シェルプログラミング 制御

[シェルプログラミング]

- if,elif,else
- for
- while
- until
- case

if, elif, else

条件の判定とステートメントの実行

```
if condition
then
   statements
elif condition
then
   statements
else
   statements
fi
```

例

```
#!/bin/sh

echo "Are you over 18years old ?"
read isAdult

if [ $isAdult = "yes" ];then
  echo "Welcom! enter the site."
elif [ $isAdult = "no" ];then
  echo "Sorry. you shuld go http://yahoo.co.jp"
else
  echo "Sorry $isAdult no recognized. Enter yes or no"
  exit 1
fi

exit 0
```

for

- 範囲の値を繰り返し操作
- 値には任意の文字列を指定
- 値はあらかじめプログラムに記述しておくことも、シェルによるファイル名の展開結果等を利用することも可能

```
for variable in values
do
  statements
done
```

例

```
#!/bin/sh

for fruits in apple berry banana
do
  echo $fruits
done

exit 0
```

while

・条件が真の間ループが実行される
・回数がわからない場合などの繰り返し
・指定回数の繰り返しを簡単に

```
while condition
do
  statements
done
```

例 - 繰り返し回数不明

```
#!/bin/sh

echo "Enter password"
read password

while [ "$password" != "pass" ]; do
  echo "password isn't match. try again"
  read password
done

exit 0
```

例 - 指定回数繰り返し

```
#!/bin/sh

i=0

while [ "$i" -le 10 ];do
  echo $i
  i=$(($i+1))
done

exit 0
```

until

・while ステートメントの条件テストを逆にしたもの
・条件が真になるまでループが実行される

```
#!/bin/sh

until who | grep "$1" > /dev/null
do
  sleep 60
done

echo "$1 has just logged in"

exit 0
```

case

・変数の内容をパターンと比較し結果に応じて処理を実行できる
・複数のパターン指定と複数の実行が可能
・ユーザ入力を調べるのに適している

```
case variable in
  pattern [ | pattern] ...) statements;;
```

```
    pattern [ | pattern] ...) statements;;
    ...
esac
```

上記はこの本からの覚書。

非常にわかりやすく説明されている良書