



CSCI 330

THE UNIX SYSTEM

Shell Programming

BASH CONTROL STRUCTURES

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

IF STATEMENT

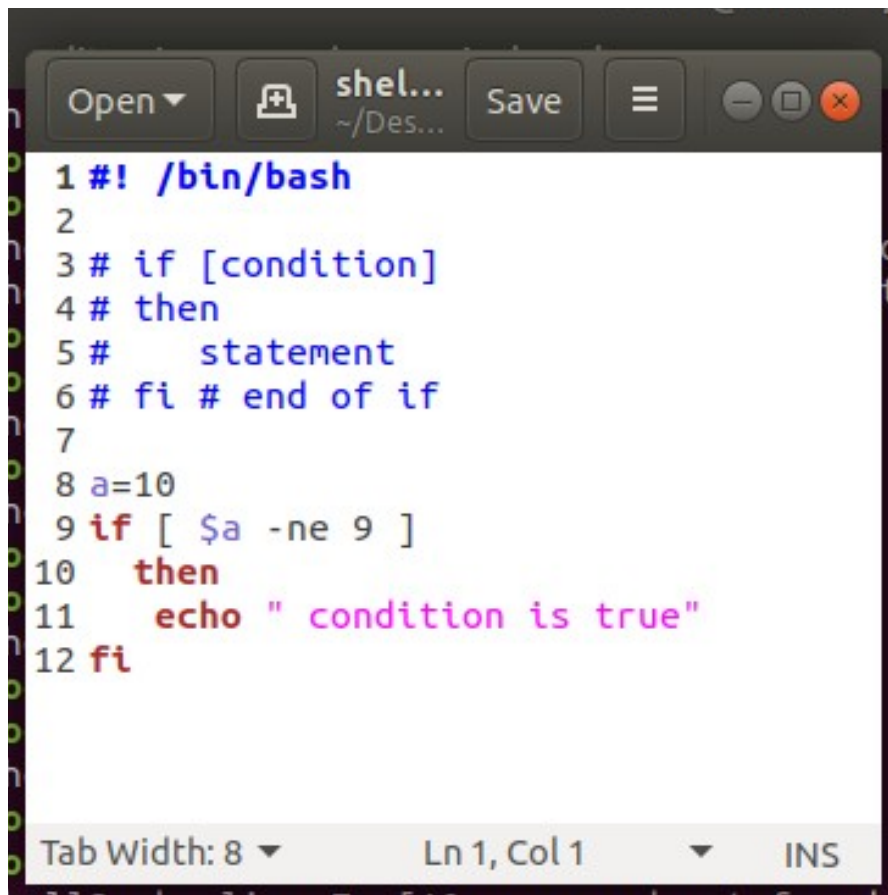
```
if command  
then  
    statements  
fi
```

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

THE SIMPLE IF STATEMENT

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

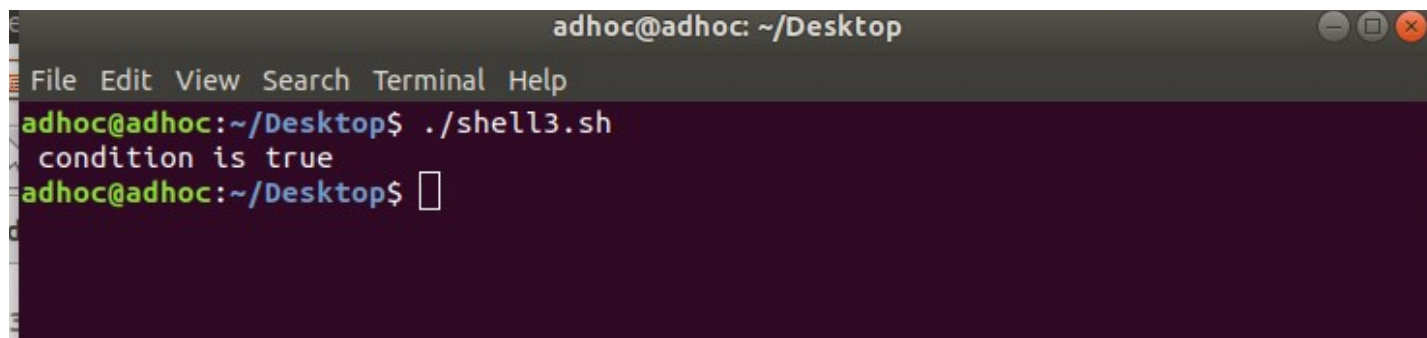
- executes the statements only if **condition** is true



A screenshot of a text editor window with a dark theme. The title bar shows 'shel...' and the file path '~/.Des...'. The menu bar includes 'Open', 'Save', and a hamburger menu icon. The script content is as follows:

```
1 #! /bin/bash
2
3 # if [condition]
4 # then
5 #     statement
6 # fi # end of if
7
8 a=10
9 if [ $a -ne 9 ]
10 then
11     echo " condition is true"
12 fi
```

The status bar at the bottom indicates 'Tab Width: 8', 'Ln 1, Col 1', and 'INS'.



A screenshot of a terminal window titled 'adhoc@adhoc: ~/Desktop'. The terminal shows the execution of the script created in the previous block:

```
File Edit View Search Terminal Help
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

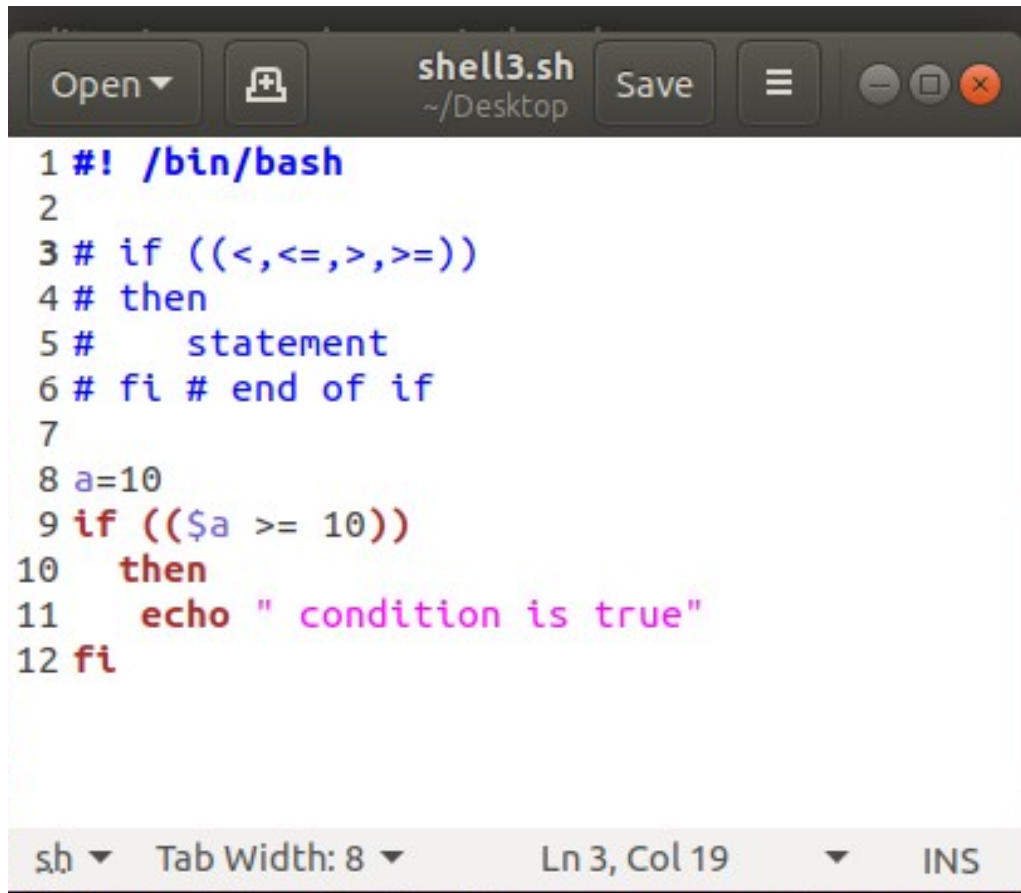
integer comparison

```
-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

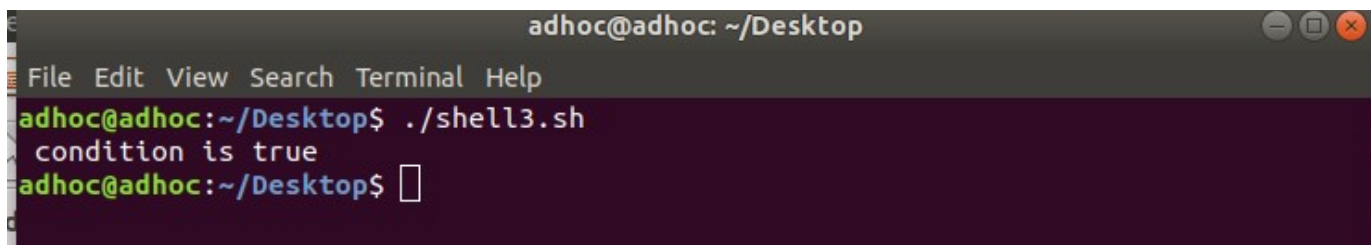
```
= - is equal to - if [ "$a" = "$b" ]  
== - is equal to - if [ "$a" == "$b" ]  
!= - is not equal to - if [ "$a" != "$b" ]  
< - is less than, in ASCII alphabetical order - if [[ "$a" < "$b" ]  
> - is greater than, in ASCII alphabetical order - if [[ "$a" > "$b" ]  
-z - string is null, that is,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

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



```
1 #! /bin/bash
2
3 # if ((<, <=, >, >=))
4 # then
5 #     statement
6 # fi # end of if
7
8 a=10
9 if (($a >= 10))
10 then
11     echo " condition is true"
12 fi
```

sb Tab Width: 8 Ln 3, Col 19 INS



```
ad hoc@ad hoc: ~/Desktop
File Edit View Search Terminal Help
ad hoc@ad hoc:~/Desktop$ ./shell3.sh
condition is true
ad hoc@ad hoc:~/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)

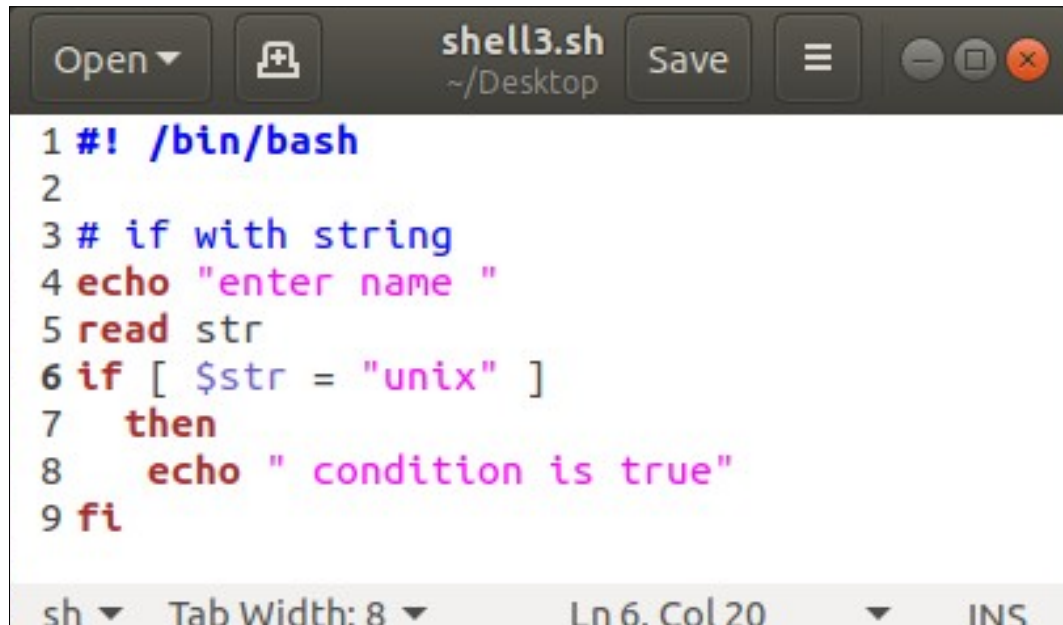
```
Open shel... Save ~/Des...
1 #! /bin/bash
2
3 #test numeric comparison
4 x=5;y=7;z=7
5 test $x -eq $y ; echo $?
6 echo
7 test $x -lt $y ; echo $?
8 echo
9 test $z -gt $y ; echo $?
10 echo
11 test $z -eq $y ; echo $?
```

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$
```

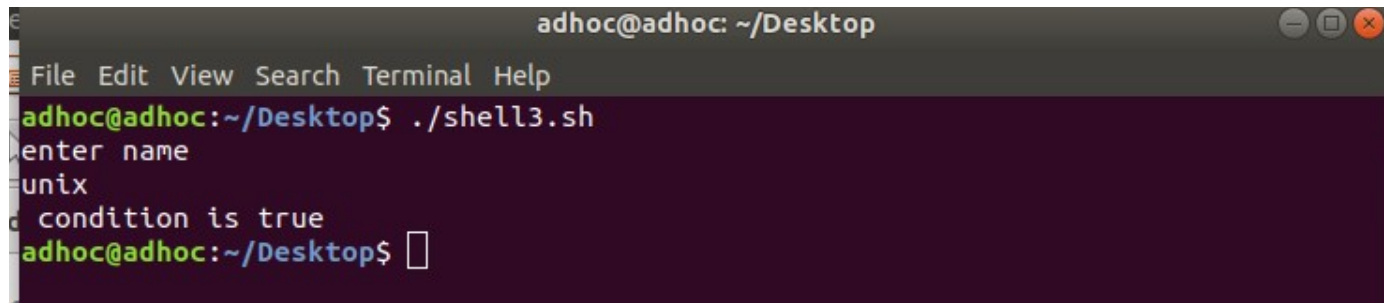


THE IF-THEN “String” STATEMENT



```
1 #! /bin/bash
2
3 # if with string
4 echo "enter name "
5 read str
6 if [ $str = "unix" ]
7 then
8     echo " condition is true"
9 fi
```

sh ▾ Tab Width: 8 ▾ Ln 6, Col 20 ▾ INS

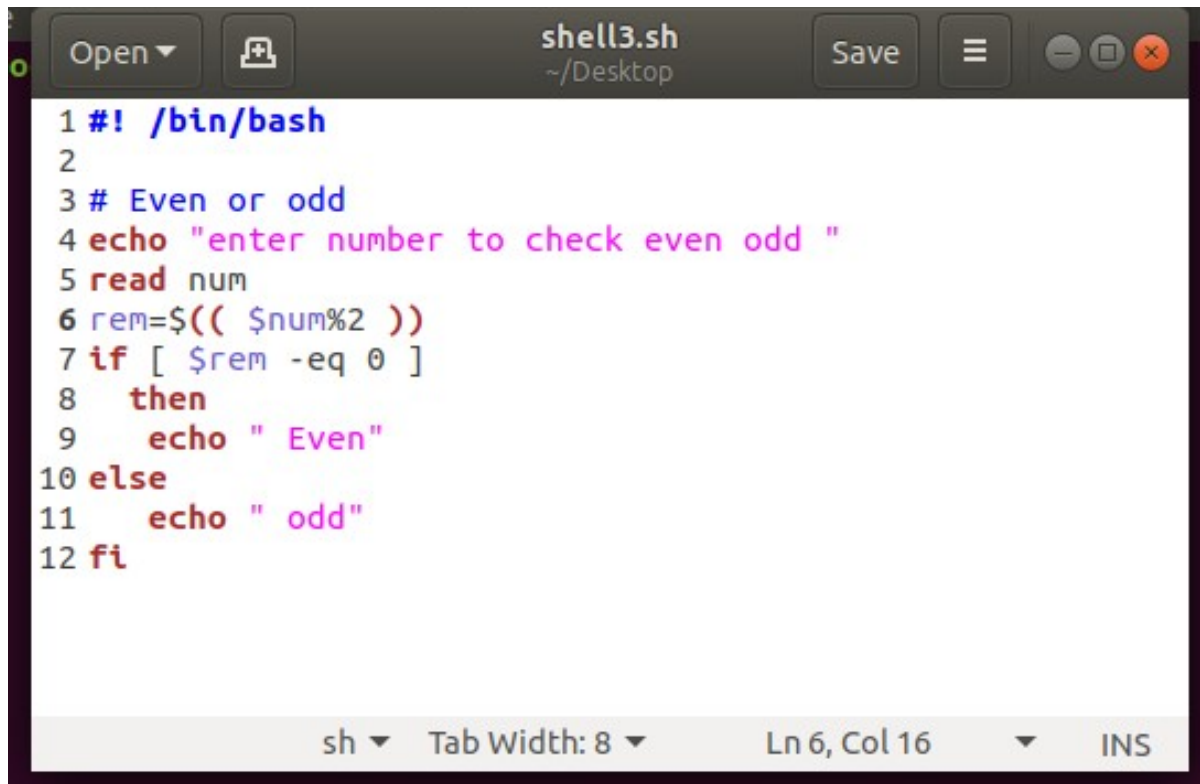


```
File Edit View Search Terminal Help
adhoc@adhoc:~/Desktop$ ./shell3.sh
enter name
unix
condition is true
adhoc@adhoc:~/Desktop$
```

THE IF-THEN-ELSE STATEMENT

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

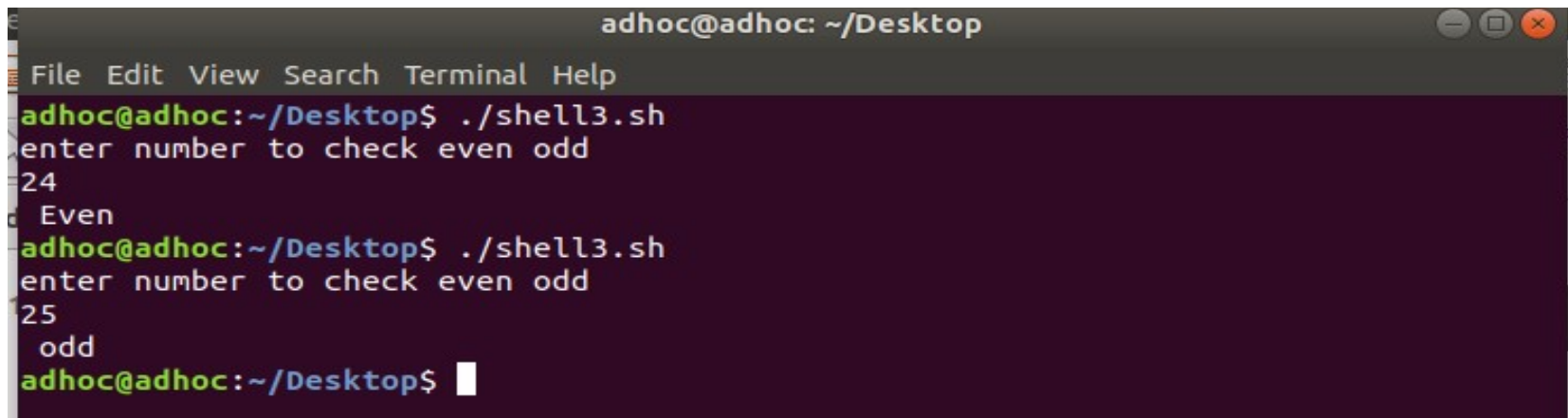
- executes statements-1 if condition is true
- executes statements-2 if condition is false



A screenshot of a text editor window titled "shell3.sh" with the path "~/Desktop". The window has a dark theme and includes buttons for "Open", "Save", and window controls. The script content is as follows:

```
1 #! /bin/bash
2
3 # Even or odd
4 echo "enter number to check even odd "
5 read num
6 rem=$(( $num%2 ))
7 if [ $rem -eq 0 ]
8 then
9     echo " Even"
10 else
11     echo " odd"
12 fi
```

The status bar at the bottom shows "sh", "Tab Width: 8", "Ln 6, Col 16", and "INS".



A screenshot of a terminal window titled "adhoc@adhoc: ~/Desktop". The terminal shows the execution of the script "shell3.sh" twice. The first run takes input "24" and outputs "Even". The second run takes input "25" and outputs "odd".

```
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  Save  ~ / Desktop
1 #! /bin/bash
2 echo "Enter the year (YYYY)"
3 read year
4
5 if [ $((year % 4)) -eq 0 ]
6 then
7     if [ $((year % 100)) -eq 0 ]
8     then
9         if [ $((year % 400)) -eq 0 ]
10        then
11            echo "its a leap year"
12        else
13            echo "its not a leap year"
14        fi
15    else
16        echo "Its a leap year"
17    fi
18 else
19     echo "its not a leap year"
20 fi
sh ▾  Tab Width: 8 ▾  Ln 20, Col 1  ▾  INS
```

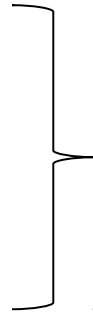
```
ad hoc@a
File Edit View Search Terminal Help
ad hoc@ad hoc:~/Desktop$ ./shell3.sh
Enter the year (YYYY)
4000
its a leap year
ad hoc@ad hoc:~/Desktop$ ./shell3.sh
Enter the year (YYYY)
1900
its not a leap year
ad hoc@ad hoc:~/Desktop$ ./shell3.sh
Enter the year (YYYY)
1982
its not a leap year
ad hoc@ad hoc:~/Desktop$
```



COMPOUND LOGICAL EXPRESSIONS

! not

&& and
|| or



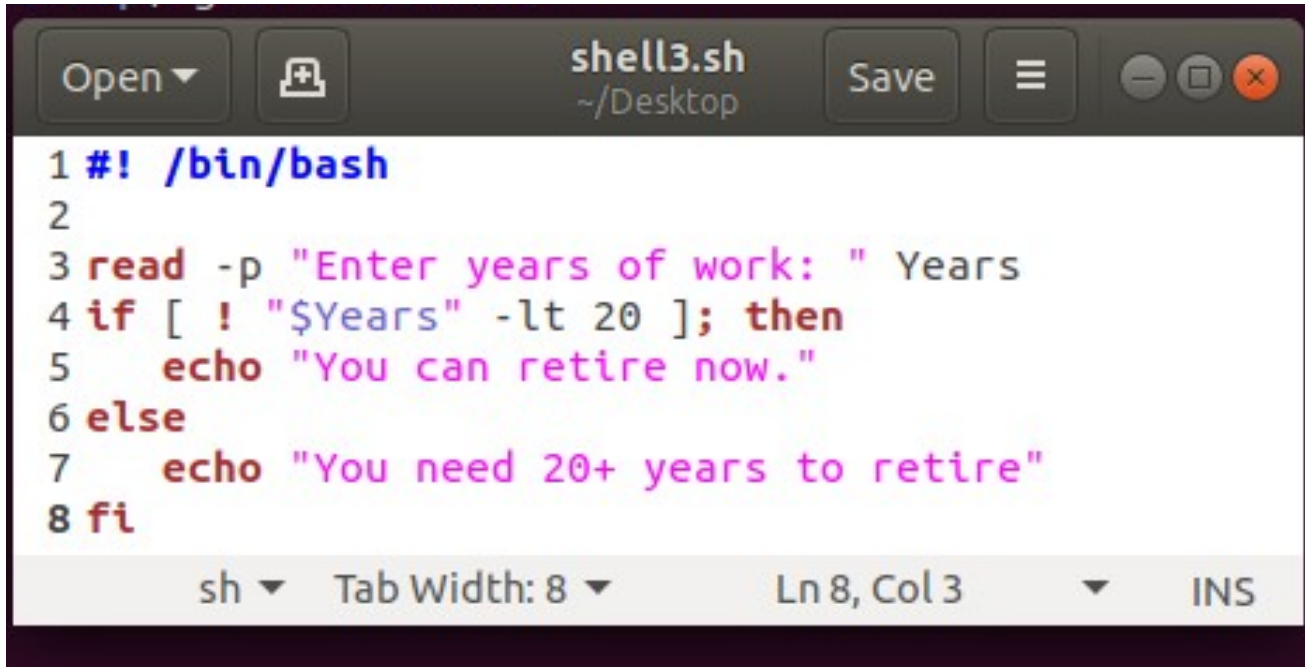
and, or
must be enclosed within

[[]]

EXAMPLE: USING THE ! OPERATOR

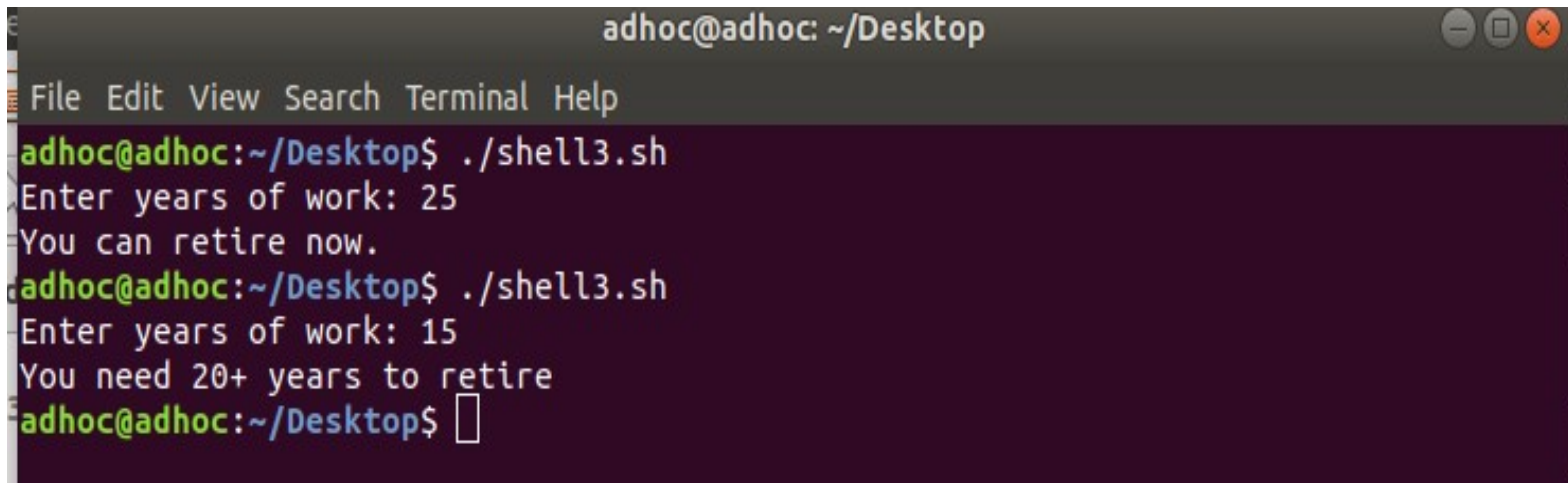
```
#!/bin/bash
```

```
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
```



```
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



```
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"
```

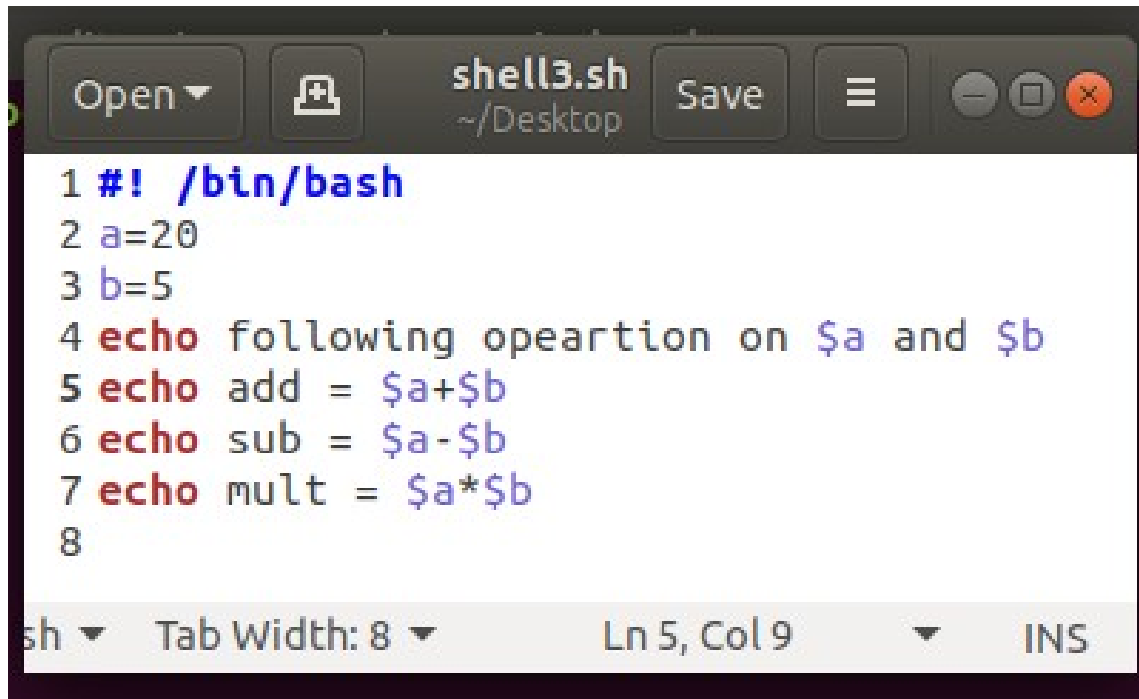
```
fi
```

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
```

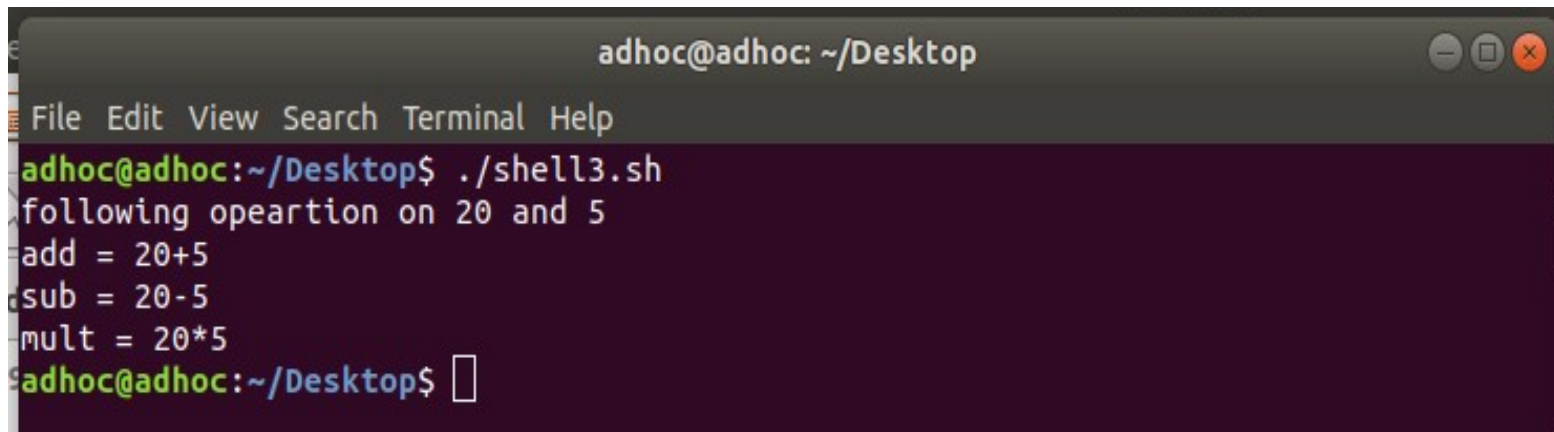
Calculator



A screenshot of a text editor window with a dark theme. The title bar shows 'shell3.sh' and the file path '~/.Desktop'. The editor contains a shell script with the following lines:

```
1 #! /bin/bash
2 a=20
3 b=5
4 echo following opeartion on $a and $b
5 echo add = $a+$b
6 echo sub = $a-$b
7 echo mult = $a*$b
8
```

The status bar at the bottom indicates 'sh', 'Tab Width: 8', 'Ln 5, Col 9', and 'INS'.



A screenshot of a terminal window titled 'adhoc@adhoc: ~/.Desktop'. The terminal shows the execution of the script 'shell3.sh' and its output:

```
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

```
Open ▾  shell3.sh  Save  ▮  -  □  ×  
~/Desktop  
1 #!/bin/bash  
2 a=20  
3 b=5  
4 #using double bracket(( ))  
5 echo following opeartion on $a and $b  
6 echo add = $(( a+b ))  
7 echo sub = $(( a-b ))  
8 echo mult = $(( a*b ))  
9 echo div= $(( a/b ))  
10 echo rem=$(( 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  Save  ≡  -  □  ×
~/Desktop

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$
```






Floating 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"
13
```

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

```
ad hoc@ad hoc: ~/Desktop   
```

```
File Edit View Search Terminal Help
ad hoc@ad hoc:~/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
ad hoc@ad hoc:~/Desktop$ 
```



Floating calculator (Division - scale)

```
Open shel... Save
~/Des...
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 "div as for $a and $b "
8 echo "$a/$b" | bc
9 echo " division not correct"
10 echo "scale=4;$a/$b" | bc
11
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$
```



Floating calculator (math function)

```
Open ▾ *sh... Save
~/Des...

1 #! /bin/bash
2 a=9
3 echo " find sqrt of $a using bc
  and function"
4 echo "square root "
5
6 echo "scale=2;sqrt($a)" | bc -l
7 echo
8 echo "l - call math library "
9 echo
10 echo "power "
11 echo "scale=2;$a^3" | bc -l
12
```

Tab Width: 8 ▾ Ln 12, Col 1 ▾ INS

```
adhoc@adhoc: ~/Desktop
File Edit View Search Terminal Help

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 assign floating result of expression to any variable.
2. check entered number is even or odd
3. Greatest of 3 number
4. Palindrome for 4 digit number

