1. **Student Details**

```
#!/bin/sh
a=0
echo "Enter the database name:"
read fname
while [ $a -lt 3 ]
do
        echo "Enter student id:"
        read sid
        echo "Student name:"
        read sname
        echo "Enter sem:"
        read sem
        echo "Department:"
        read dept
        a=`expr $a + 1`
        echo "$sid $sname $sem $dept" >> $fname
done
echo Student details
sort $fname | uniq
echo Dept with stud count
cut -d' ' -f4 $fname | sort | uniq -c
```

```
$ sh pgm1.sh
Enter the database name:
MSRIT
Enter student id:
11
Student name:
ABC
Enter sem:
3
Department:
ISE
Enter student id:
12
Student name:
DEF
Enter sem:
3
Department:
ISE
Enter student id:
14
Student name:
XYZ
Enter sem:
5
Department:
EEE
Student details
11 ABC 3 ISE
12 DEF 3 ISE
14 XYZ 5 EEE
Dept with stud count
      1 EEE
      2 ISE
```

2. **Check if 2 file contents are same**

```
echo "Enter name of first file:"
read file1
echo "Enter name of second file:"
read file2
cmp -s "$file1" "$file2"
if [ $? -eq 0 ]
then
        echo "Both files are same"
        rm "$file2"
else
        echo "Both files are not same"
fi
```

```
rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ cat > ss
hello


rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ cat > dd
hello


rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm2.sh
Enter name of first file:
ss
Enter name of second file:
dd
Both files are same

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm2.sh
Enter name of first file:
pgm2.sh
Enter name of second file:
pgm1.sh
Both files are not same
```

3. **Check if a file is a directory, a file or something else**

```
a=`ls -l | grep "$1" | cut -b1`
if [ "$a" = "d" ]
then
        echo "File is a directory"
elif [ "$a" = "-" ]
then
        echo "File is a regualr file"
else
        echo "File is something else"
fi
```

```
rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ ls
MSRIT        pgm10.sh*  pgm2.sh  pgm5.sh  pgm9.sh
NewFolder/   pgm11.sh   pgm3.sh  pgm6.sh  ss
pgm1.sh*     pgm12.sh   pgm4.sh  pgm7.sh  svr

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm3.sh pgm2.sh
File is a regualr file

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm3.sh NewFolder
File is a directory

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm3.sh xyz
File is something else
```

4. **File name to uppercase**

```
echo "Directory contents before:"
ls
for src in "$@"
do
        dest=`echo "$src" | tr [a-z] [A-Z]`
        mv "$src" "$dest"
done
echo "Directory contents after:"
ls
~
~
```

```
rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ touch example.sh index.html

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm4.sh example.sh index.html
Directory contents before:
MSRIT       index.html  pgm11.sh  pgm3.sh  pgm6.sh  ss
NewFolder   pgm1.sh     pgm12.sh  pgm4.sh  pgm7.sh  svr
example.sh  pgm10.sh    pgm2.sh   pgm5.sh  pgm9.sh
Directory contents after:
EXAMPLE.SH  NewFolder  pgm11.sh  pgm3.sh  pgm6.sh  ss
INDEX.HTML  pgm1.sh    pgm12.sh  pgm4.sh  pgm7.sh  svr
MSRIT       pgm10.sh   pgm2.sh   pgm5.sh  pgm9.sh
```

Just check example.sh and index.html

5. **Check and compare permissions of two files**

```
p1=`ls -l $1 | cut -c 2-10`
p2=`ls -l $2 | cut -c 2-10`
if test $p1 = $p2
then
        echo "Permissions are equal they are: $p1"
else
        echo "Permissions are not equal"
        echo "$1 permissions are $p1"
        echo "$2 permissions are $p2"
fi
~
```

```
rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ vim pgm5.sh

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ touch eg1.sh

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ touch eg2.sh

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ chmod 555 eg1.sh

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ chmod 777 eg2.sh

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm5.sh eg1.sh eg2.sh
Permissions are not equal
eg1.sh permissions are r--r--r--
eg2.sh permissions are rw-r--r--

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm5.sh pgm4.sh pgm5.sh
Permissions are equal they are: rw-r--r--
```

6. **Program on string**

```
echo Enter str1:
read str1
echo Enter str2:
read str2
if test -z $str1
then
        echo str1 is null string
else
        echo str1 is $str1
fi
if test -z $str2
then
        echo str2 is null string
else
        echo str2 is $str2
fi
echo lenght of $str1 is ${#str1}
echo length of $str2 is ${#str2}
if test $str1 = $str2
then
        echo Strings are equal, string is $str1
else
        echo Strings are not equal
fi
```

```
rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ vim pgm6.sh

rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm6.sh
Enter str1:
abcde
Enter str2:
xyz
str1 is abcde
str2 is xyz
lenght of abcde is 5
length of xyz is 3
Strings are not equal
```

7. **Print arguments in reverse order**

```
echo "Given argument is: $*"
rev=""
for var in $*
do
        rev=$var" "$rev
done
echo "Argument in reversed order is: $rev"
```

```
$ sh pgm8.sh This is an example arguement
Given argument is: This is an example arguement
Argument in reversed order is: arguement example an is This
```

8. Print number in reverse order

```
echo Enter the number:
read num
rev=0
while [ $num -gt 0 ]
do
        rem=`expr $num % 10`
        rev=`expr $rev \* 10 + $rem`
        num=`expr $num / 10`
done
echo Reverse of the number is: $rev
~
~
```

```
Enter the number:
12345
Reverse of the number is: 54321
```

9. First 25 Fibonacci numbers

```
fib0=0
fib1=1
num=2
echo First 25 fibonacci number are $fib0 $fib1
while test $num -lt 25
do
        fib2=`expr $fib0 + $fib1`
        echo $fib2
        fib0=$fib1
        fib1=$fib2
        num=`expr $num + 1`
done
```

```
rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm9.sh
First 25 fibonacci number are 0 1
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
6765
10946
17711
28657
46368
```

10. Prime numbers in given range

```
echo "Enter range:"
read num1
read num2

echo "Prime numbers are:"

while [ $num1 -le $num2 ]
do
    prime=1
    i=2
    if [ $num1 -gt 1 ]
    then
        while [ $i -le $(expr $num1 / 2) ]
        do
            if [ $(expr $num1 % $i) -eq 0 ]
            then
                prime=0
                break
            fi
            i=$(expr $i + 1)
        done

        if [ $prime -eq 1 ]
        then
            echo $num1
        fi
    fi
    num1=$(expr $num1 + 1)
done
```

```
rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm10.sh
Enter range:
0
50
Prime numbers are:
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
```

11. Linear search algo

```
echo Enter number of elements:
read n
echo Enter elements:
for (( i=0; i < n ; i++ ))
do
        read arr[i]
done
echo Enter the key to search:
read key
found=0
for (( i=0 ; i < n ; i++))
do
        if [ ${arr[i]} -eq $key ]
        then
                echo "Key $key found at position $((i+1)) (Index: $i)"
                found=1
                break
        fi
done
if [ $found -eq 0 ];
then
        echo Key not found
fi
```

```
rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm11.sh
Enter number of elements:
5
Enter elements:
11
43
29
37
86
Enter the key to search:
37
Key 37 found at position 4 (Index: 3)
```

12. Find largest among given three

```
max3(){
        echo Given numbers are $1 $2 $3
        lar=$1
        if [ $1 -lt $2 ]
        then
                lar=$2
        fi
        if [ $lar -lt $3 ]
        then
                lar=$3
        fi
        return $lar
}
max3 23 45 34
echo Largest number is $?
```

```
rosha@Roshan-TUF-F15 MINGW64 ~/ShellScript
$ sh pgm12.sh
Given numbers are 23 45 34
Largest number is 45
```