```
do
j=`echo $i | cut -d"," -f 3`
k=`echo $i | cut -d"," -f 4`
if [ $j -ge 5000 -a $k -eq 10 ]
then
echo $i >>emp1
fi
done
```

Command line Arguments (or) Positional Parameters

* At the time of execution of shell script, if user passes any arguments known as Command line Arguments (or) Positional Parameters

* The Special variables holds positional parameters values. The special variables are $0,$1,$2,$3,$4,$5,$6,$7,$8,$9,$# ,$*,$#,$?,$$

- $0 => Name of the Program
- $1 => 1st parameter value
- $2 => 2nd parameter value
- $3 => 3rd parameter value
- $4 => 4th parameter value
- $5 => 5th parameter value
- $6 => 6th parameter value
- $7 => 7th parameter value
- $8 => 8th parameter value
- $9 => 9th parameter value
- $# => Counts no of arguments
- $* => all parameter values
- $@ => all parameter values but each and every parameter encloses within double quotes.
- $? => It holds last executed command status , If the command executed successfully it holds 0(zero) otherwise non-zero value.
-

${10} → 10th arg
${11} → 11th arg

- $$ => It holds user parent shell process id.

```
$vi hello
for i in $*
do
echo -n "$i  "
done
:wq
$chmod 755 hello
$./hello Tecno soft Solutions
```

```
$vi calc
if [ $# -eq 3 ]
then
c=`echo $1 $2 $3 | bc`
echo $c
else
echo "Invalid no of arguments"
fi
:wq
$chmod 755 calc
$./calc 10 + 4
Or
$sh calc 10 + 4
```

```
$vi checkuser
if [ $# -eq 1 ]
then
 if who | grep $1> /dev/null
then
echo "Logged In"
else
echo "not Logged In"
fi
else
  echo "Invalid no of arguments"
fi

:wq
$chmod 755 checkuser
$./checkuser tecno
```

AVR TECNOSOFT SOLUTIONS PVT. LTD.

AVR TECNOSOFT SOLUTIONS PVT. LTD.

Database connectivity :-

1) write a shell script to connect to oracle db.

```
$vi a1.sh ↩
 sqlplus scott/tiger
:wq
$sh a1.sh ↩
```

2) write a shell script to insert data into oracle emp table.

```
$vi a2.sh ↩
clear
x=101
y="Hari"
sqlplus -s scott/tiger<<Eof
insert into emp(empno,ename)
values($x,'$y');
commit;
Eof
:wq
$sh a2.sh ↩
```

3) write a shell script to retrieve data from oracle emp table.

```
$vi a3.sh ↩
sqlplus -s scott/tiger<<Eof
select * from emp;
Eof
:wq
$sh a3.sh ↩
```

4) write a shell script to call oracle stored procedure

```
$vi a4.sh ↩
sqlplus -s scott/tiger<<Eof
set serveroutput on
exec square(9)
Eof
:wq
$sh a4.sh ↩
```

5) Write a shell script to load flat file data into oracle table.

```
$cat>emp ↩
 101, Hari, 80000, 10 ↩
 102, Sai, 75000, 20 ↩
 103, siva, 60000, 30 ↩
 104, lakshmi, 90000, 10 ↩
Ctrl d
$vi a5.sh ↩
 for i in `cat emp`
 do
  a=`echo $i | cut -d"," -f 1`
  b=`echo $i | cut -d"," -f 2`
  c=`echo $i | cut -d"," -f 3`
  d=`echo $i | cut -d"," -f 4`
sqlplus -s scott/tiger <<Eof
insert into emp(empno, ename, sal, dep
values($a,'$b', $c, $d);
commit;
Eof
done
:wq
$sh a5.sh ↩
```

SVR TECNOSOFT SOLUTIONS PVT. LTD.

* How to connect to oracle

```
$sqlplus ↩
 username : scott ↩
 password : tiger ↩

Sql>!<unixCommand> ↩

Eg:- Sql>!who ↩
     Sql>!ls ↩

Sql>! ↩ [To return to unix]
$ exit ↩ [To return to Sql]
```

Scanned by CamScanner

(5)

⑰ x = "Tecno"

x = $xsoft

$echo $x (It Prints empty value)

$

⑯ x = "Tecno"

x = ${x}soft

$echo $x

Tecnosoft

$

x="${x} Soft"
we need spaces
specify double quotations

Note :- To add some text to existing variable, use { }.

Constant variables :- "readonly" is the keyword to create constant variables.

x = 100
readonly x.

Global variables :- "export" is the keyword, to create global variables.

export y ?

export variablename

x=100
y=200

How to take input from user :- "read" is the keyword to take input from user.

read -p "Enter a name:" name

Syntax: read variablename

How to take input from user with prompt

read -p " prompt : " variablename

Eg:- ① $read -p "Enter a name :" name ↵

Enter a name : Tecnosoft

② $read -s -p "Enter a password :" name ↵

Enter a password: ↵ {without displaying in the prompt}

System defined variables :- "set" is the command, to see all system defined variables along with its values.

Environment Variable

```
$ set ↵
HOME = /home/tecno
SHELL = /bin/bash
LOGNAME = tecno
```

PATH =
MAIL = /var/spool/mail/tecno
MAILCHECK = 60
PS1 = $
PS2 = >

system prompt

.bash_profile
~~PS1~~ PS1 = "tecno$"
PS2 = " ~~xxx~~ "

AVR TECNOSOFT SOLUTIONS PVT. LTD. * HYDERABAD *

(6) a = 10 b = 4 echo

* How to change system prompt?

$\boxed{\text{PS1 = "tecno\$"}}$

* How to change Input prompt?

$\boxed{\text{PS2 = " ---> "}}$

* PATH, system defined variable is used for ~~to~~ set application software paths like Java, oracle, oracle apps

PATH=$PATH
export PATH

Operators

1. Arithmetic operators

+
-
*
/
%

AVR TECNOSOFT SOLUTIONS PVT. LTD. * HYDERABAD *

a=100
b=20
$a > $b
$a -lt $b false

2. Relational operators

a) Numeric comparision operators

-lt (less than)
-le (less than or equal to)
-gt (greater than)
-ge (greater than or equal to)
-eq (equal to)
-ne (not equal to)

b) String comparision operators

<
>
=
!=

3. Logical operators

-a (logical and)
-o (logical or)
! (logical not)

Note :- Each and Every operator, should contain space before and after operature except $\boxed{\text{assignment operator}}$.

4. Assignment operator

```
a=10
b=4
$echo $a + $b ←
    10 + 4
```

expr :- expr is the keyword to convert ~~into~~ string expression to integer expression.

```
Eg:① $echo `expr $a + $b` ←
        14
```

1) expr integer expression
   Eg:- `expr $a + $b`
2) $((integer expression))
   Eg:- $(($a + $b))
3) let integer expression
   Eg:- let c=$a + $b

```
② $echo `expr $a \* $b` ←
        40
```

Float arithmetic

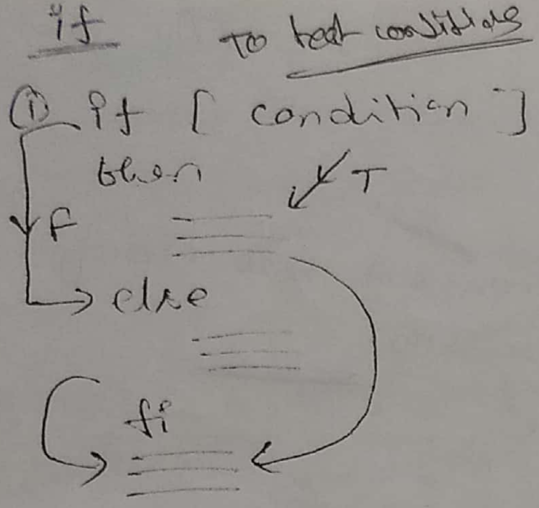
```
a=10.4
b=3.2
$echo $a + $b
    10.4 + 3.2
$echo ~~$a~~ `echo $a + $b | bc`
    13.6
$echo c=`echo $a - $b | bc`
    7.2     {#echo $c}
```

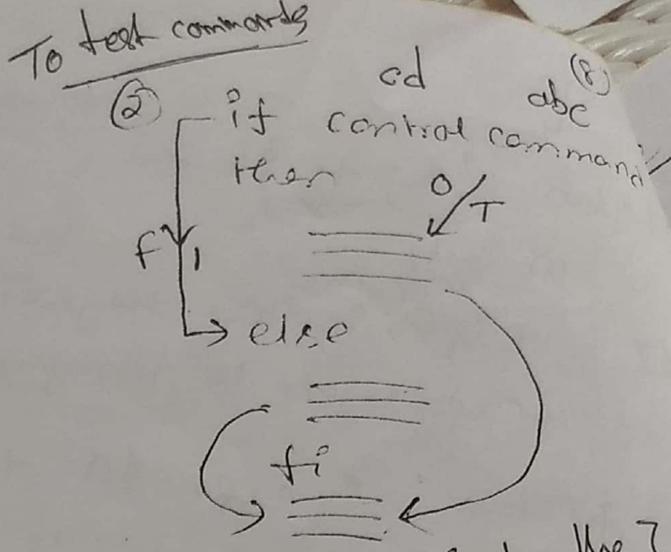
#scale=3

Note :- bc is the command, to perform any float related calculations.

```
$bc ← (binary
       calcula
10+4 ←
 14
10.3+4.2 ←
 14.5
10>4 ←
 1
10<4 ←
 0
10/4 ←
 2
sqrt(200) ←
 14
scale=2
sqrt(200)
 14.14
10/4
 2.5
ctrl d
```

if





~~if then~~ # if [-f file]

file test commands

1) -f ⇒ True, if it is regulary file
2) -d ⇒ True, " " " directory file
3) -l ⇒ True, " " " link file
4) -b ⇒ True, " " " block special file } device files
5) -c ⇒ True, " " " Characte " "
6) -e ⇒ True, if file ~~exist~~
7) -s ⇒ True, if file is not empty.
8) -r ⇒ True, if file has read permission
9) -w ⇒ True, if file has write permission
10) -x ⇒ True, if file has execute permission.

AVR TECNOSOFT SOLUTIONS PVT. LTD. HYDERABAD

AVR TECNOSOFT SOLUTIONS PVT. LTD.

String test commands

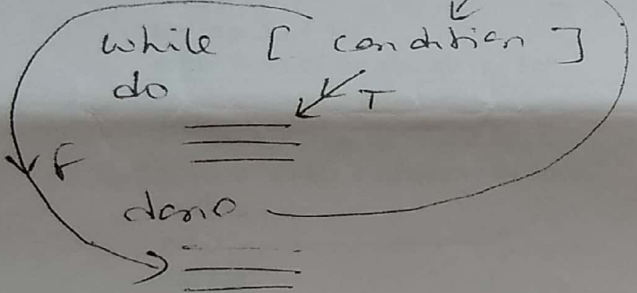
1) -z ⇒ True, if String is empty
2) -n ⇒ True, if String is not empty.

$\boxed{\#}$ is a single line comment.

Case

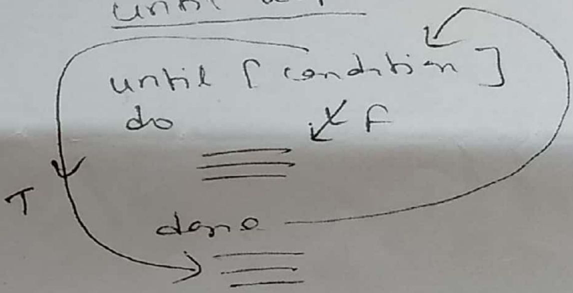
```
case variable in
Pattern1) ~~stmts~~
          ___
          ___
        ;;  # to terminate the case
Pattern2) ___
          ___
        ;;
   :
Pattern n) ___
           ___
        ;;
  *) ___
     ___
    ;;
esac
```

while loop



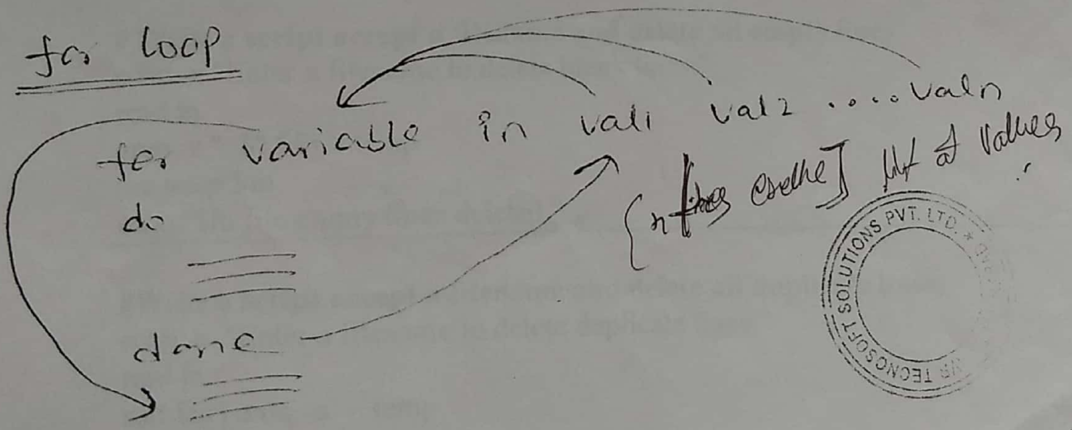
```
while [ condition ]
do
   ___
   ___
done
   ___
```

until loop



```
until [ condition ]
do
   ___
   ___
done
   ___
```

for loop



```
for variable in val1 val2 ....valn
do
   ___
   ___
done
   ___
```

break :- break is the keyword, to terminate the loop

```
while [ condition ]
do
   ___
   break
   ___
done
   ___
```

<u>Continue</u> → Continue is the keyword, to start the loop again.

```
while [ condition ]
do
    ____
    ____
    continue
    ____
    ____
done
    ____
    ____
```

AVR TECNOSOFT SOLUTIONS PVT. LTD. * HYD *

Shell Scripting Programs

```
#Example of variables
a=10
b=20
echo "a is : $a"
echo "b is : $b"
```

```
#Write a script accept 2 intgere no's and display
echo -e "Enter a number1 : \c"
read a
echo -e "Enter a number2 : \c"
read b
echo "a value is : $a"
echo "b value is : $b"
```

```
#Write a script accept 2 intgere no's and find sum
echo -e "Enter a number1 : \c"
read a
echo -e "Enter a number2 : \c"
read b
c=`expr $a + $b`
echo "Sum of $a and $b is : $c"
```

SOLUTIONS PVT. LTD. TECNOSOFT

```
#Write a script accept a filename and open
echo -n "Enter a filename to open:"
read fn
echo "------------------------"
cat $fn
echo "------------------------"
```

```
# Write a script accept a filename and delete all empty lines
echo -n "Enter a filename to delete blank lines:"
read fn
grep -v "^$" $fn > temp
mv temp $fn
echo "$fn file empty lines deleted."
```

```
#Write a script accept a filename and delete all duplicate lines.
echo -n  "Enter a filename to delete duplicate lines:"
read fn
sort $fn | uniq -u  > temp
mv temp $fn
echo "$fn file Duplicate lines are deleted."
```

```
#write a script accept a number and check the given no is +ve or -ve.
echo -n "Enter a number :"
read n
if [ $n -gt 0 ]
then
echo "$n is a +ve no."
else
echo "$n is a -ve no."
fi
```

[$n -gt 0]

```
#Write a script accept a intger no and check the given no is even or odd number
echo -n "Enter a number:"
read n
if [ `expr $n % 2` -eq 0 ]
then
echo "$n is  an Even no."
else
echo "$n is  an Odd no."
fi
```

AVR TECNOSOFT SOLUTIONS PVT. LTD.

```
#Write a script accept 2 strings and check the given 2 strings are equal or not
echo -n "Enter a string1 : "
read str1
echo -n "Enter a string2 : "
read str2
if [ "$str1"  =  "$str2"  ]
then
echo "Strings are Equal"
else
echo "Strings are not Equal"
fi
```

```
#Write a script accept a filename and delete given file
echo -n "Enter a filename:"
read fn
if rm $fn
then
echo "$fn file deleted."
else
echo "No such file"
fi
```

```
#Write a script check today is Sunday or not
x=`date +%a` # x=`date | cut -c 1-3`
if [ $x = "Sun" ]
then
echo "Yes.Today is Sunday"
else
echo "Sorry. Today is $x day"
fi
```

AVR TECNOSOFT SOLUTIONS PVT. LTD.

#Write a script accept a user and check the given user exist or not

```
echo -n "Enter a username : "
read un
if grep -w $un /etc/passwd > /dev/null
then
echo "$un user exist"
else
echo "$un user doesn't exist"
fi
# /dev/null is a special file.It is used for to write unwanted output.
```

#Write a script accept a user and check the user is connect to the server or not.

```
echo -n  "Enter user name:"
read un
if grep -w $un /etc/passwd > /dev/null
then
 if who | grep -w $un > /dev/null
 then
   echo "Logged In"
 else
   echo "Not Logged In"
 fi
else
echo "$un user doesn't exist",
fi
```

#Write a script accept a filename and open

```
echo -n "Enter a filename : "
read fn
if [ -e $fn ]
then
 if [ -f $fn ]          elif [-d $fn ]
 then
  if [ -r $fn ]
  then
   cat $fn
  else
  echo "No read permission"
  #chmod 644 $fn
  #cat $fn
  # echo "No read permission"
  fi
 else
 echo "It is not a file"
 fi
else
echo "$fn file dosen't exist"
fi
```

#Write a script accept a filename and check the file is regular file or directory file

```
echo -n "Enter a filename : "
read fn
if [ -e $fn ]
then
 if [ -f $fn ]
 then
  echo "$fn is a regular file"
 elif [ -d $fn ]
 then
  echo "$fn is a directory file"
 else
 echo "It is not a file or directory"
 fi
else
 echo "$fn file dosen't exist"
fi
```

AVR TECNOSOFT SOLUTIONS PVT. LTD.

#Write a script accept 2 filenames and chjeck the given 2 files are same or not

```
echo -n "Enter a filename 1 :"
read fn1
echo -n "Enter a filename 2 :"
read fn2
x=`cmp $fn1 $fn2`
if [ -z "$x" ]
then
echo "Given 2 files are same"
else
echo "Given 2 files are not same"
fi
```

z - String validations

#write a script aceept a string and check the given string is empty or not

```
echo -n "Enter a string : "
read str
if [ -z "$str" ]
then
echo  "Given string empty"
else
echo "Given string not empty"
fi
```

```
#case example
echo "enter a number b/w 1 to 4:"
read n
case $n in
1)echo "one";;
2)echo "two";;
3)echo "three";;
4)echo "four";;
*)echo "invalid number";;
esac
```

AVR TECNOSOFT SOLUTIONS PVT. LTD

#Write a script accept a single character and check the given character is alphabet or digit or special character.

```
echo -n "Enter a single character: "
read ch
case $ch in
[a-zA-Z])echo "Alphabet";;
[0-9])echo "Digit";;
[^a-zA-Z0-9])echo "Special Character";;
*)echo "You enetered more than one character";;
esac
```

#write a script accept a single character and check the given character is special character or digit or vowel or consonent

```
echo -n "Enter a single character: "
read ch
case $ch in
[^a-zA-Z0-9])echo "Special character";;
[0-9])echo "Digit";;
[AEIOUaeiou])echo "Vowel";;
[^AEIOUaeiou])echo "Consonent";;
*)echo "You enetered more than one character";;
esac
```

#write a script accept a month and display quarter of the given month

```
echo -n "Enter a month[mon] : "
read mm
case $mm in
jan|feb|mar)echo "1st Quarter";;
apr|may|jun)echo "2nd Quarter";;
jul|aug|sep)echo "3rd Quarter";;
oct|nov|dec)echo "4th Quarter";;
*)echo "Invalid month.";;
esac
#[Jj]an|[Ff]eb|[Mm]ar)echo "1st Quarter";;
# [Aa][Uu][Gg]
# [Aa][Uu][Gg]*
```

AVR TECNOSOFT SOLUTIONS

```
#Example of Menu Program
clear
tput cup 6  10
echo "MAIN MENU"
tput cup 7 10
echo "*********"
tput cup 8 10
echo "1.Date"
tput cup 9 10
echo "2.List of users"
tput cup 10 10
echo "3.Open a file"
tput cup 11 10
echo "4.delete a file"
tput cup 12 10
echo "5. Exit"
tput cup 20 5
echo "enter a choice[1-5] : "
read choice
case $choice in
 1)echo "Today date is : `date`";;
 2)who ;;
 3)sh fopen.sh;; # ./fopen.sh
 4)sh del.sh;; #./del.sh
 5)echo "Thank You"
 exit(;;)  # to terminate the program
 *)echo "choice wrong. try again";;
esac
```

```
#Write a script print no's from 1 to 10
i=1
while [ $i -le 10 ]
do
echo $i
i=`expr $i + 1`
done
```

```
#Write a script accept a string and display reverse of the given string
echo -n "Enter a string : "
read str
l=`echo $str | wc -c` # length of the string
while [ $l -gt 0 ]
do
ch=`echo $str | cut -c $l`
temp=$temp$ch
l=`expr $l - 1`
done
echo "Reverse of $str is : $temp"
```

```
#Example of while loop
ans="y"
while [ $ans  =  "y" ]
do
echo "Enter a filename to open:"
read fn
if [ -e $fn -a -f $fn ]
then
cat $fn
else
echo "no such file"
fi
echo "Do u want to open one more file [y/n] : "
read ans
done
```

```
#Example of while loop
while true # until false          true
do
echo "Enter a filename to open:"          echo -n "enter a the name"
read fn
if [ -e $fn -a -f $fn ]
then
cat $fn
break
else
continue
fi
done
```

```
#Example of sleep
# sleep is used for to stop the execution specified no of seconds.
while true
do
clear
tput cup 5 8
echo "WELCOME TO"
sleep 2
clear
tput cup 5 8
echo "TECNOSOFT"
sleep 2
done
```

```
#Write a script create "n"  no of users
echo -n "Enter no of users to create : "
read n
$i=1
while [ $i -le $n ]
do
$x=tecno$i # It creates users with tecno name
useradd $x
i=`expr $i + 1`
done
```

```
#Example of for loop
for i in 1 2 3 4 5
do
echo $i
done
```

```
#Example of for loop
a=10
b=20
c=30
for i in a b c
do
echo $i
done
```

```
#Example of for loop
a=10
b=20
c=30
for i in $a $b $c
do
echo $i
done
```

```
#Write a script to display all sub directories of current directory
for i in *
do
if [ -d  $i ]
then
echo  $i
fi
done
```

```
#Write a script to display all empty files in the current directory
for i in *
do
if [ ! -s $i ]
then
echo  $i # rm $i => To delete empty files
fi
```

```
done
```

```
#Write a script to display alll exe files in the current directory
for i in *
do
if [ -f $i -a -r $i -a -x $i ]
then
echo $i
fi
done
```

```
#Write a script to  display details of employees who are receiving salary more than 5000 from
emp file

#101,hari,9000,10
#102,madhu,4000,20
#103,anu,000,30
#104,priya,8000,10

for i in `cat emp`
do
j=`echo $i | cut -d"," -f 3`
if [ $j -ge 5000 ]
then
echo $i
fi
done
```

R TECNOSOFT SOLUTIONS PVT. LTD

```
#Write a script   retrieve  details of employees who are receiving salary more than 5000 in
deptno 10 from emp file and insert into emp1 file

#101,hari,9000,10
#102,madhu,4000,20
#103,anu,666666000,30
#104,priya,8000,10

for i in `cat emp`
do
j=`echo $i | cut -d"," -f 3`
k=`echo $i | cut -d"," -f 4`
if [ $j -ge 5000 -a $k -eq 10 ]
then
echo $i >> emp1
fi
done
```

R TECNOSOFT SOLUTIONS PVT. LTD