Heaven’s Light is Our Guide

Computer Science And Engineering

Rajshahi University of Engineering and Technology

Course No: CSE3202

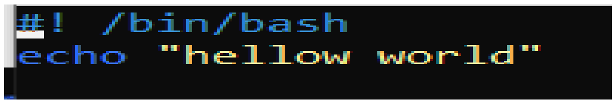
Course Title: Operating system

Date of Submission: 31-10-22

| Submitted To | Submitted By |
|---|---|
| Mohiuddin Ahmed<br>Lecturer,<br>Department Of Computer Science And Engineering<br>Rajshahi University Of Engineering And Technology | Nazmul Haque<br>Roll: 1803109<br>Section: B<br>Department Of Computer Science And Engineering<br>Rajshahi University Of Engineering And Technology |

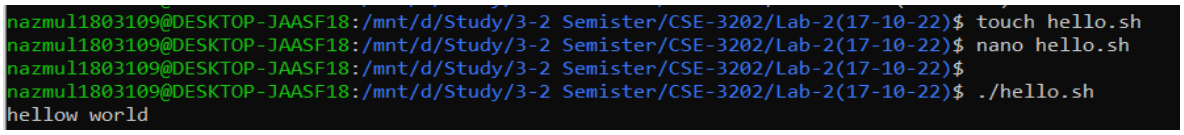
## 1)First Shell Code:

### Code:



```
#! /bin/bash
echo "hellow world"
```

### Output:

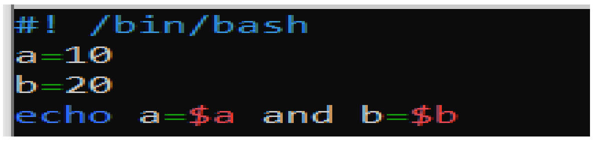


```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch hello.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano hello.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./hello.sh
hellow world
```

**Note:** First of all create hello.sh file using touch. Then open that file using nano and write code. Finally run that code using ./file_name(such as ./hello.sh). Here, echo work as a printf("") function like c programming.

## 2)Variables:

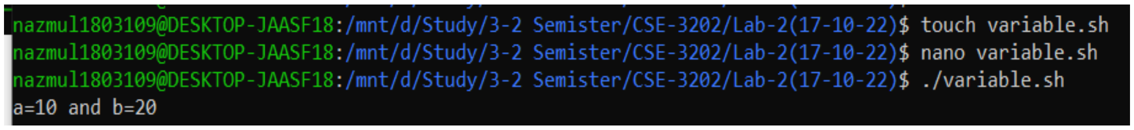
### Code:



```
#! /bin/bash
a=10
b=20
echo a=$a and b=$b
```

**Note:** To assign the value of a variable we need to use '$' sign at the front of that variable.

### Output:

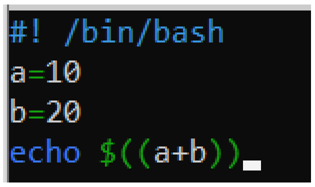


```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch variable.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano variable.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./variable.sh
a=10 and b=20
```

## 3)Arithmetic Operation:

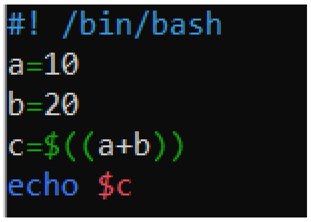
### i) Summation Operation:

### Code:



```
#! /bin/bash
a=10
b=20
echo $((a+b))
```

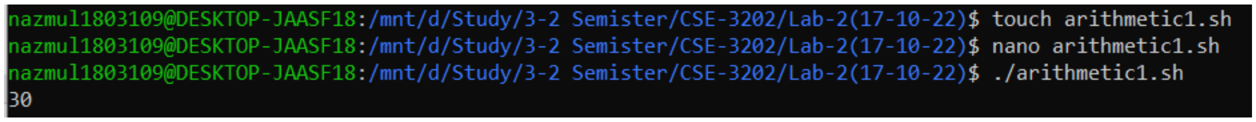
Or,



```
#! /bin/bash
a=10
b=20
c=$((a+b))
echo $c
```

**Note:** We can use one of this two way.

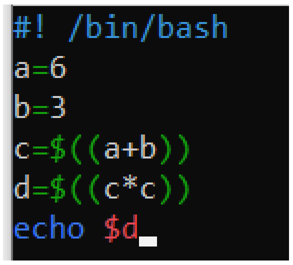
## Output:



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch arithmetic1.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano arithmetic1.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./arithmetic1.sh
30
```

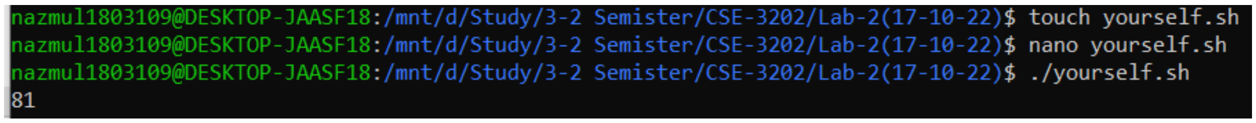
## ii)Try Yourself: (a+b)^2

## Code:



```
#! /bin/bash
a=6
b=3
c=$((a+b))
d=$((c*c))
echo $d
```

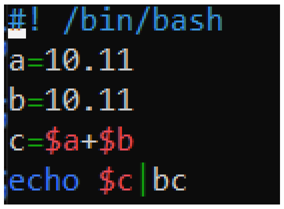
## Output:



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch yourself.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano yourself.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./yourself.sh
81
```

## iii) Sumation of floating point number:

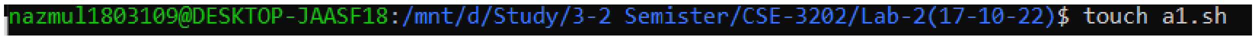
## Code:



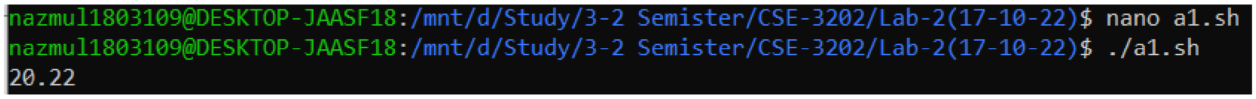
```
#! /bin/bash
a=10.11
b=10.11
c=$a+$b
echo $c|bc
```

**Note:** Here use bc which is basic calculator liabrary.

## Output:



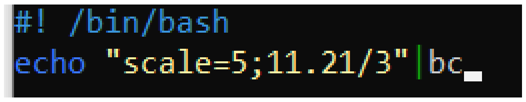
```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch a1.sh
```



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano a1.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./a1.sh
20.22
```

## iv) Precision of number:

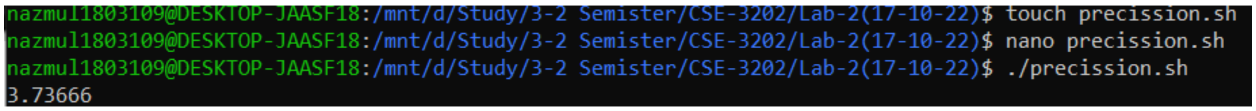
## Code:



```
#! /bin/bash
echo "scale=5;11.21/3"|bc
```

**Note:** if we want to 5 digit after decimal point then 5 should assign in scale.

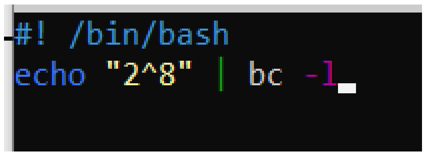
## Output:



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch precission.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano precission.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./precission.sh
3.73666
```

## v) Power:

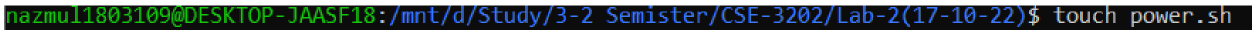
## Code:



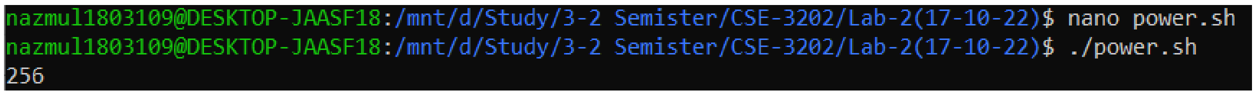
```
#! /bin/bash
echo "2^8" | bc -l
```

#-l is used to invoke math library

## Output:



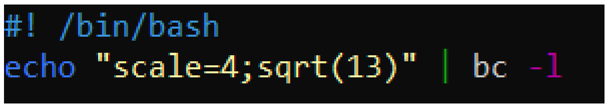
```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch power.sh
```



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano power.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./power.sh
256
```

## vi) Square Root:

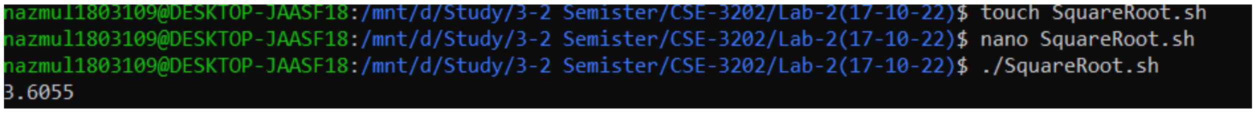
### Code:



```
#! /bin/bash
echo "scale=4;sqrt(13)" | bc -l
```

Note: to calculate square of a number need to basic calculator liabrary function.

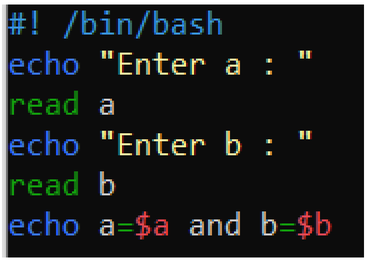
### Output:



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch SquareRoot.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano SquareRoot.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./SquareRoot.sh
3.6055
```

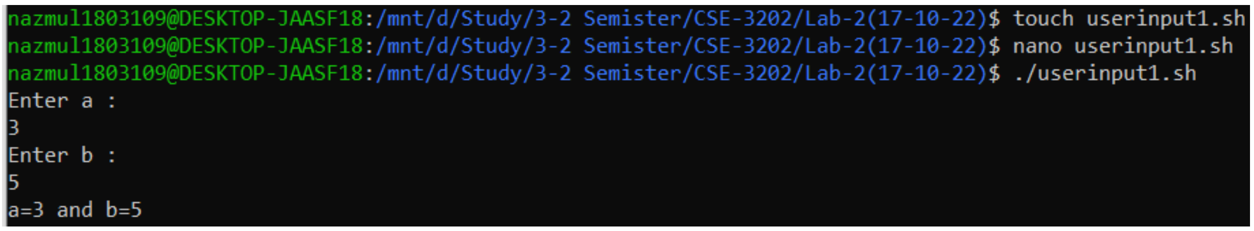
## 4)Input from User

### Code:



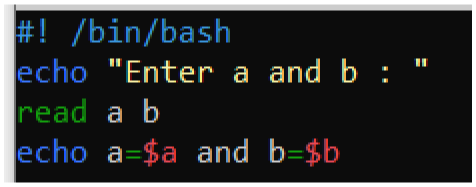
```
#! /bin/bash
echo "Enter a : "
read a
echo "Enter b : "
read b
echo a=$a and b=$b
```

### Output:



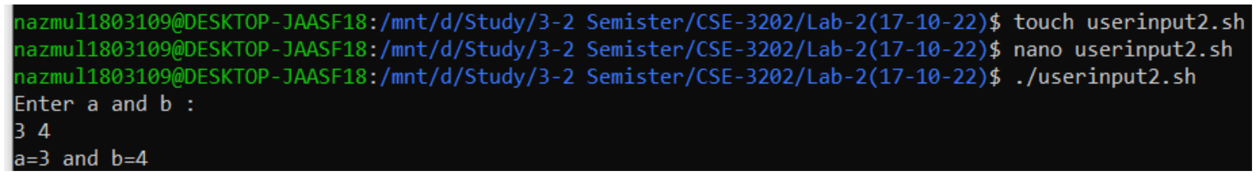
```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch userinput1.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano userinput1.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./userinput1.sh
Enter a :
3
Enter b :
5
a=3 and b=5
```

Or,



```bash
#! /bin/bash
echo "Enter a and b : "
read a b
echo a=$a and b=$b
```

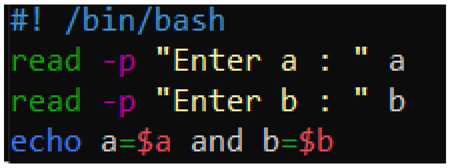
## Output:



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch userinput2.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano userinput2.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./userinput2.sh
Enter a and b :
3 4
a=3 and b=4
```

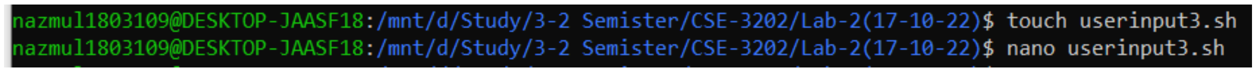
Or,

## Code:

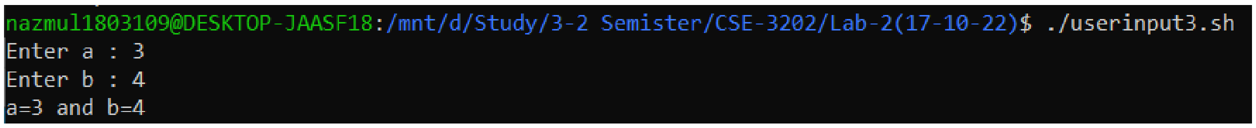


```bash
#! /bin/bash
read -p "Enter a : " a
read -p "Enter b : " b
echo a=$a and b=$b
```

## Output:



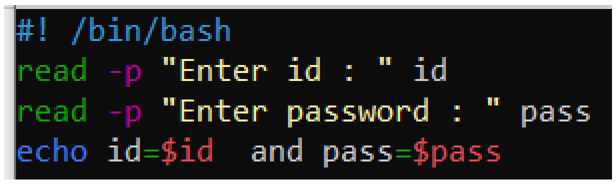
```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch userinput3.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano userinput3.sh
```



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./userinput3.sh
Enter a : 3
Enter b : 4
a=3 and b=4
```

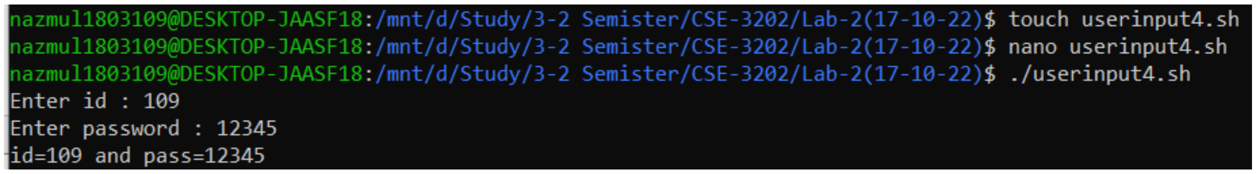
Or,

## Code:



```bash
#! /bin/bash
read -p "Enter id : " id
read -p "Enter password : " pass
echo id=$id  and pass=$pass
```

## Output:

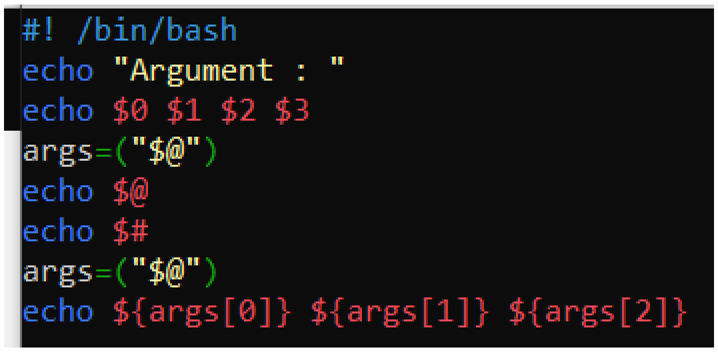


```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch userinput4.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano userinput4.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./userinput4.sh
Enter id : 109
Enter password : 12345
id=109 and pass=12345
```

**Note:** we can use any of them way to take user input using read.

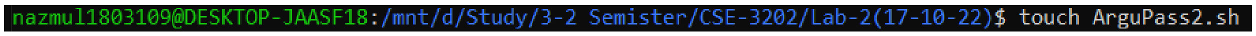
## 5) Pass Argument during execution

### Code:

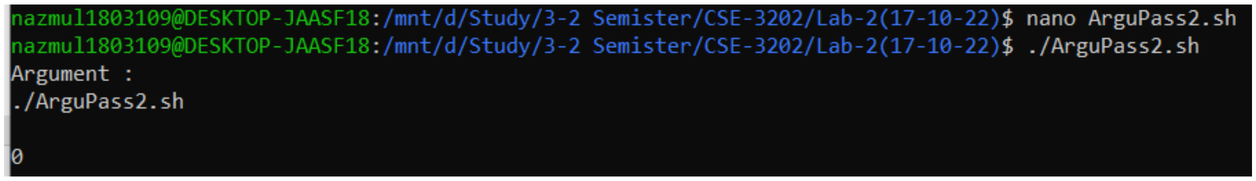


```
#! /bin/bash
echo "Argument :  "
echo $0 $1 $2 $3
args=("$@")
echo $@
echo $#
args=("$@")
echo ${args[0]} ${args[1]} ${args[2]}
```

### Output:



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch ArguPass2.sh
```



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano ArguPass2.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./ArguPass2.sh
Argument :
./ArguPass2.sh

0
```

## 6) Conditional Statement- If:

### Syntax:

```
if  [ condition ]
        then
         #code to be executed if the condition is satisfied
     else
        #code to be executed if the condition is  not satisfied
fi
```

```
if  [ condition ] && [condition]
        then
         #code to be executed if the condition is satisfied
```

```
            else
                #code to be executed if the condition is  not satisfied
        fi
```

```
        if  [ condition ] || [condition]
                then
                 #code to be executed if the condition is satisfied
            else
                #code to be executed if the condition is  not satisfie
        fi
```

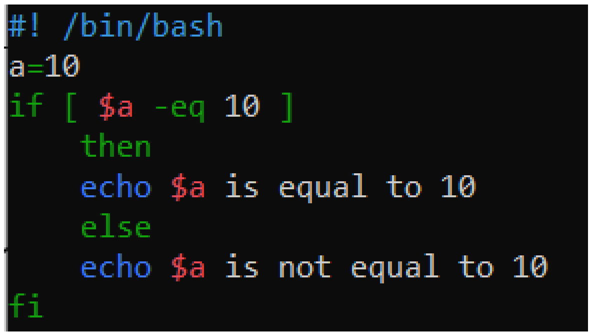
**Condition:**

- **-eq :** equals to
  example:  if [ $var -eq 0 ]
- **-ne :** not equals to
  example: if [ $var -q ne 0 ]
- **-gt** Or **> :** Greater than
  example: if [ $var -gt  0 ]
  if [ $var > 0 ]
- **-lt** Or **<** : Less than
  example: if [ $var -gt  0 ]
  if [ $var > 0 ]
- **-ge**  Or **>= :** Greater than equals to
  example: if [ $var -ge  10 ]
  if [ $var >= 10 ]
- **-le**  Or **<= :** Greater than equals to
  example: if [ $var -le  10 ]
  if [ $var <= 10 ]

using this above syntax solve some simple problem below.

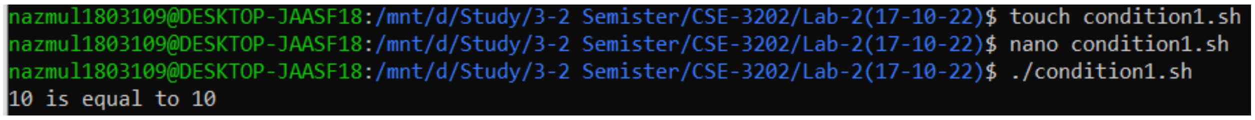
**i)use of Equal operator:**

**Code:**



```
#! /bin/bash
a=10
if [ $a -eq 10 ]
    then
    echo $a is equal to 10
    else
    echo $a is not equal to 10
fi
```

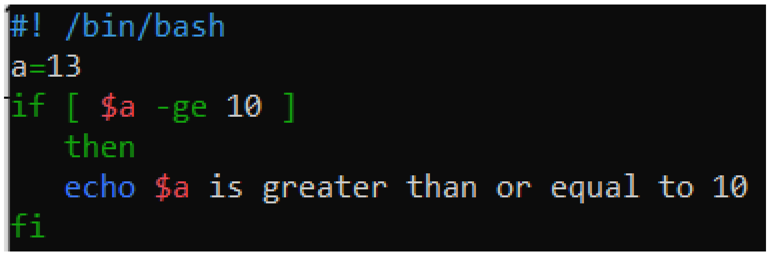
**Output:**



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch condition1.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano condition1.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./condition1.sh
10 is equal to 10
```

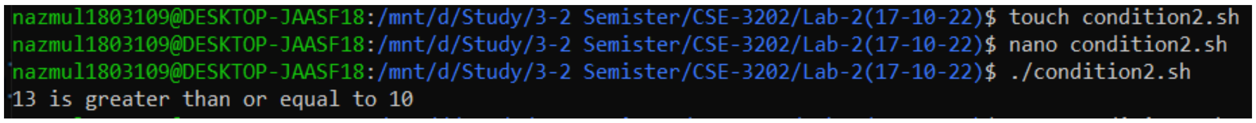
**ii)Use of greater than or equal operator:**

**Code:**



```
#! /bin/bash
a=13
if [ $a -ge 10 ]
   then
   echo $a is greater than or equal to 10
fi
```

**Output:**



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch condition2.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano condition2.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./condition2.sh
13 is greater than or equal to 10
```

**iii) Conditional for String:**

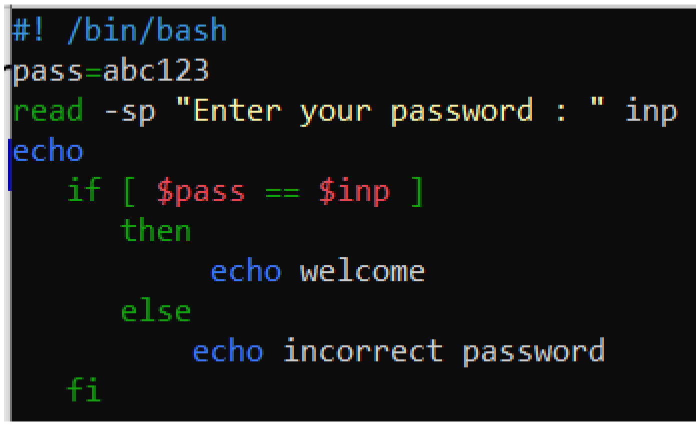
- **== :** equals to
  example: if [ $str == “value” ]
- **!= :** not equals to
  example: if [ $str != “value” ]
- **< :** is less than in ASCII value
  example: if [ $var -q ne 0 ]

- **> :** is greater than in ASCII value

  example: if [ $var -q ne 0 ]

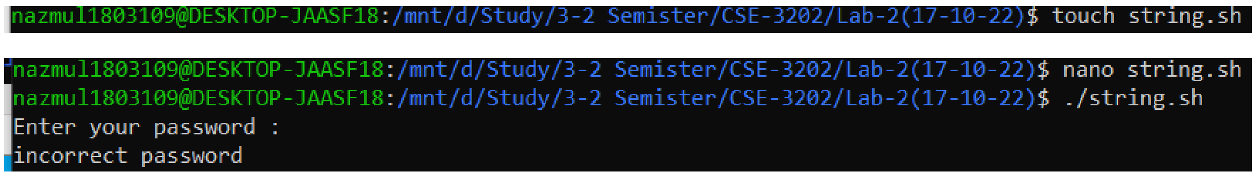
**example:**

**Code:**



```
#! /bin/bash
pass=abc123
read -sp "Enter your password : " inp
echo
   if [ $pass == $inp ]
      then
           echo welcome
      else
          echo incorrect password
   fi
```

**Output:**



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch string.sh
```

```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano string.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./string.sh
Enter your password :
incorrect password
```

## 7) Loop Statement:

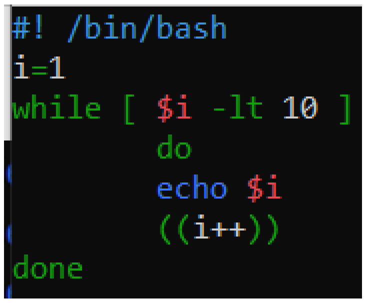
- ### While:

  **Syntax:**

```
while  [ condition ]
      do
     #code to be executed as long as the condition is satisfied
done
```

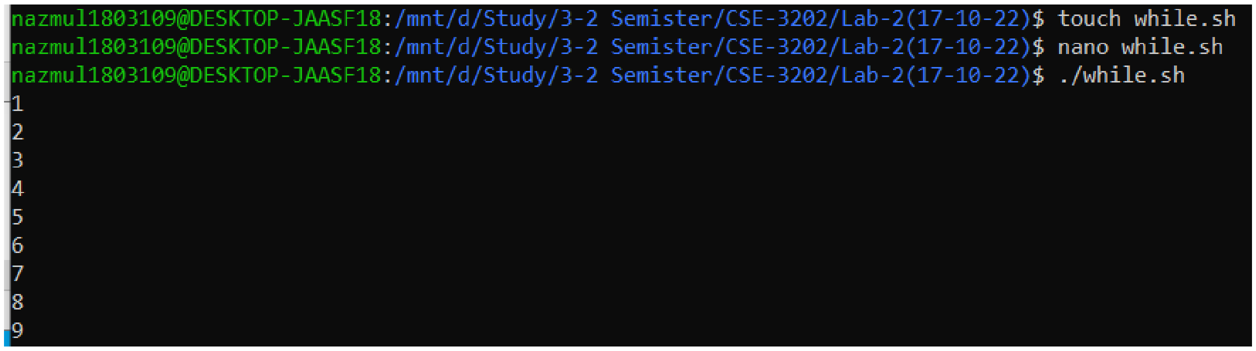
**Ex.**

**Code:**



```
#! /bin/bash
i=1
while [ $i -lt 10 ]
        do
        echo $i
        ((i++))
done
```

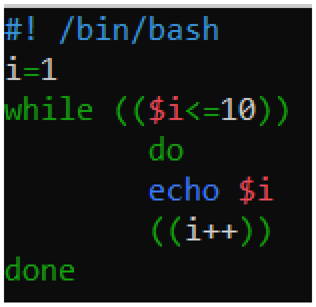
**Output:**



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch while.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano while.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./while.sh
1
2
3
4
5
6
7
8
9
```

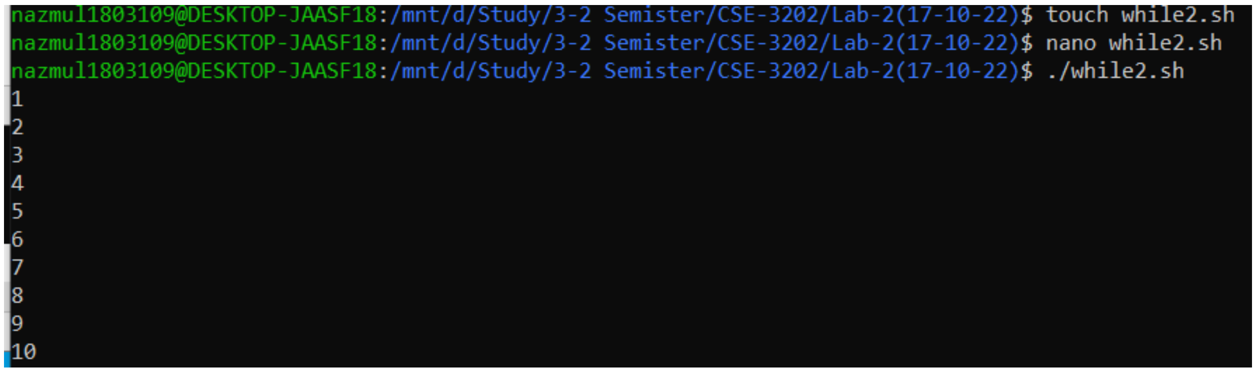
Or,

**Code:**



```
#! /bin/bash
i=1
while (($i<=10))
        do
        echo $i
        ((i++))
done
```

**Output:**



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch while2.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano while2.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./while2.sh
1
2
3
4
5
6
7
8
9
10
```

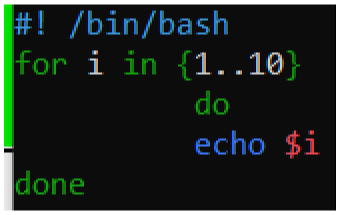
- **For :**

**Syntax:**

| |
|---|
| for  variable in {range_start..range_end}<br><br>        #code to be executed as long as the condition is satisfied<br><br>done |
| for  ((start; condition; stepsize))<br><br>          do<br><br>        #code to be executed as long as the condition is satisfied<br><br>done |

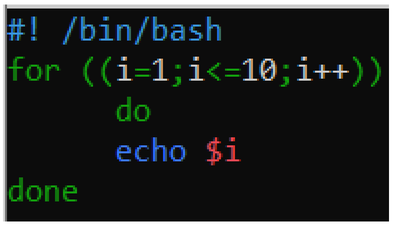
Ex.

**Code:**



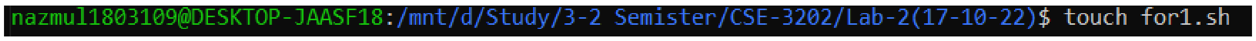
```
#! /bin/bash
for i in {1..10}
          do
          echo $i
done
```

**Or,**

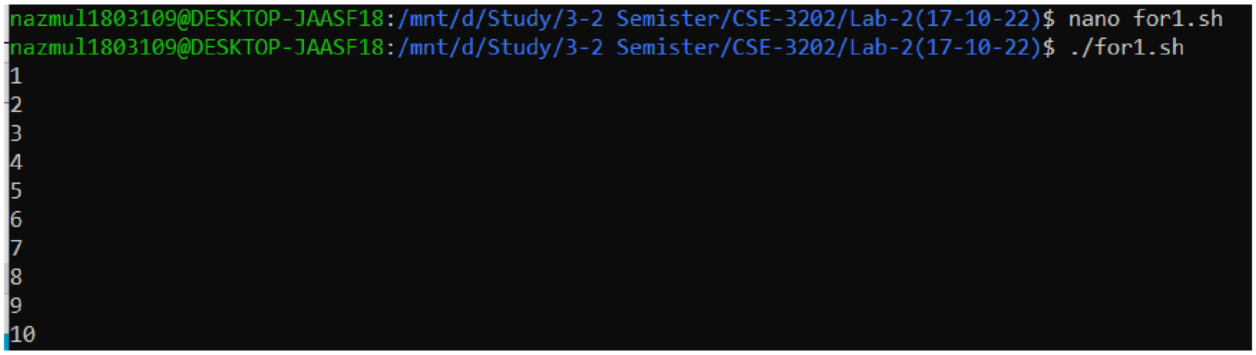


```
#! /bin/bash
for ((i=1;i<=10;i++))
      do
      echo $i
done
```

**Output:**



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch for1.sh
```


```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano for1.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./for1.sh
1
2
3
4
5
6
7
8
9
10
```

## 8) Array

### a) Indirect Declaration

```
ARRAYNAME[INDEXNR]=value
```

### b) Explicit Declaration

```
declare -a ARRAYNAME
```

### c) Compound Assignment

```
ARRAYNAME=(value1 value2  .... valueN)
```

**Or**

```
ARRAYNAME=([1]=10 [2]=20 [3]=30)
```

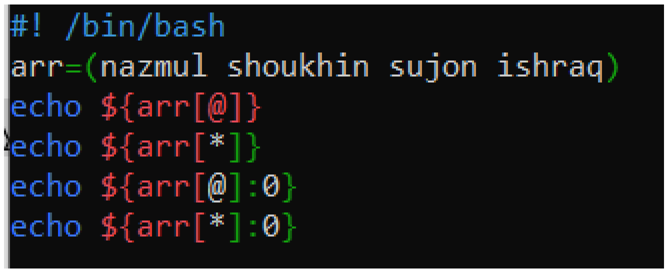
## ❖ To print all the value of an array:

```
echo ${ARRAYNAME[*]}
```

```
[@] & [*] means All elements of Array.
```

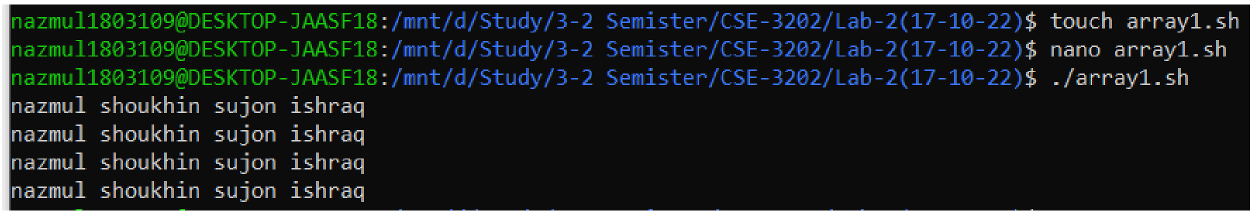
Run this program:

## Code:



```
#! /bin/bash
arr=(nazmul shoukhin sujon ishraq)
echo ${arr[@]}
echo ${arr[*]}
echo ${arr[@]:0}
echo ${arr[*]:0}
```

**Output:**



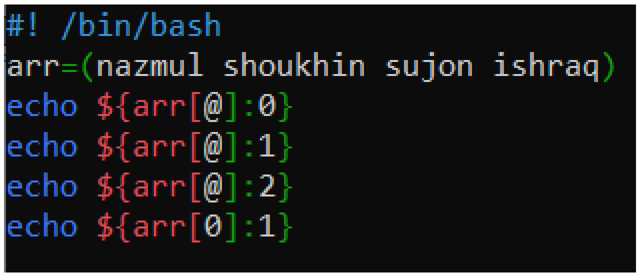
```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch array1.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano array1.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./array1.sh
nazmul shoukhin sujon ishraq
nazmul shoukhin sujon ishraq
nazmul shoukhin sujon ishraq
nazmul shoukhin sujon ishraq
```

## ❖ To print elements from a particular index

```
echo ${ARRAYNAME[WHICH_ELEMENT]:STARTING_INDEX}
```

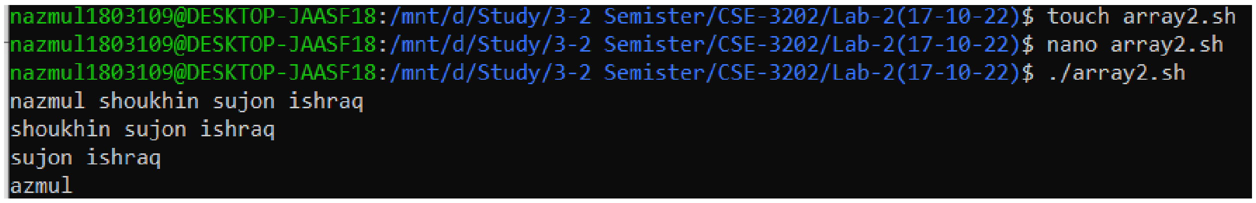
Run the following code:

**Code:**



```
#! /bin/bash
arr=(nazmul shoukhin sujon ishraq)
echo ${arr[@]:0}
echo ${arr[@]:1}
echo ${arr[@]:2}
echo ${arr[0]:1}
```

Output:



```
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ touch array2.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ nano array2.sh
nazmul1803109@DESKTOP-JAASF18:/mnt/d/Study/3-2 Semister/CSE-3202/Lab-2(17-10-22)$ ./array2.sh
nazmul shoukhin sujon ishraq
shoukhin sujon ishraq
sujon ishraq
azmul
```

**Discussion:**

All of the above code run successfully.But I faced a problem for using syntax, missing space is the main caused for an error. In shell coding for loop is almost similar like other high level language(such as c,c++,python). We can relate shell coding with other high level language partially.