1)Write a python program to create a function called collatz() which reads as parameter named number. If the number is even it should print and return number//2 and if the number is odd then it should print and return 3*number+1. The function should keep calling on that number until the function returns a value 1.

```
def collatz(number):

    if number % 2 == 0:        #%return the remainder,/ devision will return the quotient

        print(number // 2)     #floor division // rounds the result down to the nearest whole number
        return number // 2

    elif number % 2 == 1:
        result = 3 * number + 1
        print(result)
        return result

n = input("Give me a number: ")
while n != 1:
    n = collatz(int(n))
```

output:

```
Give me a number: 3
10
5
16
8
4
2
1
```

2)define a Python function with suitable parameters to generate prime numbers between two integer values. Write a Python program which accepts two integer values m and n (note: m>0, n>0 and m < n) as inputs and passthesevalues to the function. Suitable error messages should be displayed if the conditions for input values are not followed

```
def kts_prime():
    try:
        n = int(input("Please, Enter the n Value: "))
        m = int(input("Please, Enter the m Value: "))
        if (n > 0 and m > 0):
            for num in range(n, m + 1):
                # all prime numbers are greater than 1
                if num > 1:
                    for i in range(2, num):
                        if (num % i) == 0:
                            break
```

```
            else:
                print(num)
    except Exception:
            pass
    finally:
        print("successfull")
kts_prime()
```

3.Python program to check given number is positive or negative (if-else).

```
num = float(input("Input a number: "))
if num > 0:
        print("It is positive number")
elif num == 0:
        print("It is Zero")
else:
        print("It is a negative number")
```

4.Python program to check given number is even or odd (if- else). num = int (input ("Enter any number to test whether it is odd or even: “)

```
if (num % 2) == 0:
        print (“The number is even”)
else:
        print (“The provided number is odd”)
```

Output:

Enter any number to test whether it is odd or even:

887

887 is odd.

5.Python program to check the person is Eligible to vote or not (if-else).

```
Age = int(input(“enter your age”)
If age == 18:
        Print(“eligible”)
Else:
```

Print(“not eligible”)

6.Python program to check largest of three numbers.(if-elif- else).

```
a = int(input('Enter first number  : '))
b = int(input('Enter second number : '))
c = int(input('Enter third number  : '))

if a > b and a > c :
   print(“largest of three is a”)
elif b > c :
   print(“largest of three is b”)
else :
   print(“largest of three is c”)
```

7. Python program to checm=int(input("Enter your mark "))

```
if(m>=80):
   print("You got A grade")
elif(m>=60):
   print("You got B grade")
elif(m>=40):
   print("You got C grade")
else:
   print("Failed in this Exam ")
```

8.python program for addition,subtraction,multiplication

```
def addition(x,y):
        return(x+y)
addition(4,5):


def subtraction(x,y):
        return(x-y)
subtraction(19,8):


def multiplication(x,y):
        return(x*y)
multiplication(3,7)
```

9.python program for exception handling,devide by zero

```
try:
    num1 = int(input("Enter First Number: "))
    num2 = int(input("Enter Second Number: "))
    result = num1 / num2
    print(result)
except ValueError as e:
    print("Invalid Input Please Input Integer...")
except ZeroDivisionError as e:
    print(e)
```

10. **Program to illustrate handling of type exception in Python**

```
while True:
    try:
        num = int(input("Enter First Number: "))
        print(num)
        break
    except ValueError as e:
        print("Invalid Input..Please Input Integer Only..")
```

11.break statement in python

```
# Use of break statement inside the loop

for val in "string":
    if val == "i":
        break
    print(val)

print("The end")
```

OUTPUT:

```
s
t
r
The end
```

12. Python continue statement

The `continue` statement is used to skip the rest of the code inside a loop for the current iteration only. Loop does not terminate but continues on with the next iteration.

```
# Program to show the use of continue statement inside loops

for val in "string":
    if val == "i":
        continue
    print(val)

print("The end")
```

Output

```
s
t
r
n
g
The end
```