MEMPELAJARI DASAR-DASAR SHELL SCRIPT DAN MEMBUAT SHELL SCRIPT UNTUK PEMECAHAN PERMASALAHAN DAN MENJALANKANNYA

09

EDISI I - 2007

MATA DIKLAT **:**
SISTEM OPERASI

PROGRAM KEAHLIAN **:**
SEMUA PROGRAM KEAHLIAN

DEPARTEMEN PENDIDIKAN NASIONAL
BIRO PERENCANAAN DAN KERJASAMA LUAR NEGERI

2007

MEMPELAJARI DASAR-DASAR SHELL SCRIPT DAN MEMBUAT SHELL SCRIPT UNTUK PEMECAHAN PERMASALAHAN DAN MENJALANKANNYA

EDISI I - 2007

9.1 PENDAHULUAN

9.1.1 Deskripsi

NAMA MODUL	**:**	**SISTEM OPERASI**
KOMPETENSI	**:**	**MEMPELAJARI DASAR-DASAR SHELL SCRIPT DAN MEMBUAT SHELL SCRIPT UNTUK PEMECAHAN PERMASALAHAN DAN MENJALANKANNYA**
SUB KOMPETENSI	**:**	**Pemrograman Shell**
KRITERIA KERJA	**:**	**-**
LINGKUP BELAJAR	**:**	• Menggunakan struktur case-esac. • Looping dengan while, for dan do while. • Membuat fungsi. • Latihan.

9.2 Menggunakan Struktur Case-Esac

Perintah **case** memungkinkan untuk melakukan sejumlah tindakan berbeda terhadap sejumlah kemungkinan nilai. Bentuk perintah ini :

Case nilai in
Pola_1) perintah1;;
Pola_2) perintah2;;
Pola_3) perintah3;;
...
esac

Pada bentuk di atas :

- Masing-masing perintah dapat berupa satu atau beberapa perintah
- *Perintah_1* dijalankan kalau nilai cocok dengan pola_1. setelah *Perintah_1* dijalankan, eksekusi dilanjutkanke akhri **case (esac).**
- *Perintah_2* dijalankan kalau nilai coock dengan *pola_2*. setelah *Perintah_2* dijalankan, eksekusi dilanjutkan ke akhir **case (esac)**
- *Perintah_3* dijalankan kalau nilai cocok dengan *pola_3*. setelah *Perintah_3* dijalnakan, eksekusi dilanjutkan ke akhir **case (esac)**

Contoh penerapan **case** yaitu untuk menentukan tindakan terhadap suatu pilihan seperti ditunjukkan pada skrip berikut :

```
# cat menu.sh
:
# ----------------------------------------------------
# menu.sh
#
# Contoh pemakaian case
#     untuk menangani pilihan menu
# ----------------------------------------------------

Clear # haups layar

Echo “MENU MAKAN SIANG :”
echo “”

echo “1. Nasi – Soto Ayam Nikmat”
echo “2. Nasi – Sop Buntut Enak Betul”
echo “3. Nasi – Kare Ayam Gurih”

echo “”
echo –n “pilihan Anda (1,2,3) : “
read pilihan

echo “ “
```

```
case #pilihan in
1)              echo “pilihan Anda : Nasi – soto Ayam Nikmat)
echo “Sebentar lagi akan dihidangkan”
;;
2)              Echo “Pilihan Anda: Nasi – Sop Buntut Enak Betul”
Echo “Sebentar lagi akan dihidangkan”
;;
3)              Echo “Pilihan Anda : Nasi – Kare Ayam gurih”
Echo “Sebentar lagi akan dihidangkan”
;;
4)              Echo “Pilihan Anda tidak dimengerti”
Echo “Silahkan pilih-pilih dulu””
;;

Esac
#_
```

Pada contoh ini, pola * berarti kalau pilihan bernilai selain 1,2 atau 2. contoh berikut menunjukkan hasil eksekusi skrip menu.sh:

 # menu.sh

```
MENU MAKAN SIANG :
1.              Nasi – Soto Ayam Nikmat
2.              Nasi – Sop Buntut Enak Betul
3.              Nasi – Kare Ayam Gurih

Pilihan Anda (1, 2, 3) : 1<Enter>
Pilihan Anda : Nasi – Soto Ayam Nikmat
Sebentar lagi akan dihidangkan
#_
```

Adapun contoh skrip berikut digunakan untuk memberikan komentar pilihan yang dimasukkan benar atau salah. Contoh ini sekaligus menunjukkan pemakaian simbol | pada bagian pola yang berarti “atau”

 # cat pilih.sh

```
# ---------------------------------------------------------
# pilih.sh
#
# Contoh validasi pilihan
# ---------------------------------------------------------

echo –n “pilihan (1,2,3) : “
read pilihan

case #pilihan in
  1|2|3) echo “pilihan benar”;;
       *) echo “pilihan salah”;;
```

```
Esac
#_
```

9.3 Looping Dengan While, For Dan Do While

Seringkali kita melakukan suatu pengulangan proses. Hal seperti ini dapat ditangani dengan mudah dengan menggunakan perintah-perintah pengulangan. Pada Bourne shell, Bourne Again shell, dan Korn shell perintah-perintah berikut berguna untuk melakukan pengulangan proses:

- for
- until
- while

Pada C shell:

- foreach
- while
- repeat

Catatan :
Pada C shell pengulangan proses juga bisa dibentuk melalui goto.

Perintah for pada Bourne Shell, Bourne Again Shell, dan Korn Shell

Perintah **for** memungkinkan sejumlah perintah dapat dieksekusi berkali-kali untuk setiap nilai yang terletak dalam suatu daftar. Bentuk perintah ini:

for *indeks* [in *daftar_ argumen]*
do
  perintah
done

Secara bergantian, *indeks* akan berisi nilai-nilai yang tercantum pada *daftar argumen.* Untuk masing-masing nilai tersebut, *perintah* akan dijalankan. Sebagai contoh:

```
# cat for1.sh
:
# ----------------------------------------------------
# for1.sh
#
# Contoh pemakaian for dengan memproses
#     perintah untuk setiap nilai
#     yang ada pada daftar yang mengikuti
#      kata in
# ----------------------------------------------------

for nama in "Abu Nawas" "KSatria gaja Hitam" superman
do
```

```
  echo #nama
done
#_
```

Hasil eksekusi skrip di atas:

```
# forl.sh
 Abu Nawas
 Ksatria Baja Hitam
 Superman
#_
```

Dengan melibatkan *wildcard* (misalnya *) pada bagian *daftar argumen* (setelah kata *in), for* dapat dipakai untuk memperoleh nama-nama berkas yang cocok pada suatu direktori. Contoh:

```
# cat for2.sh
:
# ------------------------------------------------
# for2.sh
#
# contoh pemakaian for untuk mendapatkan
#       nama-nama berkas pada direktori
#       /usr/bin yang berawalan b
#       dengan panjang nama 5 karakter
# ------------------------------------------------

for nama_berkas in /usr/bin/b????
do
  echo "Nama lengkap : #nama_berkas"
  echo "rvama berkas . 'basename #nama_berkas
  echo ""
done
#_
```

Contoh eksekusi skrip di atas:

```
# for2.sh
Nama lengkap : /usr/bin/batch
Nama berkas   : batch

Nama lengkap : /usr/bin/bison
Nama berkas : bison

Nama lengkap : /usr/bin/burst
Nama berkas : burst

Nama lengkap : /usr/bin/byacc
```

Nama berkas : bycat

Nama lengkap : /usr/bin/bzcat
Nama berkas : bzcat

Nama lengkap : /usr/bin/bzip2
Nama berkas : bzip2
#_
Pada contoh berikut, bagian in dan yang mengikutinya tidak disertakan:

```
# cat for3.sh
:
# ----------------------------------------------------------
# for3.sh
#
# Contoh pemakaian for untuk mendapatkan
#         nama-nama parameter posisi
# ----------------------------------------------------------

I= 0

for parameter
do
  i='expr #i + 1'       # Naikkan i sebesar 1
  echo "#i. #parameter"
done
#_
```

Dengan cara seperti ini, *parameter* akan berisi parameter posisi.
Perlu diketahui, operasi aritmetika pada Bourne shell (khususnya) biasa dilakukan dengan menggunakan utulitas expr. Agar hasil expr dikirimkan ke variabel maka expr perlu ditulis dalan tanda *bacquote* (‘).

Contoh eksekusi skrip fo r 3. s h :

```
# for3.sh merapi merbabu muria
1. merapi
2. merbabu
3. muria
#_
```

Perintah foreach pada C shell

Perintah serupa **for** pada C shell berupa **foresach**. Bentuk pemakaiannya :
Foreach *indeks (daftar_argument)*
  Perintah
End
% cat forl1.csh

```
# ----------------------------------------------------
# for1.csh
#
# Contoh pemakaian foreach dengan memproses
#    perintah untuk setiap nilai
#    yang ada pada daftar yang mengkut
#    kata in
# ----------------------------------------------------

Foreach nama (“Abu Nawas” “Ksatria Baja Hitam” Superman)
Echo #nama
End
#_
```

Sedangkan bentuk skrip pada C shell yang serupa dengan for2.sh adalah seperti di bawah ini:

% cat forl2.csh

```
# ----------------------------------------------------
# for2.csh
# Contoh pemakaian foreach untuk mendapatkan
#       nama-nama berkas pada direktori
#       /usr/bin yang berawalan b
#       dengan panjang nama 5 karakter
# ----------------------------------------------------

foreach nama_berkas ( /usr/bin/b???? )
    echo "tvama lengkap : #nama_berkas"
    echo "Nama berkas . _basename #nama_berkas""
    echo ""
end
_%
```

Adapun bentuk skrip for3. sh dapat diimplementasikan pada C shell seperti berikut:

% cat for3.csh

```
# ----------------------------------------------------
# for3.csh
#
# Contoh pemakaian foreach untuk mendapatkan
#          nama-nama parameter posisi
# ----------------------------------------------------

@ i = 0
Foreach parameter ( #argv [*] )
   @ i + = 1 # Naikkan i sebesar 1
   echo "#i.  #parameter"
end
%_
```

Perintah until pada Bourne Shell, Bourne Again Shell, dan Korn Shell

Perintah **until** digunakan untuk mengulang suatu proses hingga kondisi pengulangan bernilai benar. Bentuk pemakaiannya:

Until *perintah_ koncfisi*
do
  perintah
done

contoh skrip yang menggunakan **until :**

```
# ----------------------------------------------------
# until.csh
#
# Contoh pemakaian until untuk membaca
#       data bilangan dari keyboard
# ----------------------------------------------------

bil=xxx

until echo #bil | grep – v “ [^0-9]” > /dev/null
do
  echo “Masukkan sebuah bilangan bulat positif”
  rad bil
done
#_
```

Pernytaan kondisi pada skrip di atas tampak kompleks :
Echo #bil | grep –v “[^0-9]” > /dev/null

Baris pipa di atas akan memberikan nilai keluar tidak sama dengan nol kalau terdapat karakter yang bukan berupa digit.

Sontoh eksekusi :

```
# until.sh
  Masukkan sebuah bilangan bulat positif
  -5<Enter>
  Masukkan sebuah bilangan bulat positif
  0.5<Enter>
  Masukkan sebuah bilangan bulat positif
  23<Enter>
#_
```

Perintah while pada Bourne Shell, Bourne Again Shell, dan Korn Shell

Perintah **while** pada Bourne / Bourne Again / Korn shell mempunyai fungsi serupa dengan **until**. Perbedaannya pengulangan pada **while** dilakukan selama kondisi bernilai benar. Format :

while *perintah_kondisi*
do
  perintah
done

Contoh skrip yang menggunakan **while**

cat while.sh

```
 :
 # ------------------------------------------------------
 # while.sh
 #
 # contoh pemakaian while untuk
 #     menampilkan bilangan genap kurang dari 20
 # ------------------------------------------------------

 bil = 0
 while test #bil –1t 20
 do
        echo #bil
        bil=’espr #bls  + 2’   # naikkan bil sebesar 2
 done
#_
```

Contoh eksekusi :

while.sh

```
 0
 2
 4
 6
 8
 10
 12
 14
 16
 18
#_
```

Perintah while pada C Shell

C shell jug mempunyai perintah **while,** Format :
while *(ungkapan)*
perintah
end

contoh skirp pada C shell yang serupa dengan while.sh:

%**cat while.sh**
#
 # ----------
while.csh
#
contoh pemakaian while untuk
menampilkan bilangan genap kurang dari 20

```
@ bil = 0
while (#bil < 20)
        echo #bil
        @ bil   + =  2   # Naikkan bil sebesar 2
end
%_
```

contoh eksekusi
while.sh
0
2
4
6
8
10
12
14
16
18
#_

Perintah repeat pada C shell

C shell mempunyai perintah **repeat** yang berguna untuk mengulang suatu perintah. Format pemakaiannya:
Repeat *jumlah perintah*
Dalam hal ini, *perintah* akan diulang sebanyak *numlah kali.*

Contoh:

%**repeat 5 echo “Halo”**
halo
halo
halo
halo
halo
%_

9.4 Membuat Fungsi

9.5 Latihan