1. 함수, 다중 조건문

1) 함수

"함수는 쉘 프로그램 내에서 실행할 블록을 지정하고,
필요 시 수시로 호출하여 사용하는 것을 의미"

✔ 함수를 사용하는 경우 변수의 영향 범위를 고려하여야 함

전역변수

전체 쉘 프로그래밍 영역에서
사용되는 변수

지역변수

함수 내부에서만 사용되는 변수

1) 함수

예 test9-1.sh

```
#!/bin/sh
str="I am global variable"
foo()
{
    local str=" I am local variable"
    echo "This is in function"
    echo $str
}
echo "start shell-program"
echo $str
foo
echo $str
str="I am global variable2"
echo $str
foo
echo $str
exit 0
```

〈출처 : 교수자 제작물〉

1) 함수

예 test9-1.sh

```
root@ubuntu:/home/shtest/test09# sh test9-1.sh
start shell-program
I am global variable
This is in function
I am local variable
I am global variable
I am global variable2
This is in function
I am local variable
I am global variable2
root@ubuntu:/home/shtest/test09#
```

〈출처 : 교수자 제작물〉

2) 다중 조건문

“ case 다중 조건문이 일반적임 ”

case 조건/a)../b)../…/esac

조건이 a)일 경우 a)문장을, b)일 경우 b)문장을 실행함

- ✔ 다중의 조건을 간단히 표현
- ✔ 각 조건의 종료를 **;;(세미콜론 두 개)**로 표현
- ✔ a, b, c의 조건을 표현하는 다양한 방법은 예제를 통하여 학습함

2) 다중 조건문

예 test9-2.sh

```
#!/bin/sh
echo "Do you love me?"
read ans
case $ans in
       [yY][eE][sS] ) echo Thank you. ;;
       [nN]*) echo I am very sad. ;;
       *) echo really? ;;
esac
exit 0
```

〈출처 : 교수자 제작물〉

2) 다중 조건문

```
root@ubuntu:/home/shtest/test09# sh test9-2.sh
Do you love me?
abc
really?
root@ubuntu:/home/shtest/test09# sh test9-2.sh
Do you love me?
YeS
Thank you.
root@ubuntu:/home/shtest/test09# sh test9-2.sh
Do you love me?
yse
really?
root@ubuntu:/home/shtest/test09# sh test9-2.sh
Do you love me?
Na
I am very sad.
root@ubuntu:/home/shtest/test09# sh test9-2.sh
Do you love me?
No
I am very sad.
root@ubuntu:/home/shtest/test09#
```

〈출처 : 교수자 제작물〉

2) 다중 조건문

예 test9-3.sh

```
#!/bin/sh
echo "test"
read ans
case $ans in
        [1]) echo "case 1"; echo "case 1-2"; echo "case 1-3";;
        [2]) echo "case 2";;
        *) echo "case 3";;
esac
exit 0
```

〈출처 : 교수자 제작물〉

2) 다중 조건문

예 test9-3.sh

```
root@ubuntu:/home/shtest/test09# sh test9-3.sh
test
1
case 1
case 1-2
case 1-3
root@ubuntu:/home/shtest/test09# sh test9-3.sh
test
2
case 2
root@ubuntu:/home/shtest/test09# sh test9-3.sh
test
3
case 3
root@ubuntu:/home/shtest/test09#
```

〈출처 : 교수자 제작물〉

2. 형식을 주어 출력

1) printf 사용

%s	문자를 출력
%d	숫자를 출력
%.5s	문자열이 길더라도 앞에서 5개만 출력
₩n	개행문자, 줄을 바꾸라는 의미

1) printf 사용

예 test9-4.sh

```
#!/bin/sh
printf "%s₩n" hello
printf "%s₩n" hello world
printf "%s " hello
printf "%s" world
printf "₩n"
printf "%s %d %.3s₩n" hello `expr 1 + 2 ` hello
exit 0
```

〈출처 : 교수자 제작물〉

1) printf 사용

예 test9-4.sh

```
root@ubuntu:/home/shtest/test09# sh test9-4.sh
hello
hello
world
hello world
hello 3 hel
root@ubuntu:/home/shtest/test09#
```

〈출처 : 교수자 제작물〉

3. 주기적으로 처리

1) while, sleep 활용

while :, do , done	do~done 블록을 무한 반복함

- 반복을 중지시키기 위해 ctrl + c를 사용

sleep	화면이 빠른 속도로 스크롤 되는 것과, 특정한 주기시간을 부여하기 위하여 사용함

1) while, sleep 활용

예 test9-5.sh

```
#!/bin/sh
icnt=0
while :
do
    A=`date`
    echo [ $icnt ] "******" $A
    icnt=`expr $icnt + 1`
    sleep 1
done
```

〈출처 : 교수자 제작물〉

1) while, sleep 활용

예 test9-5.sh

```
root@ubuntu:/home/shtest/test09# sh test9-5.sh
[ 0 ] ****** Tue Jan 19 21:30:34 KST 2018
[ 1 ] ****** Tue Jan 19 21:30:35 KST 2018
[ 2 ] ****** Tue Jan 19 21:30:36 KST 2018
[ 3 ] ****** Tue Jan 19 21:30:37 KST 2018
[ 4 ] ****** Tue Jan 19 21:30:38 KST 2018
^C
root@ubuntu:/home/shtest/test09#
```

〈출처 : 교수자 제작물〉

2) 직접 입력

- ✔ 한 파일의 크기를 주기적으로 체크하는 예제
- ✔ 쉘 프로그램을 만들지 않고 명령어를 나열, 입력하여 사용할 수도 있음
- ✔ 만일 큰 파일이 복사되고 있는 중이라면 시간 간격으로 모니터링 가능

2) 직접 입력

```
root@ubuntu:/home/shtest/test09# while :
> do
> ls -l test9-5.sh
> echo --------------------
> sleep 1
> done
-rw-r--r-- 1 root root 111 Jan 19 21:30 test9-5.sh
--------------------
-rw-r--r-- 1 root root 111 Jan 19 21:30 test9-5.sh
--------------------
-rw-r--r-- 1 root root 111 Jan 19 21:30 test9-5.sh
--------------------
^C
root@ubuntu:/home/shtest/test09#
```

〈출처 : 교수자 제작물〉

4. 쉘 명령어, 변수 조건문 실습

5. 반복문, 조건의 AND-OR 실습

학습활동

* 일시정지 버튼을 클릭하고 학습활동에 참여해 보세요.

Q 다양한 쉘 프로그래밍의 응용사례 실습을 바탕으로 putty를 통하여 접속한 사람을 주기적으로 체크하는 쉘을 만들어 보세요.

※ 학습활동에 대한 해설

Q 다양한 쉘 프로그래밍의 응용사례 실습을 바탕으로 putty를 통하여 접속한 사람을 주기적으로 체크하는 쉘을 만들어 보세요.

A

- 리눅스 서버에 ssh를 통하여 putty 프로그램 등으로 접속하면, 접속하는 단말 당 하나씩 bash라는 프로세스가 생성됩니다.
- 우리는 ps – ef| grep bash를 통하여 이러한 상황을 알 수 있습니다.
- 이때 출력되는 줄 수를 세어본다면, 즉 ps-ef|grep bash|wc-l을 통하여 개수를 얻을 수 있습니다.

※ 학습활동에 대한 해설

Q 다양한 쉘 프로그래밍의 응용사례 실습을 바탕으로 putty를 통하여 접속한 사람을 주기적으로 체크하는 쉘을 만들어 보세요.

A

- 그렇다면 아래와 같이 쉘을 구성하면 됩니다.

```
while:
do
  ps -ef|grep bash|wc -l
  echo ==============
  sleep 5
done
```

- 다음 장에 보다 자세히 다루도록 하겠습니다.