2 enter the value .

3. accept a number, check whether the number is even or odd.

```
gokilagopal@DESKTOP-LQA68O9:~$ nano comparefile.txt
gokilagopal@DESKTOP-LQA68O9:~$ bash comparefile.txt
 enter a number :
7
the given number is odd
gokilagopal@DESKTOP-LQA68O9:~$ cat comparefile.txt
#!/bin/bash
echo " enter a number :"
read num
if((num%2==0));
then
echo "the given number is even "
else
echo "the given number is odd "
fi
```

4. using if

Check whether a file exist and it is regular file or directory.

```
gokilagopal@DESKTOP-LQA68O9:~$ nano checkingFile.txt
gokilagopal@DESKTOP-LQA68O9:~$ bash checkingFile.txt
enter the file name:
comparefile.txt
checkingFile.txt: line 4: if[ -e comparefile.txt ]: command not found
checkingFile.txt: line 5: syntax error near unexpected token `then'
checkingFile.txt: line 5: `                then'
gokilagopal@DESKTOP-LQA68O9:~$ nano checkingFile.txt
gokilagopal@DESKTOP-LQA68O9:~$ bash checkingFile.txt
Enter the file name:
calculation.sh
File exists
It is a regular file
gokilagopal@DESKTOP-LQA68O9:~$ bash checkingFile.txt
Enter the file name:
samplefile.txt
File does not exist
```

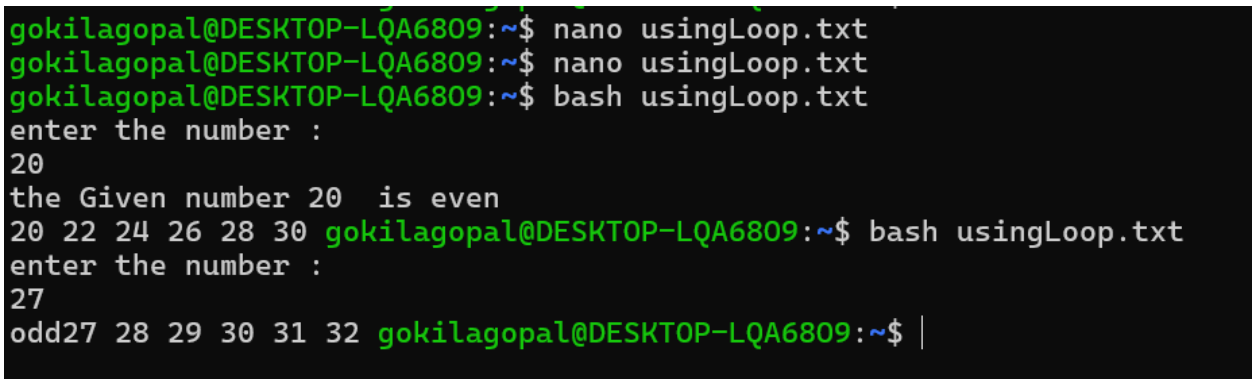
5. using for loop

Accept a num: 100

Even: 100 102 104 106 108

Accept a num: 101

Odd: 101 103 105 107 109



6.Folder name as input. get all files and sub-folder names from the given folder and print the same

7.arithmetic operation using switch case

8 example of a function

```
gokilagopal@DESKTOP-LQA68O9:~$ vi Function.sh
gokilagopal@DESKTOP-LQA68O9:~$ ./Function.sh
-bash: ./Function.sh: Permission denied
gokilagopal@DESKTOP-LQA68O9:~$ bash Function.sh
Mon Dec 15 06:21:12 UTC 2025
ArithmeticFile.txt  Function.sh        comparefile.txt  filetest.pub   practiceTest
  testOne
Disk                Myscript.sh        contentfile.txt  folderchk.txt  pratice.txt
  testfile.txt
ExampleFile         bashFile.txt.save  fie2             myscript.sh    pub
  usingLoop.txt
File1               calculation.sh     file1.txt        notesTest.txt  pub.pub
File1.pub           checkingFile.txt   filetest         practice.txt   samplefile
example of a function fun()
gokilagopal@DESKTOP-LQA68O9:~$ cat Function.sh
#!/bin/bash
funone()
{
        date
        ls
        echo "example of a function fun()"
}
funone
```

9. function program

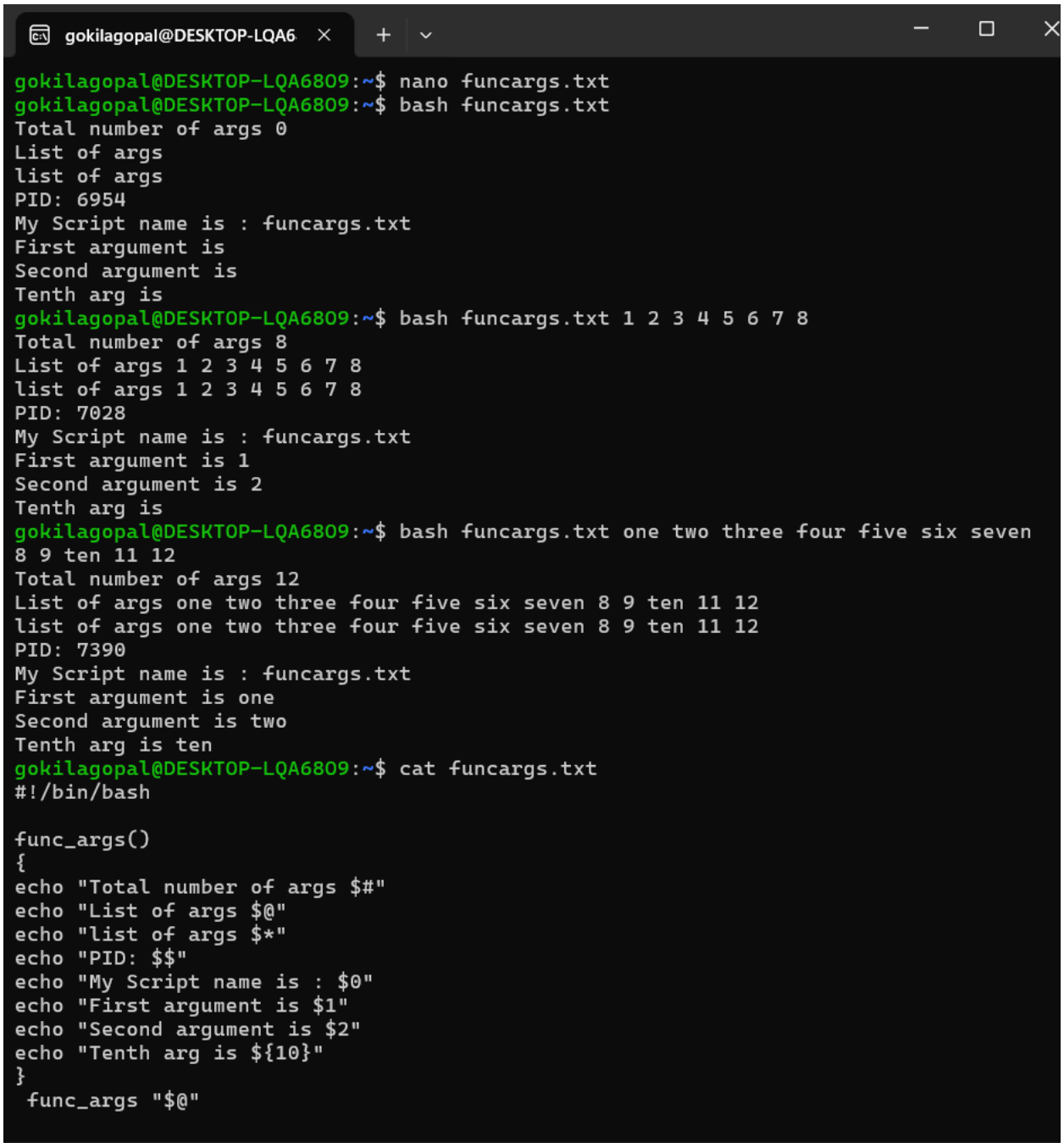
```
gokilagopal@DESKTOP-LQA68O9:~$ nano creatingfunction.t
gokilagopal@DESKTOP-LQA68O9:~$ bash creatingfunction.t
enter number1:
4
enter number2:
5
creatingfunction.txt: line 5: 9: command not found
 the sum of two numbers is:
gokilagopal@DESKTOP-LQA68O9:~$ cat creatingfunction.tx
#!/bin/bash

addFunt()
{
sum= $((num1+num2))
echo " the sum of two numbers is: $sum"
}

echo "enter number1: "
read num1
echo "enter number2: "
read num2

addFunt $num1 $num2
```

10 . command line arguments

```
Tenth arg is 0
gokilagopal@DESKTOP-LQA68O9:~$ bash argsfile.sh 1 2 3 4 5 6 7 8 9 ten eleven twelve 13 14 1
5
Total number of args 15
List of args 1 2 3 4 5 6 7 8 9 ten eleven twelve 13 14 15
List of args 1 2 3 4 5 6 7 8 9 ten eleven twelve 13 14 15
PID: 1381
My Script name is : argsfile.sh
first argument is 1
Second argument is 2
Tenth arg is ten
gokilagopal@DESKTOP-LQA68O9:~$ cat argsfile.sh
#!/bin/bash

echo "Total number of args $#"

echo "List of args $@"

echo "List of args $*"

echo "PID: $$"

echo "My Script name is : $0"

echo "first argument is $1"

echo "Second argument is $2"

echo "Tenth arg is ${10}"

gokilagopal@DESKTOP-LQA68O9:~$
```

Symbol	Meaning
`$#`	Total number of arguments
`$@`	All arguments (recommended)
`$*`	All arguments (single string)
`$0`	Script name
`$1`..`${10}`	Individual arguments
`"$@"`	Pass all args safely to function

11. using function for CLA


gokilagopal@DESKTOP-LQA6
gokilagopal@DESKTOP-LQA68O9:~$ nano funcargs.txt
gokilagopal@DESKTOP-LQA68O9:~$ bash funcargs.txt
Total number of args 0
List of args
list of args
PID: 6954
My Script name is : funcargs.txt
First argument is
Second argument is
Tenth arg is
gokilagopal@DESKTOP-LQA68O9:~$ bash funcargs.txt 1 2 3 4 5 6 7 8
Total number of args 8
List of args 1 2 3 4 5 6 7 8
list of args 1 2 3 4 5 6 7 8
PID: 7028
My Script name is : funcargs.txt
First argument is 1
Second argument is 2
Tenth arg is
gokilagopal@DESKTOP-LQA68O9:~$ bash funcargs.txt one two three four five six seven
8 9 ten 11 12
Total number of args 12
List of args one two three four five six seven 8 9 ten 11 12
list of args one two three four five six seven 8 9 ten 11 12
PID: 7390
My Script name is : funcargs.txt
First argument is one
Second argument is two
Tenth arg is ten
gokilagopal@DESKTOP-LQA68O9:~$ cat funcargs.txt
#!/bin/bash
func_args()
{
echo "Total number of args $#"
echo "List of args $@"
echo "list of args $*"
echo "PID: $$"
echo "My Script name is : $0"
echo "First argument is $1"
echo "Second argument is $2"
echo "Tenth arg is ${10}"
}
func_args "$@"

12 Arrays

```
gokilagopal@DESKTOP-LQA68O9:~$ nano arrtwo.sh
gokilagopal@DESKTOP-LQA68O9:~$ bash arrtwo.sh
first : aaple
second : orangeberry
third : cherry
 ------------------
all : aaple orangeberry cherry
all : aaple orangeberry cherry
 ----------------------------------
all aaple orangeberry
all: orangeberry cherry
gokilagopal@DESKTOP-LQA68O9:~$ cat arrtwo.sh
#!/bin/bash
fruits=(aaple orangeberry cherry)
echo "first : ${fruits[0]}"
echo "second : ${fruits[1]}"
echo "third : ${fruits[2]}"
echo " ------------------"
echo "all : ${fruits[@]}"
echo "all : ${fruits[*]}" #all the elements as single string
echo " ----------------------------------"
echo "all ${fruits[@]:0:2}"
echo "all: ${fruits[*]:1:2}" #slicing
```

13 Associate Array

```
gokilagopal@DESKTOP-LQA68O9:~$ nano assarr.txt
gokilagopal@DESKTOP-LQA68O9:~$ bash assarr.txt
Length: 4
shell ---> bash
id ---> 1001
path ---> /home/gokilagopal
name ---> GokilaGopal
gokilagopal@DESKTOP-LQA68O9:~$ cat assarr.txt
#!/bin/bash
#introduced from bash 4+ onwards

declare -A user=(
    [name]="GokilaGopal"
    [id]=1001
    [path]="/home/gokilagopal"
    [shell]="bash"
)

echo "Length: ${#user[@]}"

for key in  ${!user[@]} ; do
    echo "$key ---> ${user[$key]}"
done
gokilagopal@DESKTOP-LQA68O9:~$
```

14. redirection of input and Output

```
gokilagopal@DESKTOP-LQA68O9:~$ text=$arrtwo.sh
gokilagopal@DESKTOP-LQA68O9:~$ bash inputone.sh
#!/bin/bash
fruits=(aaple orangeberry cherry)
echo "first : ${fruits[0]}"
echo "second : ${fruits[1]}"
echo "third : ${fruits[2]}"
echo " ------------------"
echo "all : ${fruits[@]}"
echo "all : ${fruits[*]}" #all the elements as single string
echo " ----------------------------------"
echo "all ${fruits[@]:0:2}"
echo "all: ${fruits[*]:1:2}" #slicing

gokilagopal@DESKTOP-LQA68O9:~$ cat inputone.sh
#!/bin/bash

while read -r lineText ; do
        echo $lineText
done < arrtwo.sh
```

15. Mapfile is also known as read array .. which reads a file line by line …stores each line into an array here ,the array name is *lines*

```
gokilagopal@DESKTOP-LQA68O9:~$ echo "${#lines[@]}"
12
gokilagopal@DESKTOP-LQA68O9:~$ mapfile -t lines <arrtwo.sh
gokilagopal@DESKTOP-LQA68O9:~$ echo "${#lines[@]}"
12
gokilagopal@DESKTOP-LQA68O9:~$
```