**LAPORAN PRAKTIKUM 8**

**Pemrograman Shell 2**

Semester 2 Tahun Akademik 2022/2023



**DOSEN:**

**Fitri Setyorini ST., M.Sc**

**PENYUSUN:**

**Arsyita Devanaya Arianto (3122500008)**

**PROGRAM STUDI VOKASI**

**D-III TEKNIK INFORMATIKA**

**DEPARTEMEN TEKNIK INFORMATIKA DAN KOMPUTER POLITEKNIK ELEKTRONIKA NEGERI SURABAYA**

**2023**

# Praktikum 8

**Pemrograman Shell 2**

**TUGAS PENDAHULUAN**

**Percobaan 1 : Instruksi Test**

1. Menggunakan instruksi test, perhatikan spasi antara

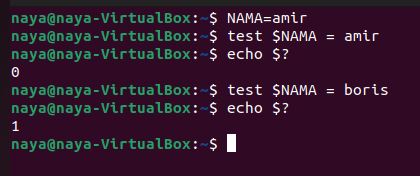
$ NAMA=amir

$ test $NAMA = amir

$ echo $?

$ test $NAMA = boris

$ echo $?



* Variabel NAMA berisi amir
* Memanggil variabel NAMA menggunakan perintah test dengan isi variabel $NAMA = amir
* Status exit bernilai 0, karena isi variabel test dengan yang diinputkan sama
* Memanggil variabel NAMA menggunakan perintah test denga nisi variabel $NAMA = boris
* Status exit bernilai 1, karena isi variabel test dengan yang diinputkan tidak sama

1. Aplikasi test dengan konstruksi if

$ vi prog06.sh

#!/bin/sh

# prog06.sh echo –n “NAMA = “

read NAMA

if test “$NAMA” = amir

then

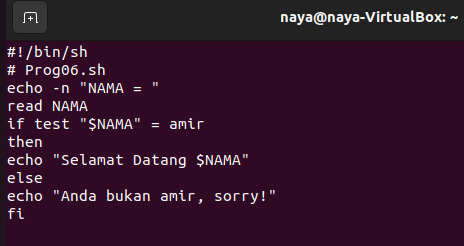
echo “Selamat Datang $NAMA”

else

echo “Anda bukan amir, sorry!”

fi



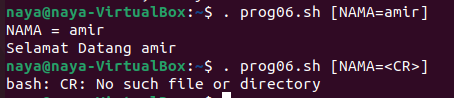


* Membuat file prog06.sh menggunakan perintah vi
* Menuliskan perintah menggunakan editor vi
* Setelah menginputkan nama, akan dilakukan pengondisian. Ketika inputan user amir akan menampilkan “Selamat datang amir”, jika tidak akan menampilkan “Anda bukan amir, sorry!”
* Script fi untuk mengakhiri program

1. Jalankan program prog06.sh dengan memasukkan NAMA = amir dan NAMA = perhatikan hasil tampilannya

$ . prog06.sh [NAMA = amir]

$ . prog06.sh [NAMA = <CR> ] (Terdapat pesan error)



* Merupakan perintah untuk menampilkan isi dari file prog06.sh
* Saat user menginputkan nama amir pada variabel NAMA maka akan menampilkan “Selamat datang amir”
* Saat user menuliskan <CR> pada variabel NAMA maka akan terjadi pesan eror

1. Modifikasi prog06.sh dengan menggunakan notasi untuk test

$ vi prog06.sh

#!/bin/sh

# prog06.sh

echo –n “NAMA = “

read NAMA

if [ “$NAMA” = amir ]

then

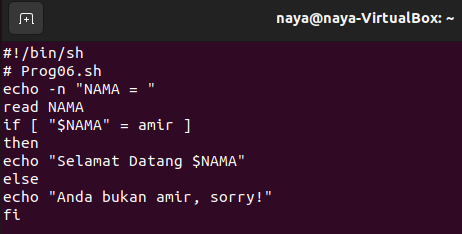
echo “Selamat Datang $NAMA”

else

echo “Anda bukan amir, sorry!”

fi

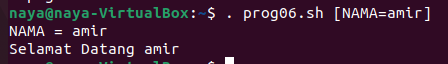




* Membuat file prog06.sh menggunakan perintah vi
* Menuliskan perintah menggunakan editor vi
* Setelah menginputkan nama, akan dilakukan pengondisian. Ketika inputan user amir akan menampilkan “Selamat datang amir”, jika tidak akan menampilkan “Anda bukan amir, sorry!”
* Script fi untuk mengakhiri program

1. Jalankan program prog06.sh dengan memasukkan NAMA = amir

$ . prog06.sh [NAMA = amir]

****

* Perintah untuk menampilkan file prog06.sh, jika user menginputkan nama amir pada variabel NAMA maka akan menampilkan “Selamat Datang amir”

**Percobaan 2 : Notasi && dan ||**

1. Bila file prog01.sh ada (TRUE), maka jalankan program berikutnya. File prog01.sh ada, karena itu exit status adalah TRUE, hasil operasi AND masih tergantung pada hasil eksekusi instruksi ke 2, dan dengan demikian instruksi echo akan dijalankan.

$ [ -f prog01.sh ] && echo “prog01.sh ada”



Analisa:

* && -> Jika file prog01.sh bernilai TRUE maka akan menjalankan program berikutnya yaitu perintah echo
* Karene file prog01.sh telah dibuat maka keadaannya TRUE dan menampilkan pesan “prog01.sh ada”

1. File prog99.sh tidak ada, karena itu exit status adalah FALSE dan instruksi echo tidak dijalankan

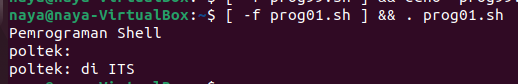
$ [ -f prog99.sh ] && echo “Prog99.sh ada”



* && -> Jika file prog99.sh bernilai TRUE maka akan menjalankan program berikutnya yaitu perintah echo
* Karena file prog99.sh belum pernah dibuat maka program akan keluar tanpa menampilkan perintah apa-apa

1. Bila prog01.sh ada maka jalankan shell script tersebut

$ [ -f prog01.sh ] && . prog01.sh



* && -> Jika file prog01.sh bernilai TRUE maka akan menjalankan program berikutnya yaitu perintah untuk menampilkan isi dari file prog01.sh
* Karena file prog01.sh telah dibuat maka bernilai TRUE dan menampilkan isi dari file prog01.sh

1. Bila prog01.sh ada maka jalankan program berikutnya. File prog01.sh memang ada, karena itu exit status adalah TRUE, dan karena sudah TRUE maka instruksi echo tidak lagi dijalankan

$ [ -f prog01.sh ] || echo “Dieksekusi tidak ?”



* || -> jika perintah sebelumnya bernilai TRUE, maka perintah selanjutnya tidak perlu dijalankan
* Karena file prog01.sh telah dibuat maka bernilai TRUE sehingga program selanjutnya tidak perlu dieksekusi dan program langusung keluar tanpa menampilkan apa-apa

1. File prog99.sh tidak ada, karena itu exit status adalah FALSE, hasil masih tergantung atas exit status instruksi ke dua, karena itu instruksi echo dijalankan

$ [ -f prog99.sh ] || echo “Dieksekusi tidak ?”



* || -> jika perintah sebelumnya bernilai TRUE, maka perintah selanjutnya tidak perlu dijalankan
* Karena file prog99.sh tidak pernah dibuat maka program akan mengeksekusi perintah selanjutnya, yaitu menampilkan pesan echo “Dieksekusi tidak ?”

1. File prog99.sh tidak ada, maka tampilkan pesan error

$ [ -f prog99.sh ] || echo “Sorry, prog99.sh tidak ada”



* || -> jika perintah sebelumnya bernilai TRUE, maka perintah selanjutnya tidak perlu dijalankan
* Karena file prog99.sh tidak pernah dibuat maka program akan mengeksekusi perintah selanjutnya, yaitu menampilkan pesan echo “Sorry, prog99.sh tidak ada”

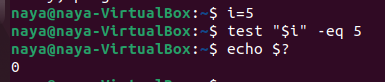
**Percobaan 3 : Operator bilangan bulat untuk test**

1. Menggunakan operator dengan notasi test

$ i=5

$ test “$i” –eq 5

$ echo $?

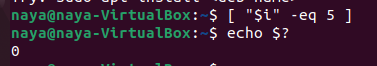


* Variabel I bernilai 5
* Perintah -eq dalam test untuk membandingkan nilai variabel i apakah sama dengan 5
* Jika status exit bernilai 0 maka sama/benar

1. Menggunakan operator dengan notasi [ ] (penganti notasi test)

$ [ “$i” –eq 5 ]

$ echo $?



* Perintah -eq dalam test untuk membandingkan nilai variabel i apakah sama dengan 5
* Jika status exit bernilai 0 maka sama/benar

**Percobaan 4 : Operator Logical dan konstruksi elif**

1. Buatlah file prog07.sh

$ vi prog07.sh

#!/bin/sh

# prog07.sh

echo –n “INCOME = “

read INCOME

if [ $INCOME –ge 0 –a $INCOME –le 10000 ]

then

BIAYA=10

elif [ $INCOME –gt 10000 –a $INCOME –le 25000 ]

then

BIAYA=25

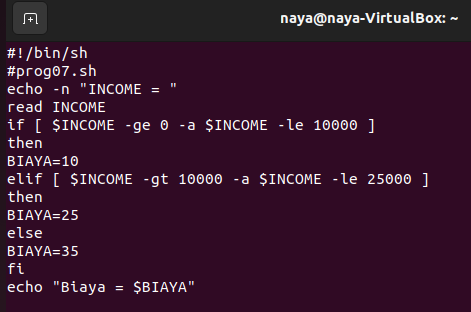
else

BIAYA=35

fi

echo “Biaya = $BIAYA”





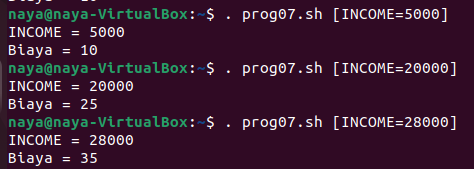
* Membuat file prog07.sh dan mengedit di editor vi
* Setelah user melakukan input nilai pada variabel INCOME maka akan dilakukan pengondisian
* $INCOME –ge 0 –a $INCOME –le 10000 artinya nilai variabel INCOME lebih dari sama dengan 0 dan kurang dari sama dengan 10000 -> menampilkan BIAYA=10
* $INCOME –gt 10000 –a $INCOME –le 25000 artinya nilai variabel INCOME lebih dari 10000 dan kurang dari sama dengan 25000 -> menampilkan BIAYA=35
* Kondisi else merupakan kondisi jika nilai INCOME tidak memenuhi akan menampilkan BIAYA=35

1. Jalankan file prog07.sh dan masukkan untuk INCOME=5000, 20000, 28000

$ . prog07.sh [INCOME=5000]

$ . prog07.sh [INCOME=20000]

$ . prog07.sh [INCOME=28000]



* Menampilkan file porg07.sh dan menginputkan nilai dari variabel INCOME
* Jika nilai INCOME=5000 maka Biaya=10
* Jika nilai INCOME=20000 maka Biaya=25
* Jika nilai INCOME=28000 maka Biaya=35

**Percobaan 5 : Hitungan aritmetika**

1. Menggunakan utilitas expr

$ expr 5 + 1

$ A=5

$ expr $A + 2

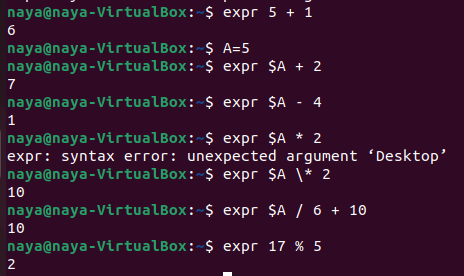
$ expr $A – 4

$ expr $A \* 2 (Ada Pesan Error)

$ expr $A \\* 2

$ expr $A / 6 +10

$ expr 17 % 5



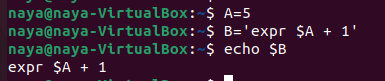
* Perintah expr digunakan untuk mengeksekusi oprasi hitung
* Jika oprasi hitung +, - \\*, /, % dapat dieksekusi, namun jika \* tidak dapat dieksekusi karena expr tidak menerima argument yang valid

1. Substitusi isi variable dengan hasil utilitas expr

$ A=5

$ B=`expr $A + 1`

$ echo $B



* Variabel A bernilai 5
* Varibel B berisi eksekusi operator hitung expr variabel $A + 1
* Echo $B untuk menampilkan isi variabel B

**Percobaan 6 : Instruksi exit**

1. Buat shell script prog08.sh

$ vi prog08.sh

#!/bin/sh

if [ -f prog01.sh ]

then

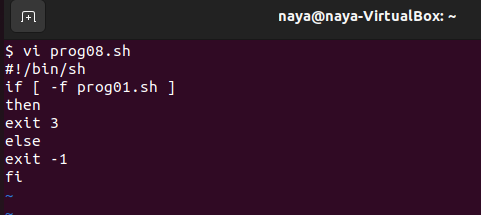
exit 3

else

exit –1

fi





* Perintah untuk meelakukan pengecekkan file
* jika terdapat file prog01.sh maka akan menghasilkan exit 3
* jika tidak terdapat file prog01.sh maka akan menampilkan exit -1
* fi menandakan program berhenti

1. Jalankan script prog08.sh dan periksa status exit

$ . prog08.sh

$ echo $?





* . prog08.sh merupakan perintah untuk menampilkan file prog08.sh
* Echo $? Merupakan perintah untuk menampilkan status keluaran dari perintah terakhir yang dijalankan

**Percobaan 7 : Konstruksi case – esac**

1. Buatlah file prog09.sh dengan editor vi

$ vi prog09.sh

#!/bin/sh

# Prog: prog09.sh

echo “1. Siapa yang aktif”

echo “2. Tanggal hari ini”

echo “3. Kalender bulan ini”

echo –n “ Pilihan : “

read PILIH

case $PILIH in

1)

echo “Yang aktif saat ini”

who

;;

2) echo “Tanggal hari ini”

date

;;

3)

echo “Kalender bulan ini”

cal

;;

\*)

echo “Salah pilih !!”

;;

Esac

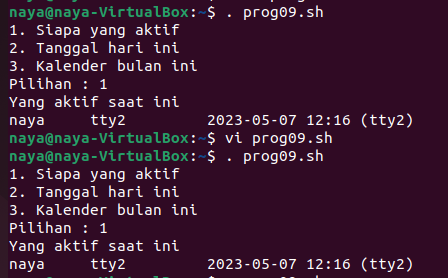




* Script tersebut menggunakan switch case untuk memilih perintah yang akan dijalankan
* Perintah echo untuk menampilkan teks
* Perintah read PILIH untuk menampung isi dari inputan pilihan user
* Jika variabel PILIH bernilai 1 maka perintah yang dijalankan adalah who (menampilkan yang aktif saat ini)
* Jika variabel PILIH bernilai 2 maka perintah yang dijalankan adalah date (menampilkan tanggal hari ini)
* Jika variabel PILIH bernilai 3 maka perintah yang dijalankan adalah cal (menapilkan kalender bulan ini)
* Jika inputan user tidak tidak terdapat dalam pilihan maka akan menampilkan pesan salah pilih

1. Jalankan program prog09.sh, cobalah beberapa kali dengan inputan yang berbeda

$ . prog09.sh



* Perintah . prog09.sh merupakan perintah untuk menampilkan isi dari file prog09.sh
* Dalam file itu berisi script case pemilihan
* Setelah perintah dieksekusi maka user diminta untuk menginputkan pilihan dari angka 1-3
* Inputan 1 menampilkan user yang sedang aktif
* Inputan 2 menampilkan tanggal hari ini
* Inputan 3 menampilkan kalender bulan ini

1. Buatlah file prog10.sh yang merupakan bentuk lain dari case

$ vi prog10.sh

#!/bin/sh

# Prog: prog10.sh

echo –n “Jawab (Y/T) : “

read JWB

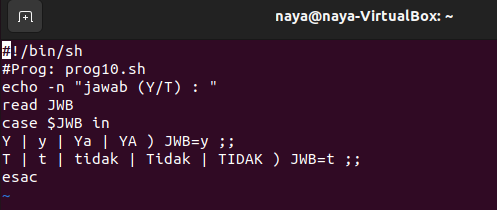
case $JWB in

y | Y | ya |Ya |YA ) JWB=y ;;

t | T | tidak | Tidak | TIDAK ) JWB=t ;;

esac

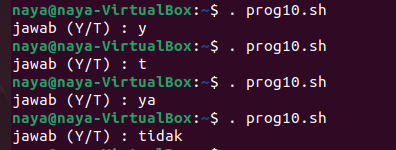




* Script tersebut merupakan script lain dari penggunaan case pilihan
* Script tersebut tidak jauh beda dengan nomor 2
* Users perlu menginputkan pilihan untuk menampilkan perintah

1. Jalankan program prog10.sh, cobalah beberapa kali dengan inputan yang berbeda

$ . prog10.sh



* Perintah tersebut merupakan perintah untuk menampilkan isi dari file prog10.sh yaitu file yang berisi script case pilihan

1. Modifikasi file prog10.sh yang merupakan bentuk lain dari case

$ vi prog10.sh

#!/bin/sh

# Prog: prog10.sh

echo –n “Jawab (Y/T) : \c“

read JWB

case $JWB in

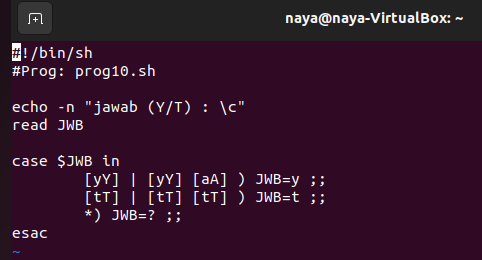
[yY] | [yY][aA] ) JWB=y ;;

[tT] | [tT]idak ) JWB=t ;;

\*) JWB=? ;;

Esac

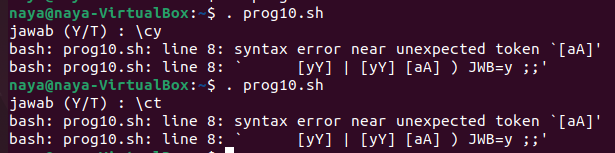




* Script tersebut merupakan script lain dari penggunaan case pilihan
* Script tersebut tidak jauh beda dengan nomor 3
* Users perlu menginputkan pilihan untuk menampilkan perintah

1. Jalankan program prog10.sh, cobalah beberapa kali dengan inputan yang berbeda

$ . prog10.sh



* Perintah tersebut merupakan perintah untuk menampilkan isi dari file prog10.sh yaitu file yang berisi script case pilihan

**Percobaan 8 : Konstruksi for-do-done**

1. Buatlah file prog11.sh

$ vi prog11.sh

#!/bin/sh

# Prog: prog11.sh

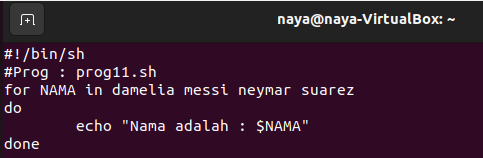
for NAMA in bambang harry kadir amir

do

echo “Nama adalah : $NAMA”

done

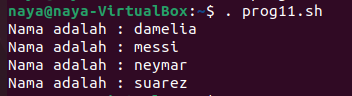




* Perintah tersebut adalah untuk membuat file prog11.sh
* Script tersebut adalah untuk melakukan perulangan for yang akan menjalankan satu kali untuk setiap nama dalam list Bambang, harry, kadir, dan amir
* Variabel NAMA akan berisi setiap nama dalam iterasi

1. Jalankan program prog11.sh

$ . prog11.sh



* Perintah tersebut adalah untuk menampilkan isi dari file prog11.sh

1. Buatlah file prog12.sh yang berisi konstruksi for dan wildcard, program ini akan menampilkan nama file yang berada di current direktori

$ vi prog12.sh

#!/bin/sh

# Prog: prog12.sh

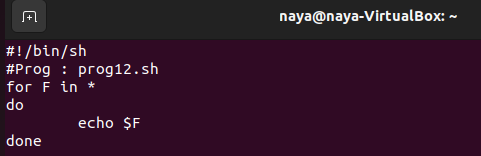
for F in \*

do

echo $F

done

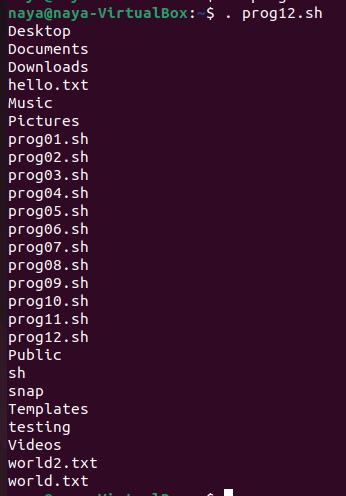




* Perintah tersebut adalah untuk memebuat file prog12.sh dan isinya dibuat menggunakan editor vi
* Script tersebut adalah untuk melakukan perulangan for yang akan menjalankan satu kali untuk setiap file atau direktori saat ini
* Variabel F berisi nama file atau direktori dalam iterasi
* Baris keempat menjalankan perintah "echo" pada setiap nama file atau direktori yang ditemukan, dengan menggunakan variabel "F" sebagai argumen. Perintah "echo" akan menampilkan nama file atau direktori

1. Jalankan program prog12.sh

$ . prog12.sh



* Perintah tersebut adalah untuk menampilkan isi dari file prog12.sh
* Isi dari file prog12.sh merupakan file direktori saat ini

1. Modifikasi file prog12.sh, program ini akan menampilkan long list dari file yang mempunyai ekstensi lst

$ vi prog12.sh

#!/bin/sh

# Prog: prog12.sh

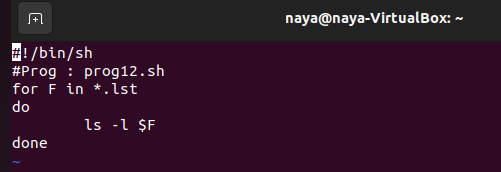
for F in \*.lst

do

ls –l $F

done





* Perintah tersebut adalah untuk mengedit file prog12.sh menggunakan editor vi
* Script tersebut adalah untuk melakukan pengulangan for yang akan berjalan satu kali untuk setiap file direktori saat ini deng ekstensi “.lst”
* Variabel F berisi nama file dalam setiap iterasi
* Baris keempat menjalankan perintah "ls -l" pada setiap file yang ditemukan, dengan menggunakan variabel "F" sebagai argumen. Perintah "ls -l" akan menampilkan izin file dan metadata lainnya pada setiap file

1. Jalankan program prog12.sh

$ . prog12.sh



* Perintah tersebut adalah untuk menampilkan isi dari file prog12.sh

**Percobaan 9 : Konstruksi while-do-done**

1. Buatlah file prog13.sh

$ vi prog13.sh

#!/bin/sh

# Prog: prog13.sh

PILIH=1

while [ $PILIH –ne 4 ]

do

echo “1. Siapa yang aktif”

echo “2. Tanggal hari ini”

echo “3. Kalender bulan ini”

echo “4. Keluar”

echo “ Pilihan : \c”

read PILIH

if [ $PILIH –eq 4 ]

then

break

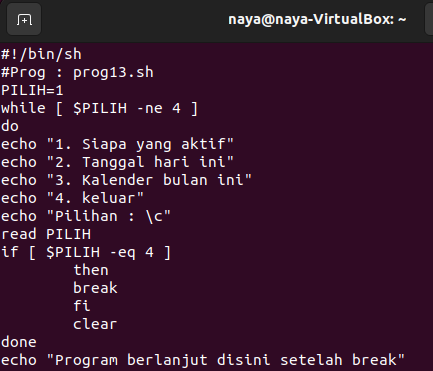
fi

clear

done

echo “Program berlanjut di sini setelah break”

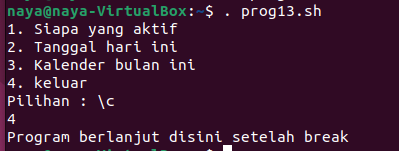




* Perintah ini adalah untuk membuat file prog13.sh
* Script tersebut adalah untuk melakukan pengulangan while yang akan berjalan selama variabel PILIH tidak sama dengan 4
* Variabel PILIH akan diupdate setelah pengguna memilih salah satu opsi pada menu
* Menampilkan menu 4 opsi dan meminta pengguna untuk meminta opsi dengan perintah read yang akan menyimpan pilihan pengguna dalam variabel PILIH
* Batas terakhir perulangan while menggunakan perintah break

1. Jalankan program prog13.sh

$ . prog13.sh



* Perintah tersebut adalah untuk menampilkan isi dari file prog13.sh
* User diminta untuk menginputkan pilihan
* Output

**Percobaan 10 : Instruksi dummy**

1. Modifikasi file prog13.sh

$ vi prog13.sh

#!/bin/sh

# Prog: prog13.sh

PILIH=1

while :

do

echo “1. Siapa yang aktif”

echo “2. Tanggal hari ini”

echo “3. Kalender bulan ini”

echo “4. Keluar”

echo “ Pilihan : \c”

read PILIH if [ $PILIH –eq 4 ]

then

break

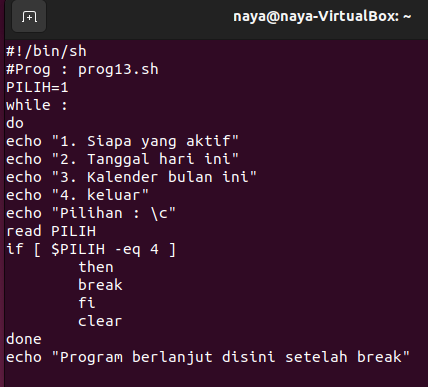
fi

clear

done

echo “Program berlanjut di sini setelah break”

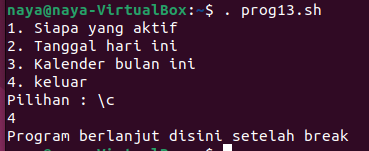




* Perintah tersebut adalah untuk mengedit file prog13.sh menggunakan editor vi
* Variabel PILIH diinisialisasi dengan nilai 1
* Menggunakan perulangan while
* Menampilkan menu 4 opsi dan meminta pengguna untuk memilih opsi menggunakan perintah read yang akan menyimpan pilihan pengguna dalam variabel PILIH
* Setelah memilih 4 opsi, script akan keluar dari loop while dan menampilkan pesan “Programm berlanjut di sini setelah break”

1. Jalankan program prog13.sh

$ . prog13.sh



* Perintah tersebut adalah untuk menampilkan isi dari file prog13.sh setelah diedit/terbaru

1. Buatlah file prog14.sh yang berisi instruksi dummy untuk konstruksi if

$ vi prog14.sh

#!/bin/sh

# Prog: prog14.sh

echo –n “Masukkan nilai : “

read A

if [ $A –gt 100 ]

then

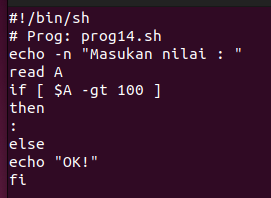
:

else

echo “OK !”

fi





* Perintah tersebut adalah untuk membuat file prog14.sh
* Pengguna diminta untuk memasukkan sebuah nilai yang ditampung ke dalam variabel A
* Melakukan pengecekan, jika nilai lebih besar dari 100 maka script tidak melakukan apa-apa dan hanya menggunakan operator “:” sebagai penandaan
* Jika tidak lebih besar dari 100 maka script akan menampilkan pesan “ok!”

1. Jalankan program prog14.sh beberapa kali dengan input yang berbeda

$ . prog14.sh



* Perintah tersebut adalah untuk menampilkan isi dari file prog14.sh

**Percobaan 11 : Fungsi**

1. Buatlah file fungsi.sh

$ vi fungsi.sh

#!/bin/sh

# Prog: fungsi.sh

F1( ) {

echo “Fungsi F1”

return 1

}

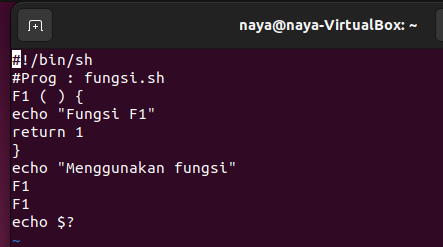
echo “Menggunakan Fungsi”

F1

F1

echo $?

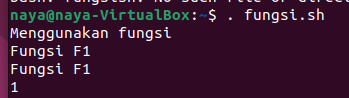




* Perintah tersebut untuk membuat file fungsi.sh
* Variabel “$?” digunakan untuk menampilkan nilai yang dikembalikan oleh fungsi terakhir yang dijalankan
* Fungsi “F1” mengembalikan nilai 1, maka nila 1 akan ditampilkan pada baris terakhir output dari skrip

1. Jalankan program fungsi.sh

$ . fungsi.sh



* Perintah tersebut adalah untuk menampilkan isi dari file fungsi.sh

1. Menggunakan variable pada fungsi dengan memodifikasi file fungsi.sh

$ vi fungsi.sh

#!/bin/sh

# Prog: fungsi.sh

F1( )

{

Honor=10000

echo “Fungsi F1”

return 1

}

echo “Menggunakan Fungsi”

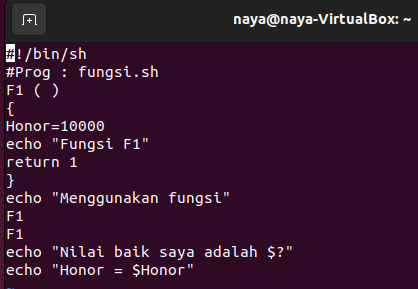
F1

F1

echo “Nilai balik adalah $?”

echo “Honor = $Honor”

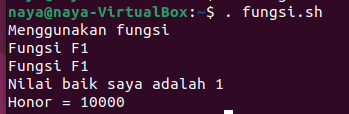




* Perintah ini untuk mengedit isi dari file fungsi.sh
* Variabel Honor di definisikan variabel local dalam fungsi F1, yang berarti nilai variabel hanya tersedia di dalam vungsi dan tidak dapat diakses di luar vungsi
* Jika tidak tersedia, saat mencoba menampilkan nilai variabel “Honor” di luar fungsi, akan muncul pesan kesalahan bahwa variabel “Honor” tidak didefinisikan

1. Jalankan program fungsi.sh

$ . fungsi.sh



* Perintah tersebut adalah untuk menampilkan isi dari file fungsi.sh terbaru/ setelah diupdate

1. Menggunakan variable pada fungsi dengan memodifikasi file fungsi.sh

$ vi fungsi.sh

#!/bin/sh

# Prog: fungsi.sh

F1( )

{

local Honor=10000

echo “Fungsi F1”

return 1

}

echo “Menggunakan Fungsi”

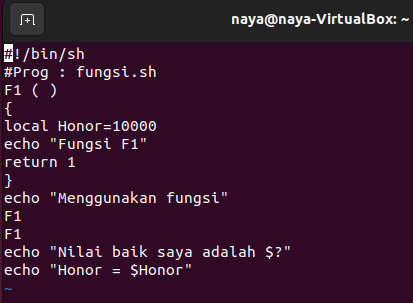
F1

F1

echo “Nilai balik adalah $?”

echo “Honor = $Honor”

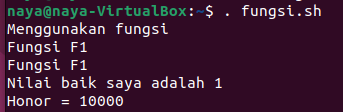




* Kode program ini menggunakan bahasa shell script untuk mendefinisikan fungsi F1 yang mengatur variabel lokal "Honor" menjadi 10000 dan mencetak pesan "Fungsi F1"
* Kemudian program mencoba memanggil fungsi F1 dua kali dan mencetak nilai kembali dari fungsi tersebut dan nilai variabel "Honor"
* Namun, terdapat masalah pada baris terakhir dari program, yaitu saat mencoba mencetak nilai variabel "Honor"
* Karena variabel "Honor" didefinisikan sebagai variabel lokal di dalam fungsi F1, maka variabel tersebut tidak dapat diakses di luar fungsi
* Oleh karena itu, program akan memberikan kesalahan saat mencoba mencetak nilai variabel "Honor"

1. Jalankan program fungsi.sh

$ . fungsi.sh



* Perintah tersebut adalah untuk menampilkan isi dari file fungsi.sh terbaru/ setelah diupdate

**LATIHAN**

1. Buatlah program salin.sh yang menyalin file (copy ) sebagai berikut :

salin.sh file-asal file-tujuan

Dengan ketentuan :

• Bila file asal tidak ada, berikan pesan, salin gagal.

• Bila file tujuan ada dan file tersebut adalah directory, beri pesan bahwa file tidak bisa disalin ke direktori

• Bila file tujuan ada dan file biasa, beri pesan apakan file tersebut akan dihapus, bila dijawab dengan “Y”, maka copy file tersebut

• Bila file tujuan belum ada, lakukan copy

Untuk mengambil nama file, gunakan parameter $1 dan $2. Bila jumlah parameter tidak sama ($#) dengan 2, maka beri pesan exit = -1

#!/bin/sh

# file: salin.sh

# Usage: salin.sh fasal ftujuan

if [ $# -ne 2]

then echo “Error, usage: salin.sh file-asal file-tujuan”

exit –1

fi

fasal=$1

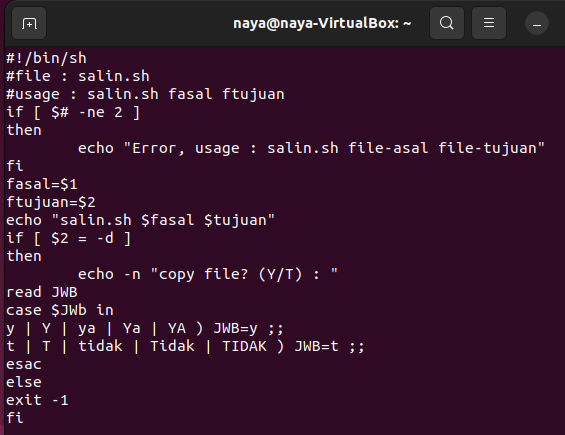
ftujuan=$2

echo “salin.sh $fasal $ftujuan”

……

……





* Perintah diatas adalah untuk copy file, dimana jika file berhasil ditemukan, maka program langsung di close
* Untuk pilihan digunakan perintah if else then, tetapi karena perintah pertama pada shell script salin.sh menyatakan jika file tidak ada maka akan close, dan memang isi file tidak ada, sehingga perintah yang dikerjakannya pada saat shell script dieksekusi menggunakan perintah
* Salin.sh adalah perintah pilihan nomor 1 yaitu “keluar” karena tidak ada filenya dan tampil seperti gambar di atas.

1. Buat program yang memeriksa nama direktori, jika parameter tersebut adalah direktori, maka jalankan instruksi ls –ld pada direktori tersebut. Namakan program tersebut checkdir.sh. Gunakan notasi [ -d NamaDirektori ] dan pilih logical && atau || pada level shell.

#!/bin/sh

# file: checkdir.sh

# Usage: checkdir.sh DirectoryName

#

if [ $# -ne 1]

then

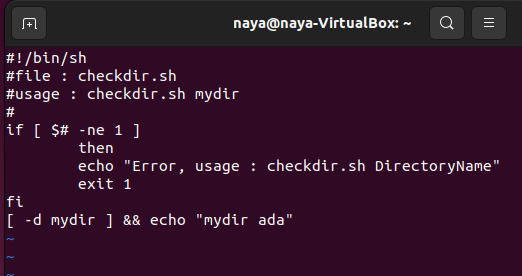
echo “Error, usage: checkdir.sh DirectoryName”

exit 1

fi

[ … ] && …





* Perintah di atas adalah untuk memeriksa nama direktori
* Jika dijalankan program checkdir.sh ini tidak menemukan direktori yang dicari, program akan otomatis di “closed” atau exit dan tampil seperti gambar di atas

1. Dengan shell script pph.sh, hitung PPH per tahun dengan ketentuan sebagai berikut:

- 10 juta pertama PPH 15%

- 25 juta berikutnya (sisa) PPH 25%

- Bila masih ada sisa, maka sisa tersebut PPH 35%

Contoh :

Gaji 8 juta

PPH = 15% \* 8 juta

Gaji 12 juta

PPH =15% \* 10 juta + 25% \* (12-10) juta

Gaji 60 juta

PPH = 15% \* 10 juta + 25% \* 25 juta + 25% \* (60-10-25) juta

Debugging : untuk melakukan tracing (debug) gunakan opsi –x pada eksekusi shell.

$ sh –x pph.sh

+ echo –n ‘Berikan gaji dalam ribuan rupiah : ‘ Berikan gaji dalam ribuan rupiah : + read gaji 20000

+ pkp=10000

+ ‘[‘ 20000 –le 10000 ‘]’

++ expr 20000 – 10000

+ gaji=10000 + pph=1500

+ pkp=25000

+ ‘[‘ 10000 –le 25000 ‘]’

+ pkp=10000

++ expr 1500

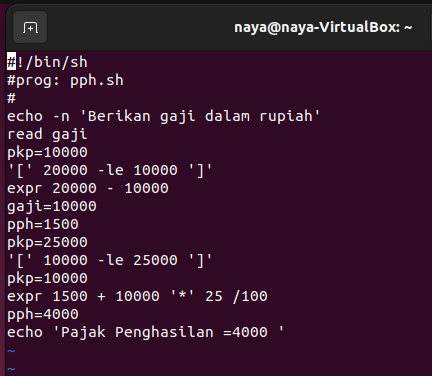
+ 10000 ‘\*’ 25 / 100

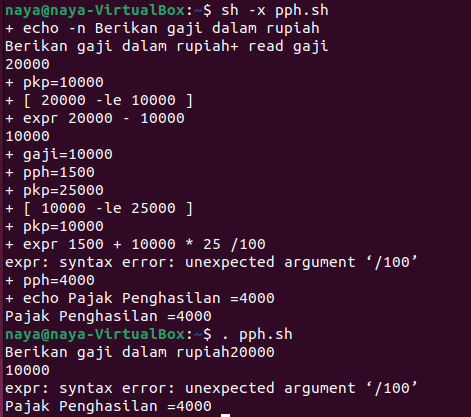
+ pph=4000

+ echo ‘Pajak Penghasilan = 4000’

Pajak Penghasilan = 4000







* Perintah di atas adalah untuk menghitung PPH per tahun dengan ketentuan 10 juta pertama PPH 15%, 25 juta berikutnya (sisa) PPH 25%,, bila masih ada sisa, maka sisa tersebut PPH 35%
* Perintah tersebut dituliskan dalam editor vi file pph.sh

1. Buatlah program myprog.sh yang memproses parameter $1, nilai parameter harus berupa string :

start

stop

status

restart

reload

Bila buka dari string tersebut, maka berikan pesan error. Sempurnakan program di bawah ini untuk keperluan tersebut

#!/bin/sh

# See how we were called

case “$1” in

start)

echo “Ini adalah start”

;;

stop)

echo “Ini adalah stop”

;;

\*)

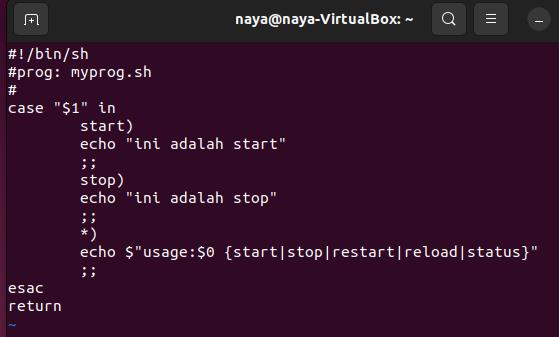
echo $”Usage:$0 {start|stop|restart|reload|status}”

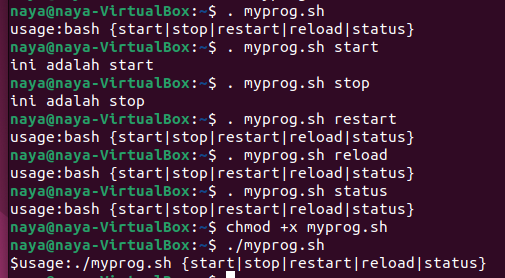
;;

esac

return







* Perintah tersebut adalah untuk membuat file myprog.sh yang memproses parameter $1, nilai parameter harus berupa string, stop, status, restart, dan reload
* Jika inputan yang dimasukkan start, stop, status, restart, atau reload maka akan muncul output seperti gambar di atas dan jika tidak, akan muncul informasi kesalahan dan tampil seperti gambar di atas

1. Buat sebuah fungsi pada script confirm.sh yang memberikan konfirmasi jawaban Yes, No atau Continue. Jika jawaban Yes, maka beri nilai balik 0, No = 1 dan Continue = 2. Modifikasi kerangka program berikut untuk memenuhi permintaan tersebut.

#!/bin/sh

# Confirm whether we really want to run this service

confirm() {

local YES=”Y”

local NO=”N”

local CONT=”C”

while :

do

echo –n “(Y)es/(N)o/(C)ontinue? {Y] “

read answer

answer=`echo “$answer” | tr ‘[a-z]’ ‘[A-Z]’`

if [ “$answer” = “” –0 “$answer” = $YES ]

then

return 0

elif ….

then

return 2

elif ….

then

return 1

fi

done

}

Test fungsi diatas dengan program berikut :

$ vi testp.sh

. confirm.sh

confirm

if [ $? –eq 0 ]

then

echo “Jawaban YES OK”

elif [ $? =eq 1 ]

then

echo “Jawaban NO”

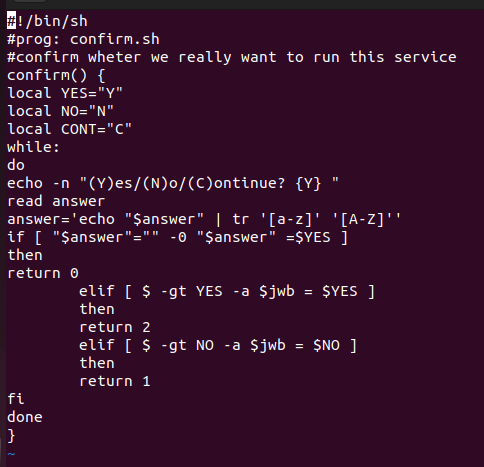
else

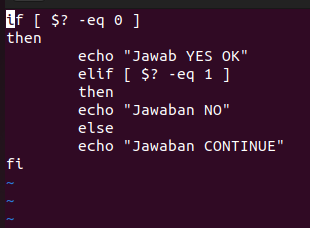
echo “Jawaban CONTINUE”

fi

Perhatikan baris pertama, adalah loading dari fungsi confirm yang terdapat di script confirm.sh. Setelah eksekusi script tersebut, maka fungsi confirm dapat digunakan.









* Perintah di atas yaitu membuat fungsi pada script confirm.sh yang memberikan konfirmasi jawaban Yes, No atau Continue
* Jika jawaban Yes, maka beri nilai balis 0, No = 1 dan continue = 2 serta memodifikasi program tersebut
* Pada perintah yang terdapat di modul tidak bisa dijalankan, sehingga digunakan modifikasi terhadap isi perintah
* Lalu digunakan perintah elif seperti yang terlihat pada gambar di atas, dan Ketika dijalankan hasilnya akan ditampilkan seperti di atas dengan output tulisan “Jawaban YES OK”