Assignment 1

August 15, 2021

[36]:
```
# 1(a): Summing first N natural numbers

%run MyLib.ipynb # Running my library

n=input("Enter a natural number : ")   # taking input

n=check_natural_number(n)  # Checking for validity

if n!='F':
    sum_natural_numbers(n)
```

```
Enter a natural number : 2
The sum of first 2 natural numbers is 3
```

[38]:
```
%run MyLib.ipynb # Running my library

n=input("Enter a natural number : ")   # taking input

n=check_natural_number(n) # Checking for validity

if n!='F':
    sum_natural_numbers(n)
```

```
Enter a natural number : -1
Invalid input. Please enter a natural number.
```

[39]:
```
%run MyLib.ipynb # Running my library

n=input("Enter a natural number : ")  # taking input

n=check_natural_number(n)  # Checking for validity

if n!='F':
    sum_natural_numbers(n)
```

```
Enter a natural number : 5
The sum of first 5 natural numbers is 15
```

[40]:
```
%run MyLib.ipynb # Running my library

n=input("Enter a natural number : ")  # taking input

n=check_natural_number(n)  # Checking for validity

if n!='F':
    sum_natural_numbers(n)
```

```
Enter a natural number : 9
The sum of first 9 natural numbers is 45
```

[41]:
```
%run MyLib.ipynb # Running my library

n=input("Enter a natural number : ")  # taking input

n=check_natural_number(n)  # Checking for validity

if n!='F':
    sum_natural_numbers(n)
```

```
Enter a natural number : 0
Invalid input. Please enter a natural number.
```

[37]:
```
# 1(b): Summing first N odd natural numbers

%run MyLib.ipynb # Running my library

n=input("Enter a natural number : ") # taking input


n=check_natural_number(n) # Checking for validity

if n!='F':
    sum_odd_numbers(n)
```

```
Enter a natural number : 2
The sum of first 2 odd numbers is 4
```

[42]:
```
%run MyLib.ipynb # Running my library

n=input("Enter a natural number : ") # taking input


n=check_natural_number(n) # Checking for validity

if n!='F':
    sum_odd_numbers(n)
```

```
Enter a natural number : 4
The sum of first 4 odd numbers is 16
```

[43]:
```
%run MyLib.ipynb # Running my library

n=input("Enter a natural number : ") # taking input


n=check_natural_number(n) # Checking for validity

if n!='F':
    sum_odd_numbers(n)
```

```
Enter a natural number : -3
Invalid input. Please enter a natural number.
```

[44]:
```
%run MyLib.ipynb # Running my library

n=input("Enter a natural number : ") # taking input


n=check_natural_number(n) # Checking for validity

if n!='F':
    sum_odd_numbers(n)
```

```
Enter a natural number : 12
The sum of first 12 odd numbers is 144
```

[45]:
```
%run MyLib.ipynb # Running my library

n=input("Enter a natural number : ") # taking input


n=check_natural_number(n) # Checking for validity

if n!='F':
    sum_odd_numbers(n)
```

```
Enter a natural number : 0
Invalid input. Please enter a natural number.
```

[4]:
```
num = int(input())

factorial = 1

if num < 0:   #if num is negative
   print("Does not exist")
```

```
elif num == 0:   #if num is zero
   print("The factorial of 0 is 1")

else:   #if num is positive
   for i in range(1,num + 1):   #iterate the given num using for loop
       factorial = factorial*i    # multiply with the previous result
   print("THE FACTORIAL OF",num,"IS",factorial) #print
```

```
3
THE FACTORIAL OF 3 IS 6
```

[5]:
```
num = int(input())

factorial = 1

if num < 0:   #if num is negative
   print("Does not exist")

elif num == 0:   #if num is zero
   print("The factorial of 0 is 1")

else:   #if num is positive
   for i in range(1,num + 1):   #iterate the given num using for loop
       factorial = factorial*i    # multiply with the previous result
   print("THE FACTORIAL OF",num,"IS",factorial) #print
```

```
-4
Does not exist
```

[6]:
```
num = int(input())

factorial = 1

if num < 0:   #if num is negative
   print("Does not exist")

elif num == 0:   #if num is zero
   print("The factorial of 0 is 1")

else:   #if num is positive
   for i in range(1,num + 1):   #iterate the given num using for loop
       factorial = factorial*i    # multiply with the previous result
   print("THE FACTORIAL OF",num,"IS",factorial) #print
```

```
8
THE FACTORIAL OF 8 IS 40320
```

[7]:
```
num = int(input())

factorial = 1

if num < 0:   #if num is negative
   print("Does not exist")

elif num == 0:   #if num is zero
   print("The factorial of 0 is 1")

else:   #if num is positive
   for i in range(1,num + 1):   #iterate the given num using for loop
       factorial = factorial*i    # multiply with the previous result
   print("THE FACTORIAL OF",num,"IS",factorial) #print
```

```
-4
Does not exist
```

[8]:
```
num = int(input())

factorial = 1

if num < 0:   #if num is negative
   print("Does not exist")

elif num == 0:   #if num is zero
   print("The factorial of 0 is 1")

else:   #if num is positive
   for i in range(1,num + 1):   #iterate the given num using for loop
       factorial = factorial*i    # multiply with the previous result
   print("THE FACTORIAL OF",num,"IS",factorial) #print
```

```
0
The factorial of 0 is 1
```

[13]:
```
# sum of N terms of an A.P starting with 'a'
def sumOfAP( a, d,n) :
    sum = 0   # setting initial sum to zero
    i = 0 # introducing the variable
    if n > 0:
        while i < n :
            sum = sum + a  # adding sum by the common difference
            a = a + d    # increasing the common difference each time by a sum␣
 ↪of 'd' in the loop
            i = i + 1    # increasing the variable
        return sum
```

```
    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this A.P Series: : "))
a = float(input("Please Enter First Number of an A.P Series: : "))
d = 1.5
print("The sum of Arithmetic Progression starting with " + str(a) + " having " ␣
 ↪+ str(n) +  " numbers is : ", end = " ")
print (sumOfAP(a, d, n))      # printing the output
```

```
Please Enter the Total Numbers in this A.P Series: : 2
Please Enter First Number of an A.P Series: : 6
The sum of Arithmetic Progression starting with 6.0 having 2 numbers is :  13.5
```

[14]:
```
# sum of N terms of an A.P starting with 'a'
def sumOfAP( a, d,n) :
    sum = 0   # setting initial sum to zero
    i = 0 # introducing the variable
    if n > 0:
        while i < n :
            sum = sum + a  # adding sum by the common difference
            a = a + d    # increasing the common difference each time by a sum␣
 ↪of 'd' in the loop
            i = i + 1    # increasing the variable
        return sum
    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this A.P Series: : "))
a = float(input("Please Enter First Number of an A.P Series: : "))
d = 1.5
print("The sum of Arithmetic Progression starting with " + str(a) + " having " ␣
 ↪+ str(n) +  " numbers is : ", end = " ")
print (sumOfAP(a, d, n))      # printing the output
```

```
Please Enter the Total Numbers in this A.P Series: : -2
Please Enter First Number of an A.P Series: : 4
The sum of Arithmetic Progression starting with 4.0 having -2 numbers is :
Enter a positive integer for 'n'
None
```

[15]:
```
# sum of N terms of an A.P starting with 'a'
def sumOfAP( a, d,n) :
```

```
    sum = 0   # setting initial sum to zero
    i = 0 # introducing the variable
    if n > 0:
        while i < n :
            sum = sum + a  # adding sum by the common difference
            a = a + d    # increasing the common difference each time by a sum␣
 ↪of 'd' in the loop
            i = i + 1    # increasing the variable
        return sum
    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this A.P Series: : "))
a = float(input("Please Enter First Number of an A.P Series: : "))
d = 1.5
print("The sum of Arithmetic Progression starting with " + str(a) + " having " ␣
 ↪+ str(n) +  " numbers is : ", end = " ")
print (sumOfAP(a, d, n))      # printing the output
```

```
Please Enter the Total Numbers in this A.P Series: : 5
Please Enter First Number of an A.P Series: : 3
The sum of Arithmetic Progression starting with 3.0 having 5 numbers is :  30.0
```

[16]:
```
# sum of N terms of an A.P starting with 'a'
def sumOfAP( a, d,n) :
    sum = 0   # setting initial sum to zero
    i = 0 # introducing the variable
    if n > 0:
        while i < n :
            sum = sum + a  # adding sum by the common difference
            a = a + d    # increasing the common difference each time by a sum␣
 ↪of 'd' in the loop
            i = i + 1    # increasing the variable
        return sum
    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this A.P Series: : "))
a = float(input("Please Enter First Number of an A.P Series: : "))
d = 1.5
print("The sum of Arithmetic Progression starting with " + str(a) + " having " ␣
 ↪+ str(n) +  " numbers is : ", end = " ")
print (sumOfAP(a, d, n))      # printing the output
```

```
Please Enter the Total Numbers in this A.P Series: : 4
Please Enter First Number of an A.P Series: : 0.4
The sum of Arithmetic Progression starting with 0.4 having 4 numbers is :  10.6
```

[19]:
```
# sum of N terms of an A.P starting with 'a'
def sumOfAP( a, d,n) :
    sum = 0   # setting initial sum to zero
    i = 0 # introducing the variable
    if n > 0:
        while i < n :
            sum = sum + a  # adding sum by the common difference
            a = a + d    # increasing the common difference each time by a sum␣
 ↪of 'd' in the loop
            i = i + 1    # increasing the variable
        return sum
    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this A.P Series: : "))
a = float(input("Please Enter First Number of an A.P Series: : "))
d = 1.5
print("The sum of Arithmetic Progression starting with " + str(a) + " having " ␣
 ↪+ str(n) +  " numbers is : ", end = " ")
print (sumOfAP(a, d, n))      # printing the output
```

```
Please Enter the Total Numbers in this A.P Series: : 3
Please Enter First Number of an A.P Series: : 8
The sum of Arithmetic Progression starting with 8.0 having 3 numbers is :  28.5
```

[2]:
```
# sum of N terms of an G.P starting with 'a'
def sumOfGP( a, r,n) :
    if n > 0:
        sum = 0   # setting initial sum to zero
        i = 0     # introducing the variable
        while i < n :
            sum = sum + a  # adding sum by the common ratio
            a = r*a    # increasing the common ratio each time by a muliple of␣
 ↪'r' in the loop
            i = i + 1    # increasing the variable
        return sum


    elif n <= 0:
        print("Enter a positive integer for 'n' ")
```

```
# taking inputs
n = int(input("Please Enter the Total Numbers in this G.P Series: : "))
a = float(input("Please Enter First Number of an G.P Series: : "))
r = 0.5
print("The sum of Geometric Progression starting with " + str(a) + " having "  +␣
 ↪str(n) +  " numbers is : ", end = " ")
print (sumOfGP(a, r, n))      # printing the output
```

```
Please Enter the Total Numbers in this G.P Series: : 2
Please Enter First Number of an G.P Series: : 6
The sum of Geometric Progression starting with 6.0 having 2 numbers is :  9.0
```

[18]:
```
# sum of N terms of an G.P starting with 'a'
def sumOfGP( a, r,n) :
    if n > 0:
        sum = 0   # setting initial sum to zero
        i = 0     # introducing the variable
        while i < n :
            sum = sum + a  # adding sum by the common ratio
            a = r*a    # increasing the common ratio each time by a muliple of␣
 ↪'r' in the loop
            i = i + 1    # increasing the variable
        return sum


    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this G.P Series: : "))
a = float(input("Please Enter First Number of an G.P Series: : "))
r = 0.5
print("The sum of Geometric Progression starting with " + str(a) + " having "  +␣
 ↪str(n) +  " numbers is : ", end = " ")
print (sumOfGP(a, r, n))      # printing the output
```

```
Please Enter the Total Numbers in this G.P Series: : 4
Please Enter First Number of an G.P Series: : 6
The sum of Geometric Progression starting with 6.0 having 4 numbers is :  11.25
```

[ ]:
```
# sum of N terms of an G.P starting with 'a'
def sumOfGP( a, r,n) :
    if n > 0:
        sum = 0   # setting initial sum to zero
        i = 0     # introducing the variable
```

```
        while i < n :
            sum = sum + a  # adding sum by the common ratio
            a = r*a    # increasing the common ratio each time by a muliple of␣
 ↪'r' in the loop
            i = i + 1    # increasing the variable
        return sum


    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this G.P Series: : "))
a = float(input("Please Enter First Number of an G.P Series: : "))
r = 0.5
print("The sum of Geometric Progression starting with " + str(a) + " having "  +␣
 ↪str(n) +  " numbers is : ", end = " ")
print (sumOfGP(a, r, n))      # printing the output
```

[20]:
```
# sum of N terms of an G.P starting with 'a'
def sumOfGP( a, r,n) :
    if n > 0:
        sum = 0   # setting initial sum to zero
        i = 0     # introducing the variable
        while i < n :
            sum = sum + a  # adding sum by the common ratio
            a = r*a    # increasing the common ratio each time by a muliple of␣
 ↪'r' in the loop
            i = i + 1    # increasing the variable
        return sum


    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this G.P Series: : "))
a = float(input("Please Enter First Number of an G.P Series: : "))
r = 0.5
print("The sum of Geometric Progression starting with " + str(a) + " having "  +␣
 ↪str(n) +  " numbers is : ", end = " ")
print (sumOfGP(a, r, n))      # printing the output
```

```
Please Enter the Total Numbers in this G.P Series: : 4
Please Enter First Number of an G.P Series: : -1
```

```
The sum of Geometric Progression starting with -1.0 having 4 numbers is :
-1.875
```

[21]:
```python
# sum of N terms of an G.P starting with 'a'
def sumOfGP( a, r,n) :
    if n > 0:
        sum = 0   # setting initial sum to zero
        i = 0     # introducing the variable
        while i < n :
            sum = sum + a  # adding sum by the common ratio
            a = r*a    # increasing the common ratio each time by a muliple of␣
 ↪'r' in the loop
            i = i + 1    # increasing the variable
        return sum


    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this G.P Series: : "))
a = float(input("Please Enter First Number of an G.P Series: : "))
r = 0.5
print("The sum of Geometric Progression starting with " + str(a) + " having "  +␣
 ↪str(n) +  " numbers is : ", end = " ")
print (sumOfGP(a, r, n))      # printing the output
```

```
Please Enter the Total Numbers in this G.P Series: : -2
Please Enter First Number of an G.P Series: : 0.4
The sum of Geometric Progression starting with 0.4 having -2 numbers is :  Enter
a positive integer for 'n'
None
```

[5]:
```python
# sum of N terms of an H.P starting with 'a'
def sumOfHP( a, d,n) :
    if n > 0:
        sum = 0  # setting initial sum to zero
        i = 0     # introducing the variable
        while i < n :
            x = 1/a   #inverting
            sum = sum + x  # adding sum by the common difference
            a = a + d # increasing the common difference each time by a sum of␣
 ↪'d' in the denominator
            i = i + 1    # increasing the variable
        return sum
```

```
    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this H.P Series: : "))
a = int(input("Please Enter First Number of an H.P Series: : "))
d = 1.5
print("The sum of Geometric Progression starting with " + str(a) + " having "  +␣
 ↪str(n) +  " numbers is : ", end = " ")
print (sumOfHP(a, d, n))      # printing the output
```

```
Please Enter the Total Numbers in this H.P Series: : 3
Please Enter First Number of an H.P Series: : 1
The sum of Geometric Progression starting with 1 having 3 numbers is :  1.65
```

[22]:
```
# sum of N terms of an H.P starting with 'a'
def sumOfHP( a, d,n) :
    if n > 0:
        sum = 0  # setting initial sum to zero
        i = 0     # introducing the variable
        while i < n :
            x = 1/a   #inverting
            sum = sum + x  # adding sum by the common difference
            a = a + d # increasing the common difference each time by a sum of␣
 ↪'d' in the denominator
            i = i + 1    # increasing the variable
        return sum


    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this H.P Series: : "))
a = int(input("Please Enter First Number of an H.P Series: : "))
d = 1.5
print("The sum of Geometric Progression starting with " + str(a) + " having "  +␣
 ↪str(n) +  " numbers is : ", end = " ")
print (sumOfHP(a, d, n))      # printing the output
```

```
Please Enter the Total Numbers in this H.P Series: : 3
Please Enter First Number of an H.P Series: : 5
The sum of Geometric Progression starting with 5 having 3 numbers is :
0.47884615384615387
```

[25]:
```
# sum of N terms of an H.P starting with 'a'
def sumOfHP( a, d,n) :
    if n > 0:
        sum = 0  # setting initial sum to zero
        i = 0     # introducing the variable
        while i < n :
            x = 1/a   #inverting
            sum = sum + x  # adding sum by the common difference
            a = a + d # increasing the common difference each time by a sum of␣
 ↪'d' in the denominator
            i = i + 1    # increasing the variable

        return sum


    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this H.P Series: : "))
a = int(input("Please Enter First Number of an H.P Series: : "))
d = 1.5
print("The sum of Geometric Progression starting with " + str(a) + " having "  +␣
 ↪str(n) +  " numbers is : ", end = " ")
print (sumOfHP(a, d, n))      # printing the output
```

```
Please Enter the Total Numbers in this H.P Series: : 3
Please Enter First Number of an H.P Series: : 5
The sum of Geometric Progression starting with 5 having 3 numbers is :
0.47884615384615387
```

[26]:
```
# sum of N terms of an H.P starting with 'a'
def sumOfHP( a, d,n) :
    if n > 0:
        sum = 0  # setting initial sum to zero
        i = 0     # introducing the variable
        while i < n :
            x = 1/a   #inverting
            sum = sum + x  # adding sum by the common difference
            a = a + d # increasing the common difference each time by a sum of␣
 ↪'d' in the denominator
            i = i + 1    # increasing the variable
        return sum


    elif n <= 0:
```

```
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this H.P Series: : "))
a = int(input("Please Enter First Number of an H.P Series: : "))
d = 1.5
print("The sum of Geometric Progression starting with " + str(a) + " having "  +␣
 ↪str(n) +  " numbers is : ", end = " ")
print (sumOfHP(a, d, n))      # printing the output
```

```
Please Enter the Total Numbers in this H.P Series: : -1
Please Enter First Number of an H.P Series: : 4
The sum of Geometric Progression starting with 4 having -1 numbers is :  Enter a
positive integer for 'n'
None
```

[27]:
```
# sum of N terms of an H.P starting with 'a'
def sumOfHP( a, d,n) :
    if n > 0:
        sum = 0  # setting initial sum to zero
        i = 0     # introducing the variable
        while i < n :
            x = 1/a   #inverting
            sum = sum + x  # adding sum by the common difference
            a = a + d # increasing the common difference each time by a sum of␣
 ↪'d' in the denominator
            i = i + 1    # increasing the variable
        return sum


    elif n <= 0:
        print("Enter a positive integer for 'n' ")


# taking inputs
n = int(input("Please Enter the Total Numbers in this H.P Series: : "))
a = int(input("Please Enter First Number of an H.P Series: : "))
d = 1.5
print("The sum of Geometric Progression starting with " + str(a) + " having "  +␣
 ↪str(n) +  " numbers is : ", end = " ")
print (sumOfHP(a, d, n))      # printing the output
```

```
Please Enter the Total Numbers in this H.P Series: : 4
Please Enter First Number of an H.P Series: : -1
The sum of Geometric Progression starting with -1 having 4 numbers is :
1.7857142857142856
```

[75]:
```
# 4(a): Inverse of exponential function

%run MyLib.ipynb # Running my library

x=input("Enter argument: ") # taking input

x=check_number(x) # Checking for validity

if int(x)!=False:
    eps=10**-6
    i=1

    # the loop will run till the value doesn't match with the actual value
    # of exp(-x) and stops as it matches till the desired decimal places
    while abs(Exp(x,i)-math.exp(-x))>eps:
        i+=1
    print("\nexp(-" + str(x) + ") = " + str(Exp(x,i)))
```

```
Enter argument: 1

exp(-1.0) = 0.3678791887125221
```

[76]:
```
%run MyLib.ipynb # Running my library

x=input("Enter argument: ") # taking input

x=check_number(x) # Checking for validity

if int(x)!=False:
    eps=10**-6
    i=1

    # the loop will run till the value doesn't match with the actual value
    # of exp(-x) and stops as it matches till the desired decimal places
    while abs(Exp(x,i)-math.exp(-x))>eps:
        i+=1
    print("\nexp(-" + str(x) + ") = " + str(Exp(x,i)))
```

```
Enter argument: -1

exp(--1.0) = 2.7182815255731922
```

[77]:
```
%run MyLib.ipynb # Running my library

x=input("Enter argument: ") # taking input
```

```
x=check_number(x) # Checking for validity

if int(x)!=False:
    eps=10**-6
    i=1

    # the loop will run till the value doesn't match with the actual value
    # of exp(-x) and stops as it matches till the desired decimal places
    while abs(Exp(x,i)-math.exp(-x))>eps:
        i+=1
    print("\nexp(-" + str(x) + ") = " + str(Exp(x,i)))
```

```
Enter argument: -2

exp(--2.0) = 7.389055882389215
```

[78]:
```
%run MyLib.ipynb # Running my library

x=input("Enter argument: ") # taking input

x=check_number(x) # Checking for validity

if int(x)!=False:
    eps=10**-6
    i=1

    # the loop will run till the value doesn't match with the actual value
    # of exp(-x) and stops as it matches till the desired decimal places
    while abs(EXP(x,i)-math.exp(-x))>eps:
        i+=1
    print("\nexp(-" + str(x) + ") = " + str(Exp(x,i)))
```

```
Enter argument: -4

exp(--4.0) = 54.59814947621461
```

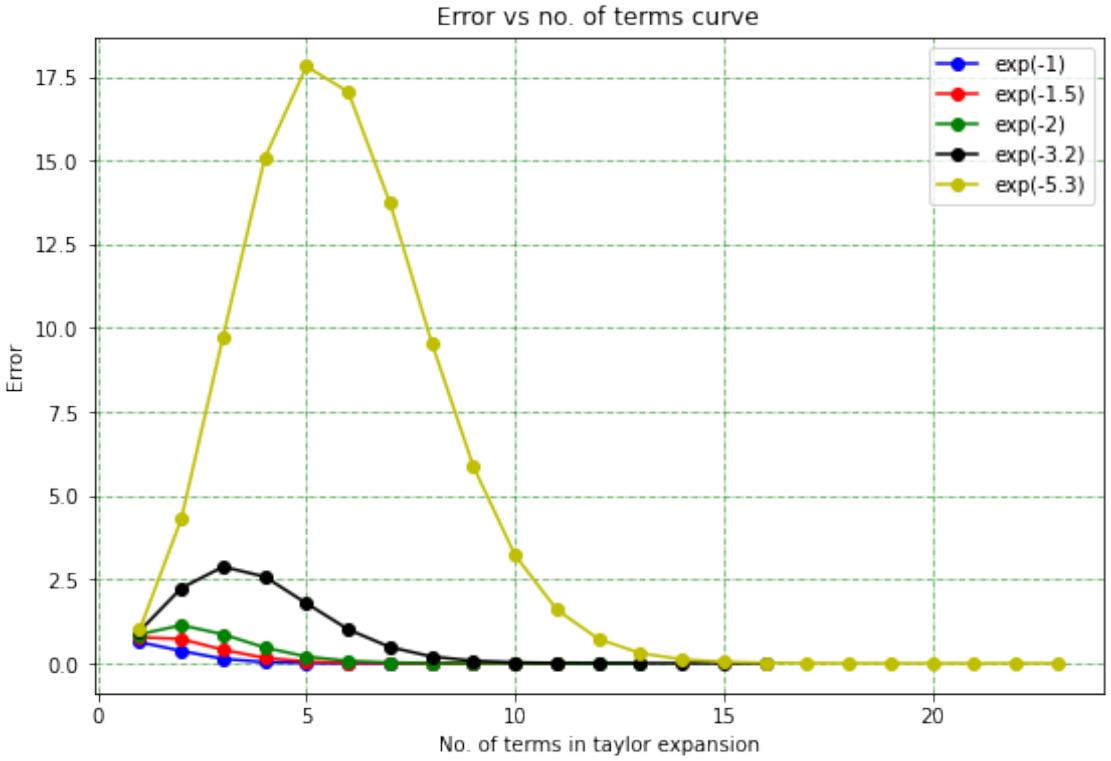
[79]:
```
%run MyLib.ipynb # Running my library

x=input("Enter argument: ") # taking input

x=check_number(x) # Checking for validity

if int(x)!=False:
    eps=10**-6
```

```
    i=1

    # the loop will run till the value doesn't match with the actual value
    # of exp(-x) and stops as it matches till the desired decimal places
    while abs(Exp(x,i)-math.exp(-x))>eps:
        i+=1
    print("\nexp(-" + str(x) + ") = " + str(EXP(x,i)))
```

```
Enter argument: 4

exp(-4.0) = 0.018315259717021463
```

[2]:
```
# 4(a): Plotting error of Inverse exponential function versus number of␣
 ↪iterations in the taylor series

%run MyLib.ipynb # Running my library


import math
import matplotlib.pyplot as plt
plt.figure(figsize=(9,6))

e = 10**-6 # decimal places till which accuracy is needed. 2 decimal places are␣
 ↪increased to reduce round off error

color = ['b-o', 'r-o', 'g-o', 'k-o', 'y-o'] # array of colors for plotting

argument = [1, 1.5, 2, 3.2, 5.3] # array of arguments given for comparison

for j in range(len(argument)):
    # creating arrays to store indices and errors
    index = []
    error = []

    x = argument[j] # argument if sine function
    i = 1

    # the loop will run till the value doesn't match with
    # the actual value of sine and stops as it matches
    # till the desired decimal places
    while abs(Exp(x,i)-math.exp(-x)) > e:
        index.append(i)
        error.append(abs(Exp(x,i)-math.exp(-x)))
        i+=1
    plt.plot(index, error, color[j], label = 'exp(-'+ str(x) +')')
```

```
plt.grid(color='g', ls = '-.', lw = 0.5)
plt.xlabel('No. of terms in taylor expansion')
plt.ylabel('Error')
plt.title('Error vs no. of terms curve')
plt.legend()
plt.show()
```



[71]:
```
# 4(b): Sine function

%run MyLib.ipynb # Running my library

x=input("Enter argument for sine function : ") # taking input

x=check_number(x) # Checking for validity

if int(x)!=False:
    eps=10**-6
    i=1

    #the loop will run till the value doesn't match with the actual value
    #of sin(x) and stops as it matches till the desired decimal places
```

```
    while abs(Sin(x,i)-math.sin(x))>eps:
        i+=1
    print("\nsin(" + str(x) + ") = " + str(Sin(x,i)))
```

```
Enter argument for sine function : 3.5

sin(3.5) = -0.3507830540486009
```

[70]:
```
%run MyLib.ipynb # Running my library

x=input("Enter argument for sine function : ") # taking input

x=check_number(x) # Checking for validity

if int(x)!=False:
    eps=10**-6
    i=1

    #the loop will run till the value doesn't match with the actual value
    #of sin(x) and stops as it matches till the desired decimal places

    while abs(Sin(x,i)-math.sin(x))>eps:
        i+=1
    print("\nsin(" + str(x) + ") = " + str(Sin(x,i)))
```

```
Enter argument for sine function : 5.3

sin(5.3) = -0.8322675214965258
```

[81]:
```
%run MyLib.ipynb # Running my library

x=input("Enter argument for sine function : ") # taking input

x=check_number(x) # Checking for validity

if int(x)!=False:
    eps=10**-6
    i=1

    #the loop will run till the value doesn't match with the actual value
    #of sin(x) and stops as it matches till the desired decimal places

    while abs(Sin(x,i)-math.sin(x))>eps:
        i+=1
    print("\nsin(" + str(x) + ") = " + str(Sin(x,i)))
```

```
Enter argument for sine function : 1.5
```

```
sin(1.5) = 0.9974949556821353
```

[69]:
```
%run MyLib.ipynb # Running my library

x=input("Enter argument for sine function : ") # taking input

x=check_number(x) # Checking for validity

if int(x)!=False:
    eps=10**-6
    i=1

    #the loop will run till the value doesn't match with the actual value
    #of sin(x) and stops as it matches till the desired decimal places

    while abs(Sin(x,i)-math.sin(x))>eps:
        i+=1
    print("\nsin(" + str(x) + ") = " + str(Sin(x,i)))
```

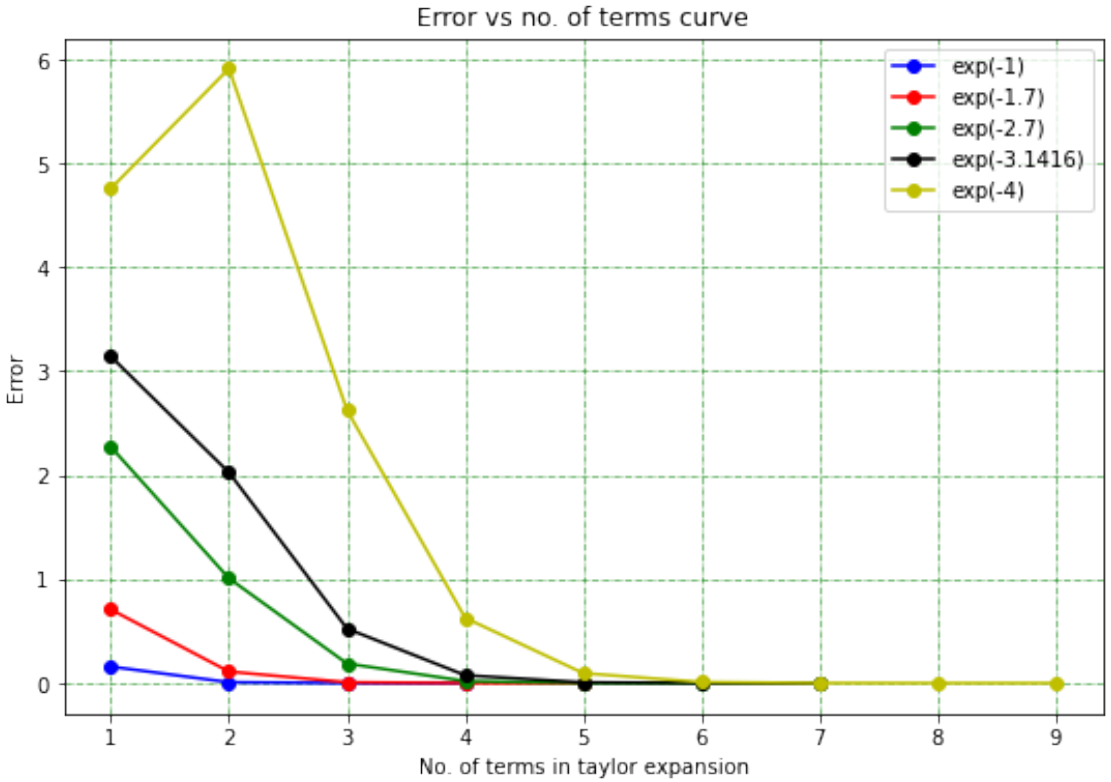
```
Enter argument for sine function : -1

sin(-1.0) = -0.8414710097001764
```

[73]:
```
%run MyLib.ipynb # Running my library

x=input("Enter argument for sine function : ") # taking input

x=check_number(x) # Checking for validity

if int(x)!=False:
    eps=10**-6
    i=1

    #the loop will run till the value doesn't match with the actual value
    #of sin(x) and stops as it matches till the desired decimal places

    while abs(Sin(x,i)-math.sin(x))>eps:
        i+=1
    print("\nsin(" + str(x) + ") = " + str(Sin(x,i)))
```

```
Enter argument for sine function : 1

sin(1.0) = 0.8414710097001764
```

[3]:
```
# 4(b): Plotting of error of Sine function versus number of iterations in the␣
 ↪taylor series

%run MyLib.ipynb # Running my library
```

```
import math
import matplotlib.pyplot as plt
plt.figure(figsize=(9,6))

e = 10**-6 # decimal places till which accuracy is needed. 2 decimal places are␣
 ↪increased

color = ['b-o', 'r-o', 'g-o', 'k-o', 'y-o'] # array of colors for plotting

argument = [1, 1.7, 2.7, 3.1416, 4] # array of arguments given for comparison

for j in range(len(argument)):
    # creating arrays to store indices and errors
    index = []
    error = []

    x = argument[j] # argument if sine function
    i = 1

    # the loop will run till the value doesn't match with
    # the actual value of sin(x) and stops as it matches
    # till the desired decimal places
    while abs(Sin(x,i)-math.sin(x)) > e:
        index.append(i)
        error.append(abs(Sin(x,i)-math.sin(x)))
        i+=1
    plt.plot(index, error, color[j], label = 'exp(-'+ str(x) +')')
plt.grid(color='g', ls = '-.', lw = 0.5)
plt.xlabel('No. of terms in taylor expansion')
plt.ylabel('Error')
plt.title('Error vs no. of terms curve')
plt.legend()
plt.show()
```


Error vs no. of terms curve
exp(-1)
exp(-1.7)
exp(-2.7)
exp(-3.1416)
exp(-4)
Error
No. of terms in taylor expansion
0
1
2
3
4
5
6
1
2
3
4
5
6
7
8
9

[ ]: