CSCI 330
THE UNIX SYSTEM

Shell Programming

BASH CONTROL STRUCTURES

- if-then-else
- case
- loops
  - for
  - while
  - until
  - select

IF STATEMENT

```
if command
then
  statements
fi
```

- statements are executed only if **`command`** succeeds, i.e. has return status “0”

THE SIMPLE IF STATEMENT

```
if [ condition ]
 then
  statements
fi
```

- executes the statements only if **`condition`** is true

```
#! /bin/bash

# if [condition]
# then
#     statement
# fi # end of if

a=10
if [ $a -ne 9 ]
  then
   echo " condition is true"
fi
```

Tab Width: 8 Ln 1, Col 1 INS

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
 condition is true
adhoc@adhoc:~/Desktop$
```

RELATIONAL OPERATORS

Meaning	Numeric	String
Greater than	-gt	
Greater than or equal	-ge	
Less than	-lt	
Less than or equal	-le	
Equal	-eg	= or ==
Not equal	-ne	!=
str1 is less than str2		str1 < str2
str1 is greater str2		str1 > str2
String length is greater than zero		-n str
String length is zero		-z str

RELATIONAL OPERATORS

```
integer comparison

-eq - is equal to - if [ "$a" -eq "$b" ]
-ne - is not equal to - if [ "$a" -ne "$b" ]
-gt - is greater than - if [ "$a" -gt "$b" ]
-ge - is greater than or equal to - if [ "$a" -ge "$b" ]
-lt - is less than - if [ "$a" -lt "$b" ]
-le - is less than or equal to - if [ "$a" -le "$b" ]
< - is less than  - (("$a" < "$b"))
<= - is less than or equal to - (("$a" <= "$b"))
> - is greater than - (("$a" > "$b"))
>= - is greater than or equal to  - (("$a" >= "$b"))

string comparison
= - is equal to - if [ "$a" = "$b" ]
== - is equal to - if [ "$a" == "$b" ]
!= - is not equal to - if [ "$a" != "$b" ]
< - is less than, in ASCII alphabetical order - if [[ "$a" < "$b" ]
> - is greater than, in ASCII alphabetical order - if [[ "$a" > "$b
-z - string is null, that is, has zero length
```

THE IF-THEN with Relationa operator If ((cond))...

shell3.sh ~/Desktop

```
#! /bin/bash

# if ((<,<=,>,>=))
# then
#    statement
# fi # end of if

a=10
if (($a >= 10))
  then
   echo " condition is true"
fi
```

sh Tab Width: 8 Ln 3, Col 19 INS

adhoc@adhoc: ~/Desktop

File Edit View Search Terminal Help

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
 condition is true
adhoc@adhoc:~/Desktop$
```

TEST COMMAND

Test work in 3 ways:

1. Compares two number

2.Compare two string or single one for null

3. Checks a file attributes

Syntax:

```
test expression
[ expression ]
```

- evaluates ‘expression’ and returns true (0) or false(1)

Open ▾ shel... ~/Des... Save

```
1 #! /bin/bash
2
3 #test numeric comparison
4 x=5;y=7;z=7
5 test $x -eq $y ; echo $?
6 echo
7 test $x -lt $y ; echo $?
8 echo
9 test $z -gt $y ; echo $?
10 echo
11 test $z -eq $y ; echo $?
```

Tab Width: 8 ▾ Ln 4, Col 12 ▾ INS

adhoc@adhoc: ~/Desktop

File Edit View Search Terminal Help

```
adhoc@adhoc:~/Desktop$ ./shell32_test.sh
1

0

1

0
adhoc@adhoc:~/Desktop$
```

THE IF-THEN " String" STATEMENT

shell3.sh ~/Desktop

```
#! /bin/bash

# if with string
echo "enter name "
read str
if [ $str = "unix" ]
  then
   echo " condition is true"
fi
```

sh Tab Width: 8 Ln 6, Col 20 INS

adhoc@adhoc: ~/Desktop

File Edit View Search Terminal Help

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
enter name
unix
 condition is true
adhoc@adhoc:~/Desktop$
```

THE IF-THEN-ELSE STATEMENT

```
if [ condition ]; then
    statements-1
else
  statements-2
fi
```

- executes statements-1 if condition is true
- executes statements-2 if condition is false

shell3.sh
~/Desktop

```
#! /bin/bash

# Even or odd
echo "enter number to check even odd "
read num
rem=$(( $num%2 ))
if [ $rem -eq 0 ]
  then
   echo " Even"
else
   echo " odd"
fi
```

sh Tab Width: 8 Ln 6, Col 16 INS

adhoc@adhoc: ~/Desktop

File Edit View Search Terminal Help

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
enter number to check even odd
24
 Even
adhoc@adhoc:~/Desktop$ ./shell3.sh
enter number to check even odd
25
 odd
adhoc@adhoc:~/Desktop$
```

THE IF…STATEMENT

```
if [ condition ]; then
    statements
elif [ condition ]; then
    statement
else
    statements
fi
```

- The word **elif** stands for “else if”
- It is part of the if statement and cannot be used by itself

Open ▾ shell3.sh ~/Desktop Save

```
#! /bin/bash
echo "Enter the year (YYYY)"
read year

if [ $((year % 4)) -eq 0 ]
then
     if [ $((year % 100)) -eq 0 ]
         then
              if [ $((year % 400)) -eq 0 ]
              then
               echo "its a leap year"
              else
               echo "its not a leap year"
               fi
    else
        echo "Its a leap year"
     fi
else
    echo "its not a leap year"
fi
```

sh ▾ Tab Width: 8 ▾ Ln 20, Col 1 ▾ INS

adhoc@a

File Edit View Search Terminal Help

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
Enter the year (YYYY)
4000
its a leap year
adhoc@adhoc:~/Desktop$ ./shell3.sh
Enter the year (YYYY)
1900
its not a leap year
adhoc@adhoc:~/Desktop$ ./shell3.sh
Enter the year (YYYY)
1982
its not a leap year
adhoc@adhoc:~/Desktop$
```

COMPOUND LOGICAL EXPRESSIONS

!	not
&&	and
||	or

<u>and, or</u>
must be enclosed within

[[ ]]

EXAMPLE: USING THE ! OPERATOR

```
#!/bin/bash

read -p "Enter years of work: " Years
if [ ! "$Years" -lt 20 ]; then
   echo "You can retire now."
else
   echo "You need 20+ years to retire"
fi
```

Open ▾ shell3.sh ~/Desktop Save

```
1 #! /bin/bash
2
3 read -p "Enter years of work: " Years
4 if [ ! "$Years" -lt 20 ]; then
5    echo "You can retire now."
6 else
7    echo "You need 20+ years to retire"
8 fi
```

sh ▾ Tab Width: 8 ▾ Ln 8, Col 3 ▾ INS

adhoc@adhoc: ~/Desktop

File Edit View Search Terminal Help

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
Enter years of work: 25
You can retire now.
adhoc@adhoc:~/Desktop$ ./shell3.sh
Enter years of work: 15
You need 20+ years to retire
adhoc@adhoc:~/Desktop$
```

EXAMPLE: USING THE && OPERATOR

```
#!/bin/bash

num=150
if [ $num -gt 100 ] && [ $num -lt 200 ]
then
  echo "The number lies between 100 and
  200"
fi
```

EXAMPLE: USING THE || OPERATOR

```
#!/bin/bash

read -p "Enter calls handled:" CHandle
read -p "Enter calls closed: " CClose
if [[ "$CHandle" -gt 150 || "$CClose" -gt 50 ]]
   then
   echo "You are entitled to a bonus"
else
   echo "You get a bonus if the calls"
   echo "handled exceeds 150 or"
   echo "calls closed exceeds 50"
fi
```

Calculator

shell3.sh ~/Desktop

```
#! /bin/bash
a=20
b=5
echo following opeartion on $a and $b
echo add = $a+$b
echo sub = $a-$b
echo mult = $a*$b
```

Tab Width: 8 Ln 5, Col 9 INS

adhoc@adhoc: ~/Desktop

File Edit View Search Terminal Help

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
following opeartion on 20 and 5
add = 20+5
sub = 20-5
mult = 20*5
adhoc@adhoc:~/Desktop$
```

Calculator

shell3.sh ~/Desktop

```
#! /bin/bash
a=20
b=5
#using double bracket(())
echo following opeartion on $a and $b
echo add = $(( a+b ))
echo sub = $(( a-b ))
echo mult = $(( a*b ))
echo div= $(( a/b ))
echo rem=$(( a%b ))
```

Tab Width: 8 Ln 9, Col 15 INS

adhoc@adhoc: ~/Desktop

File Edit View Search Terminal Help

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
following opeartion on 20 and 5
add = 25
sub = 15
mult = 100
div= 4
rem=0
adhoc@adhoc:~/Desktop$
```

Calculator
Using expr and single bracket

Open ▾ shell3.sh ~/Desktop Save

```
1 #! /bin/bash
2 a=20
3 b=5
4 #using double bracket(())
5 echo following opeartion on $a and $b
6 echo add = $(expr $a + $b )
7 echo sub = $(expr $a - $b )
8 echo mul = $(expr $a * $b )
9 echo "for expr * not work use \*"
10 echo mul= $(expr $a \* $b )
11 echo div=$(expr $a / $b )
12 echo rem=$(expr $a % $b )
13
```

Tab Width: 8 ▾ Ln 12, Col 22 ▾ INS

adhoc@adhoc: ~/Desktop

File Edit View Search Terminal Help

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
following opeartion on 20 and 5
add = 25
sub = 15
expr: syntax error
mul =
for expr * not work use \*
mul= 100
div=4
rem=0
adhoc@adhoc:~/Desktop$
```

Flaoting calculator

Open ▾ *shell3.sh ~/Desktop Save

```
1 #! /bin/bash
2 a=10.5
3 b=5
4 echo " for fraction / decimal use
5 special tool bc- basic caculator"
6 echo
7 echo "add sub mul div as for $a and $b "
8 echo "$a+$b" | bc
9 echo "$a-$b" | bc
10 echo "$a*$b" | bc
11 echo "$a/$b" | bc
12 echo " division not correct"
```

sh ▾ Tab Width: 8 ▾ Ln 12, Col 29 ▾ INS

adhoc@adhoc: ~/Desktop

File Edit View Search Terminal Help

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
 for fraction / decimal use
special tool bc- basic caculator

add sub mul div as for 10.5 and 5
15.5
5.5
52.5
2
 division not correct
adhoc@adhoc:~/Desktop$
```

Flaoting calculator (Division - scale)

Open ▾ shel... ~/Des... Save

```
#! /bin/bash
a=10.5
b=5
echo " for fraction / decimal use
special tool bc- basic caculator"
echo
echo "div as for $a and $b "
echo "$a/$b" | bc
echo " division not correct"
echo "scale=4;$a/$b" | bc
```

Tab Width: 8 ▾ Ln 10, Col 26 ▾ INS

adhoc@adhoc: ~/Desktop

File Edit View Search Terminal Help

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
 for fraction / decimal use
special tool bc- basic caculator

div as for 10.5 and 5
2
 division not correct
2.1000
adhoc@adhoc:~/Desktop$
```

Flaoting calculator (math function)

```
#! /bin/bash
a=9
echo " find sqrt of $a using bc and function"
echo "square root "

echo "scale=2;sqrt($a)" | bc -l
echo
echo "l - call math library "
echo
echo "power "
echo "scale=2;$a^3" | bc -l
```

```
adhoc@adhoc:~/Desktop$ ./shell3.sh
 find sqrt of 9 using bc and function
square root
3.00

l - call math library

power
729
adhoc@adhoc:~/Desktop$
```

Questions..

1. How to assigen floationg result of expression to any variable.
2. check enterd number is even or odd
3. Greatest of 3 number
4. Palindron for 4 digit number