Q1 Number classification

```
check_num() {
        if [$num -gt 0]; then
                echo "Positive"
        else if [$num -lt 0]; then
                echo "Negative"
        else
                echo "Zero"
        fi
}
read -p "Enter num: " num
check_num $num
```

```
RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q1.sh
Enter num: 10
Positive

RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q1.sh
Enter num: -10
Negative

RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q1.sh
Enter num: 0
Zero
```

Q2 Factorial of a number

```
read -p "Enter num: " num
fact=1

while [ $num -gt 0 ]; do
        fact=$((fact * num))
        num=$((num - 1))
done

echo "Factorial: $fact"
```

```
RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q2.sh
Enter num: 3
Factorial: 6

RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q2.sh
Enter num: 5
Factorial: 120
```

Q3 Sum of N numbers

```
sum() {
        sum=0
        for ((i=0; i<=$1; i++)); do
                sum=$((sum + i))
        done
        echo "Sum: $sum"
}
read -p "Enter num: " num
sum $num
```

```
RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q3.sh
Enter num: 10
Sum: 55
```

Q4 Prime Number

```
check_prime() {
        flag=0
        for ((i=2; i<=$1/2; i++)); do
                if  (($1 % i == 0)); then
                        flag=1
                        break
                fi
        done

        if [ $flag == 0 ]; then
                echo "Prime"
        else
                echo "Not Prime"
        fi
}

read -p "Enter num: " num
check_prime $num
```

```
RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q4.sh
Enter num: 5
Prime

RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q4.sh
Enter num: 10
Not Prime
```

Q5 Reverse a number

```
read -p "Enter num: " num

rev=0
while (($num > 0)); do
        digit=$((num % 10))
        rev=$((rev * 10 + digit))
        num=$((num / 10))
done

echo "Reverse: $rev"
```

```
RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q5.sh
Enter num: 12345
Reverse: 54321
```

Q6 Check file existence and permission

```
check_file() {
        permissions=()
        if [ -e "$1" ]; then
                echo "File $1 exists !!!"
                [ -r "$1" ] && permissions+=("Read")
                [ -w "$1" ] && permissions+=("Write")
                [ -x "$1" ] && permissions+=("Execute")
                echo "Permissions: ${permissions[@]}"
        else
                echo "File does not exists !!!"
        fi
}

read -p "Enter filename: " filename
check_file "$filename"
```

```
RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q6.sh
Enter filename: q1.sh
File q1.sh exists !!!
Permissions: Read Write

RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q6.sh
Enter filename: nofile
File does not exists !!!
```

Q7 Menu-driven calculator

```
operation() {
        [ "$1" == "+" ] && result=$(($2 + $3))
        [ "$1" == "-" ] && result=$(($2 - $3))
        [ "$1" == "*" ] && result=$(($2 * $3))
        [ "$1" == "/" ] && result=$(awk "BEGIN{print $2 / $3}")
        echo "Result: $result"
}
while [ 1 ]; do
        echo "****************calculator*****************"
        read -p "Choose operation(+,-,*,/)(x => exit): " op

        if [[ "$op" == 'x' ]]; then
                echo "Exiting..."
                break
        fi

        if [[ "$op" == '+' || "$op" == '-' || "$op" == '*' || "$op" ==
'/' ]]; then
                read -p "Enter num1: " n1
                read -p "Enter num2: " n2
                operation "$op" $n1 $n2
        else
                echo "Invalid ! Enter valid input"
        fi
done
```

```
RAPHEL M L@RNLD MINGW64 ~/Desktop/SHELL
$ ./q7.sh
****************calculator*****************
Choose operation(+,-,*,/)(x => exit): +
Enter num1: 5
Enter num2: 15
Result: 20
****************calculator*****************
Choose operation(+,-,*,/)(x => exit): -
Enter num1: 10
Enter num2: 5
Result: 5
****************calculator*****************
Choose operation(+,-,*,/)(x => exit): *
Enter num1: 5
Enter num2: 3
Result: 15
****************calculator*****************
Choose operation(+,-,*,/)(x => exit): /
Enter num1: 5
Enter num2: 2
Result: 2.5
****************calculator*****************
Choose operation(+,-,*,/)(x => exit): x
Exiting...
```