조건문 & 반복문

조건문(Conditional Constructs)

IF 문

```
if test-commands; then
    consequent-commands;
[elif more-test-commands; then
    more-consequents;]
[else alternate-consequents;]
fi
```

test-commands이 실행되고 반환 상태가 0이면 consequent-commands 이 실행된다.
test-commands가 0이 아닌(non-zero) 상태를 반환하면 각 elif 목록이 차례로 실행되고 종료 상태가 0이면 일치하는 more-consequents가 실행되고 조건문이 완료된다.
if 와 각 elif 절이 0이 아닌 종료 상태가 있는 경우 else 절의 alternate-consequents가 실행된다.
반환 상태는 마지막으로 실행된 명령의 종료 상태이거나 참으로 테스트된 조건이 없으면 0이다.

CASE 문

```
case word in
    [ [(] pattern [| pattern]...) command-list ;;] ...
esac
```

case 문은 단어와 일치하는 첫 번째 패턴에 해당하는 명령 목록을 실행한다.
'|'는 여러 패턴을 구분하는 데 사용되며 ')' 연산자는 패턴 목록을 종료한다.
각 문장은 ';;', ';&' 또는 ';;&'로 끝나야 한다.
단어는 일치를 시도하기 전에 물결표 확장(tilde expansion), 파라미터 확장(parameter expansion), 명령 대체(command substitution), 산술 확장(arithmetic expansion), 따옴표 제거(quote removal)를 거친다.
'*' 문자를 사용해 항상 일치하는 기본 케이스를 정의하는 것이 일반적인 관용구이다.

```
echo -n "Enter the name of an animal: "
read ANIMAL
echo -n "The $ANIMAL has "
case $ANIMAL in
  horse | dog | cat) echo -n "four";;
  man | kangaroo ) echo -n "two";;
  *) echo -n "an unknown number of";;
esac
echo " legs."
```

반복문(Looping Constructs)

UNTIL 문

```
until test-commands;
do
    consequent-commands;
done
```

테스트 명령의 종료 상태가 0이 아닌 한 계속 명령을 실행한다.
반환 상태는 결과 명령에서 실행된 마지막 명령의 종료 상태이거나 test-commands조건에 만족하지 않아 실행되지 않은 경우 0이다.

```
#!/bin/sh

a=10

until [ $a -lt 10 ]
do
    echo $a
    a=`expr $a + 1`
done
```

WHILE 문

```
while test-commands;
do
    consequent-commands;
done
```

test-commands의 종료 상태가 0이면 결과 명령을 실행한다.
반환 상태는 결과 명령에서 실행된 마지막 명령의 종료 상태이거나 test-commands조건에 만족하지 않아 실행되지 않은 경우 0이다.

```
#!/bin/sh
a=0

while [ $a -lt 10 ]
do
    echo $a
    if [ $a -eq 5 ] then
        break
    fi
    a=`expr $a + 1`
done
```

FOR 문 형식1

```
for name [ [in [words ...] ] ; ] do
    commands;
done
```

목록에 있는 단어(words)들을 변수(name)에 담아 한번씩 실행한다. -> Python의 for문과 동일함
반환 상태는 실행되는 마지막 명령의 종료 상태이다. 단어 목록에 항목이 없으면 명령이 실행되지 않으면 반환 상태는 0이다.

FOR 문 형식2

```
for (( expr1 ; expr2 ; expr3 )) ;
do
    commands ;
done
```

다른 프로그래밍 언어의 for문과 같다. expr1, expr2, expr3은 산술표현식이다.
expr1은 초기 한 번만 평가되고, expr2는 0이 될 때 까지 반복적으로 평가된다.
expr2가 0이 아닌 값으로 평가될 때마다 명령을 실행하고 expr3이 평가된다.
만약 어떤 식을 생략하면 그 식은 1로 평가되는 것처럼 작동한다.
반환 값은 실행된 명령에서 마지막 명령의 종료 상태이거나 식 중 하나라도 유효하지 않은 경우 false이다.

```
#!/bin/sh

for var1 in 1 2 3
do
    for var2 in 0 5
    do
        if [ $var1 -eq 2 -a $var2 -eq 0 ]
        then
            break 2
        else
            echo "$var1 $var2"
      fi
   done
done

출력 결과
1 0
1 5
```

continue [n]
둘러싸는 for, while, until 또는 select 루프의 다음 반복을 재개한다.
n을 지정하면 n번째 둘러싸는(nth enclosing) 루프의 실행이 재개된다.
break [n]
둘러싸는 for, while, until 또는 select 루프를 종료한다.
n이 제공되면 n번재 둘러싸는 루프가 종료된다.
exit [n]
셸을 종료하고 셸의 부모에게 n상태를 반환한다. n이 생략되면 종료 상태는 실행된 마지막 명령의 상태이다.
EXIT의 모든 트랩(trap)은 셸이 종료되기 전에 실행된다.