẽ trả về một exit status. Lệnh cuối cùng thực hiện trong hàm sẽ xác định exit status. Để biết được một giá trị trả về của một lệnh hay script, dùng biến đặc biệt có sẵn của shell là : **$?**. Sau khi một hàm trả về, **$?** sẽ đưa ra exit status của lệnh được thực hiện cuối cùng của hàm đó. Đây là cách trả về một giá trị trong một hàm của Linux shell.

Ví dụ 4:

```
echo Hello
echo $?
abeckdf
echo $?
```

4. Hàm trong lập trình Shell

Hàm trong lập trình shell có cú pháp:

```
function-name()
{
        command1
        command2
        ...
        commandN
        return
}
```

Gọi hàm:

```
function-name arg1 arg2 arg3 argN
```

Ví dụ 5:

```
#!/bin/bash sayHello()
{
        echo "Hello $LOGNAME, Toi la An!"
}
sayHello
```

Ví dụ 6:

```
cal()
{
        n1=$1
        op=$2
        n2=$3
        ans=0
        if [ $# -eq 3 ]
         then
              ans=$(( n1 $op n2 ))
               echo "$n1 $op $n2 =$ans"
        else
              echo "Function cal requires at least three args"
        fi

}
cal 5 + 10
cal 10 - 2
cal 10 / 2
```

5. **Trả về giá trị từ một hàm**

Nếu muốn trả về giá trị cho hàm, dùng biến toàn cục để lưu lại giá trị hoặc *echo* in ra giá trị cần trả về trong hàm và dùng lệnh $(function_name) để lấy lại giá trị đó.

Ví dụ 7:

```
#!/bin/bash
sum()
{
 var=0
        for((i = 1; i<=10; i++))
         do
          var=`expr $var + $i`
        done
        echo $var
}
tong=$(sum)
echo "Tong la $tong"
```

Kết quả: Tong la 55

Ngoài ra, có thể dùng lệnh *return* để trả về giá trị của một hàm. Thông thường giá trị trả về của hàm được lưu trữ trong biến *$*?

Ví dụ 8:

```
#!/bin/bash
sum()
{
 var=0
 for i in 1 2 3 4
 do
                var=`expr $var + $i`
 done
 return $var
}
sum
echo "Tong la $?"
```

Một hàm cũng có thể gọi lại chính nó (đệ quy) hoặc có thể gọi đến các hàm khác.

Ví dụ 9:

```
#!/bin/sh
number_one ()
{
        echo "This is the first function speaking..."
         number_two
}

number_two ()
{
       echo "This is now the second function speaking..."
}
```

6. Biến cục bộ và biến toàn cục trong một shell script

1. Biến cục bộ: Chỉ có hiệu lực trong hàm. Để khai báo biến cục bộ, dùng từ khóa local
2. Biến toàn cục: Có hiệu lực tại tất cả các hàm trong cùng script. Nếu không có từ khóa local, các biến sẽ được coi là biến toàn cục

Trường hợp đã có biến toàn cục, nhưng trong hàm lại khai báo biến cục bộ cùng tên, biến cục bộ sẽ được ưu tiên và có hiệu lực cho đến khi hàm đó chấm dứt.

Ví dụ 10: Viết một script có nội dung như sau:

```
#!bin/bash
sample_text=“global variable”
 foo() {
        local sample_text=“local variable”
        local sample_text1=“local variable 1”
        echo “sample_text = $sample_text”
}
echo “sample_text = $sample_text”
foo
echo “sample_text = $sample_text”
echo “sample_text1 = $sample_text1”
```

Kết quả:

sample_text = global variable
sample_text = local variable
sample_text = global variable
sample_text1 =

Phần 2: Bài tập thực hành

Lưu ý: Thiết kế chương trình theo hàm

1. Viết 1 shell cho phép nhận đối dòng lệnh là 1 số nguyên (>0), sau đấy in ra kết quả là số đó nhân với các số từ 1→10. Ví dụ
   $./multiply 7

7x1 = 7

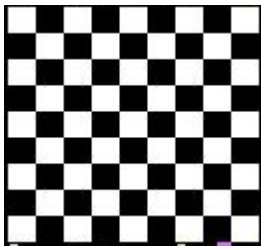
7x2 = 14

…

7x10 = 70

2. Viết 1 shell cho phép nhận đối dòng lệnh là một số nguyên (n>0), sau đó tính giá thực hiện các yêu cầu sau và in kết quả ra màn hình:
   a) Tính tổng S = 1 + 2 + 3 +....+ *n*.
   b) Tính tổng các số chẵn nhỏ hơn *n*.
   c) Tính tổng các số lẻ nhỏ hơn *n*.

3. Viết 1 shell cho phép nhận đối dòng lệnh là một số nguyên (n>0), sau đó thực hiện các yêu cầu sau:
   a) Tính $S_1 = n!$
   b) Tính $S_2 = 1! + 2! + \ldots + n!$
4. Viết 1 shell tìm dòng có độ dài lớn nhất trong một tập tin
5. Viết shell thực hiện in ra bàn cờ vua như sau:



Gợi ý:

Để đặt màu chữ dùng cú pháp: echo –e "\033[35m
text" Trong đó các màu chữ như sau:

30	Đen
31	Đỏ
32	Xanh lá cây
33	Nâu
34	Xanh nước biển
35	Hồng
36	Xanh da trời
37	Xám

Để đặt màu nền cho chuỗi kết xuất hoặc cho màn hình terminal dùng cú pháp: echo –e –n "\033[40m" để đặt nền đen

Trong đó các màu nền như sau:

40	Đen
41	Đỏ
42	Xanh lá cây
43	Nâu
44	Xanh nước biển
45	Hồng
46	Xanh da trời
47	Trắng

6. Viết 1 shell trong đó có hàm sum() trả về tổng các đối số truyền vào của nó. In tổng vừa tính được ra màn hình.
7. Viết 1 shell trong đó chứa hàm count() có đối số truyền vào là tên của một thư mục, và trả về số lượng file trong thư mục đó.