Question 1

In [1]:

```
##############################################################################

# Question 1(a): Sum of first N natural numbers

%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

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

# Checking for validity
n=check_natural_number(n)

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

##############################################################################
```

```
Enter a natural number 9

The sum of first 9 odd numbers is 45
```

In [7]:

```
%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

# array of input
n=['3', '-19', '42', 'abc', '250', '0']

for i in range(len(n)):
    print("\nValue of n given : " + n[i])
    print("Output: ")
    n[i]=check_natural_number(n[i])
    if n[i]!='F':
        sum_natural_numbers(n[i])
```

```

Value of n given : 3
Output: 
The sum of first 3 odd numbers is 6

Value of n given : -19
Output: 
Invalid input. Please enter a natural number.

Value of n given : 42
Output: 
The sum of first 42 odd numbers is 903

Value of n given : abc
Output: 
Invalid input. Please enter a natural number.

Value of n given : 250
Output: 
The sum of first 250 odd numbers is 31375

Value of n given : 0
Output: 
Invalid input. Please enter a natural number.
```

In [9]:

```
##########################################################################

# Question 1(b): Sum of first N odd natural numbers
%run CPL_Library.ipynb

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

# Checking for validity
n=check_natural_number(n)
if n!='F':
    sum_odd_numbers(n)

##########################################################################
```

```
Enter a natural number :  6

The sum of first 6 odd numbers is 36
```

In [10]:
```python
%run CPL_Library.ipynb # Running the entire library in one line because
                       # importing does not work with JupyterLab

# array of input
n=['3', '-19', '42', 'abc', '250', '0']

for i in range(len(n)):
    print("\nValue of n given : " + n[i])
    print("Output: ")
    n[i]=check_natural_number(n[i])
    if n[i]!='F':
        sum_odd_numbers(n[i])
```

```
Value of n given : 3
Output:
The sum of first 3 odd numbers is 9

Value of n given : -19
Output:
Invalid input. Please enter a natural number.

Value of n given : 42
Output:
The sum of first 42 odd numbers is 1764

Value of n given : abc
Output:
Invalid input. Please enter a natural number.

Value of n given : 250
Output:
The sum of first 250 odd numbers is 62500

Value of n given : 0
Output:
Invalid input. Please enter a natural number.
```

Question 2

In [12]:

```
################################################################################

# Question 2(a): Sum of first N entries of an AP

%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

# taking input from the user
a=input("Enter first term of AP: ")
n=input("Enter a number of terms: ")

# Checking for validity
a=check_number(a)
n=check_natural_number(n)

if (a!='F' and n!='F'):
    sum_AP(a,n)

################################################################################
```

```
Enter first term of AP:  5
Enter a number of terms:  3

Common difference is : 1.5

The sum of first 3 terms of an AP is 19.5
```

In [13]:

```
%run CPL_Library.ipynb # Running the entire library in one line because
                       # importing does not work with JupyterLab

# array of input
a=['-2', '1', '5', '2.3', '7.5', '10']
n=['3', '-19', '14', 'abc', '25', '0']

for i in range(len(n)):
    print("\nValue of n given : " + n[i])
    print("Output: ")
    a[i]=check_number(a[i])
    n[i]=check_natural_number(n[i])
    if (a[i]!='F' and n[i]!='F'):
        sum_AP(a[i],n[i])
```

```
Value of n given : 3
Output:
Common difference is : 1.5

The sum of first 3 terms of an AP is -1.5

Value of n given : -19
Output:
Invalid input. Please enter a natural number.

Value of n given : 42
Output:
Common difference is : 1.5

The sum of first 42 terms of an AP is 1501.5

Value of n given : abc
Output:
Invalid input. Please enter a natural number.

Value of n given : 250
Output:
Common difference is : 1.5

The sum of first 250 terms of an AP is 48562.5

Value of n given : 0
Output:
Invalid input. Please enter a natural number.
```

In [14]:

```
#############################################################################

# Question 2(b): Sum of first N terms of a GP

%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

# taking input from the user
a=input("Enter first term of GP: ")
n=input("Enter number of terms: ")

# Checking for validity
a=check_number(a)
n=check_natural_number(n)
r=0.5
if (a!='F' and n!='F'):
    sum_GP(a,n,r)

#############################################################################
```

```
Enter first term of GP:  3
Enter number of terms:  5

Common ratio is : 0.5

The sum of first 5 terms of an GP is 5.8125
```

In [15]:

```
%run CPL_Library.ipynb # Running the entire library in one line because
                       # importing does not work with JupyterLab

# array of input
a=['-2', '1', '5', '2.3', '7.5', '10']
n=['3', '-19', '14', 'abc', '25', '0']

for i in range(len(n)):
    print("\nValue of n given : " + n[i])
    print("Output: ")
    a[i]=check_number(a[i])
    n[i]=check_natural_number(n[i])
    if (a[i]!='F' and n[i]!='F'):
        sum_GP(a[i],n[i])
```

```
Value of n given : 3
Output:
Common ratio is : 0.5

The sum of first 3 terms of an GP is -3.5

Value of n given : -19
Output:
Invalid input. Please enter a natural number.

Value of n given : 14
Output:
Common ratio is : 0.5

The sum of first 14 terms of an GP is 9.9993896484375

Value of n given : abc
Output:
Invalid input. Please enter a natural number.

Value of n given : 25
Output:
Common ratio is : 0.5

The sum of first 25 terms of an GP is 14.999999552965164

Value of n given : 0
Output:
Invalid input. Please enter a natural number.
```

In [8]:

```
##########################################################################

# Question 2(c): Sum of first N terms of a HP

%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

# taking input from the user
a=input("Enter first term of HP: ")
n=input("Enter number of terms: ")

# Checking for validity
a=check_number(a)
n=check_natural_number(n)

if (a!='F' and n!='F'):
    sum_HP(a,n)

##########################################################################
```

```
Enter first term of HP:  3
Enter number of terms:  3


The sum of first 3 terms of an HP is 0.7222222222222222
```

In [32]:
```
%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

# array of input
a=['-2', '1', '5', '2.3', '7.5', '10']
n=['3', '-19', '14', 'abc', '25', '0']

for i in range(len(n)):
    print("\nValue of n given : " + n[i])
    print("Output: ")
    a[i]=check_number(a[i])
    n[i]=check_natural_number(n[i])
    if (a[i]!='F' and n[i]!='F'):
        sum_HP(a[i],n[i])
```

```
Value of n given : 3
Output:
Common difference is : 1.5

The sum of first 3 terms of an HP is -1.5

Value of n given : -19
Output:
Invalid input. Please enter a natural number.

Value of n given : 14
Output:
Common difference is : 1.5

The sum of first 14 terms of an HP is 1.184646901883784

Value of n given : abc
Output:
Invalid input. Please enter a natural number.

Value of n given : 25
Output:
Common difference is : 1.5

The sum of first 25 terms of an HP is 1.2522136428358162

Value of n given : 0
Output:
Invalid input. Please enter a natural number.
```

Question 3

In [45]:

```
# ###########################################################################

# Question 3: Factorial of a number

%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

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

# Checking for validity
n=check_natural_number(n)
if n!='F':
    # finding factorial of 0 separately and other natural numbers separately
    if n=='zero':
        print("Factorial of 0 is : 1")
    elif n!='F':
        print("Factorial of " + str(n) + " is : " + str(FACTORIAL(n)))

############################################################################
```

```
Enter a natural number :  7

Factorial of 7 is : 5040
```

In [44]:

```
%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

# array of input
n=['3', '-19', '7', 'abc', '20', '0']

for i in range(len(n)):
    print("\nValue of n given : " + n[i])
    print("Output: ")

    # Checking for validity
    n[i]=check_natural_number(n[i])
    if n[i]!='F':
        # finding factorial of 0 separately and other natural numbers separat
        if n[i]=='zero':
            print("Factorial of 0 is : 1")
        elif n!='F':
            print("Factorial of " + str(n[i]) + " is : " + str(FACTORIAL(n[i]
```

```
Value of n given : 3
Output:
Factorial of 3 is : 6

Value of n given : -19
Output:
Invalid input. Please enter a natural number.

Value of n given : 7
Output:
Factorial of 7 is : 5040

Value of n given : abc
Output:
Invalid input. Please enter a natural number.

Value of n given : 20
Output:
Factorial of 20 is : 2432902008176640000

Value of n given : 0
Output:
Factorial of 0 is : 1
```

Question 4

In [51]:

```
################################################################################
# Question 4(a): Sine function

%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

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

# Checking for validity
x=check_number(x)

if x!='F':
    eps=10**-6
    i=1

    # the loop runs till the function value doesn't match with the actual val
    # of sin(x) and terminates as it matches upto desired decimal places
    while abs(SINE(x,i)-math.sin(x))>eps:
        i+=1
    print("\nsin(" + str(x) + ") = " + str(SINE(x,i)))
    print("The value is accurate atleast upto 4 decimal places")

################################################################################
```

```
Enter argument for sine function :  y

Invalid input. Please enter a number.
```

In [54]:

```
%run CPL_Library.ipynb # Running the entire library in one line because
                       # importing does not work with JupyterLab

# array of input
x=['0.5', '-1', '1.2', 'abc', '4', '0']
eps=10**-6
for j in range(len(x)):
    print("\nValue of n given : " + x[j])
    print("Output: ")

    # Checking for validity
    x[j]=check_number(x[j])

    if x[j]!='F':
        i=1

        # the loop runs till the function value doesn't match with the actual
        # of sin(x) and terminates as it matches upto desired decimal places
        while abs(SINE(x[j],i)-math.sin(x[j]))>eps:
            i+=1
        print("sin(" + str(x[j]) + ") = " + str(SINE(x[j],i)))
        print("The value is accurate atleast upto 4 decimal places")
```

```
Value of n given : 0.5
Output:
sin(0.5) = 0.479425533234127
The value is accurate atleast upto 4 decimal places

Value of n given : -1
Output:
sin(-1.0) = -0.8414710097001764
The value is accurate atleast upto 4 decimal places

Value of n given : 1.2
Output:
sin(1.2) = 0.9320392703999999
The value is accurate atleast upto 4 decimal places

Value of n given : abc
Output:
Invalid input. Please enter a number.

Value of n given : 4
Output:
sin(4.0) = -0.7568025787396139
The value is accurate atleast upto 4 decimal places

Value of n given : 0
Output:
sin(0.0) = 0.0
The value is accurate atleast upto 4 decimal places
```

In [56]:

```
################################################################################
# Question 4(a): Inverse of exponential function

%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

# taking input from the user
x=input("Enter argument for inverse of exponential function : ")

# Checking for validity
x=check_number(x)

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

    # the loop runs till the function value doesn't match with the actual val
    # of exp(-x) and terminates as it matches upto desired decimal places
    while abs(EXP(x,i)-math.exp(-x))>eps:
        i+=1
    print("\nexp(-" + str(x) + ") = " + str(EXP(x,i)))
    print("The value is accurate atleast upto 4 decimal places")

################################################################################
```

```
Enter argument for inverse of exponential function :  5


exp(-5.0) = 0.006738328152479823
The value is accurate atleast upto 4 decimal places
```

In [57]:

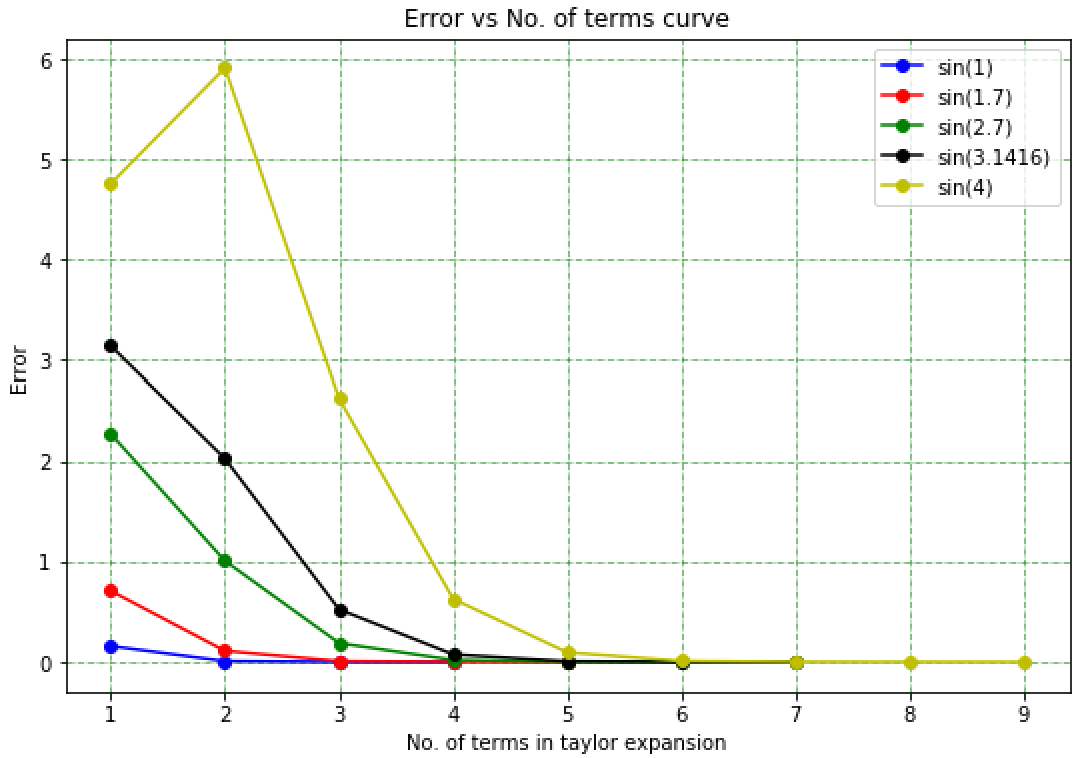
```
%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

# array of input
x=['0.5', '-1', '1.2', 'abc', '4', '0']
eps=10**-6
for j in range(len(x)):
    print("\nValue of n given : " + x[j])
    print("Output: ")

    # Checking for validity
    x[j]=check_number(x[j])

    if x[j]!='F':
        i=1

        # the loop runs till the function value doesn't match with the actual
        # of exp(x) and terminates as it matches upto desired decimal places
        while abs(SINE(x[j],i)-math.sin(x[j]))>eps:
            i+=1
        print("exp(-" + str(x[j]) + ") = " + str(EXP(x[j],i)))
        print("The value is accurate atleast upto 4 decimal places")
```

```
Value of n given : 0.5
Output:
exp(-0.5) = 0.6041666666666666
The value is accurate atleast upto 4 decimal places

Value of n given : -1
Output:
exp(--1.0) = 2.708333333333333
The value is accurate atleast upto 4 decimal places

Value of n given : 1.2
Output:
exp(-1.2) = 0.3184
The value is accurate atleast upto 4 decimal places

Value of n given : abc
Output:
Invalid input. Please enter a number.

Value of n given : 4
Output:
exp(-4.0) = -0.19223985890652528
The value is accurate atleast upto 4 decimal places

Value of n given : 0
Output:
exp(-0.0) = 1.0
The value is accurate atleast upto 4 decimal places
```

Plotting the errors

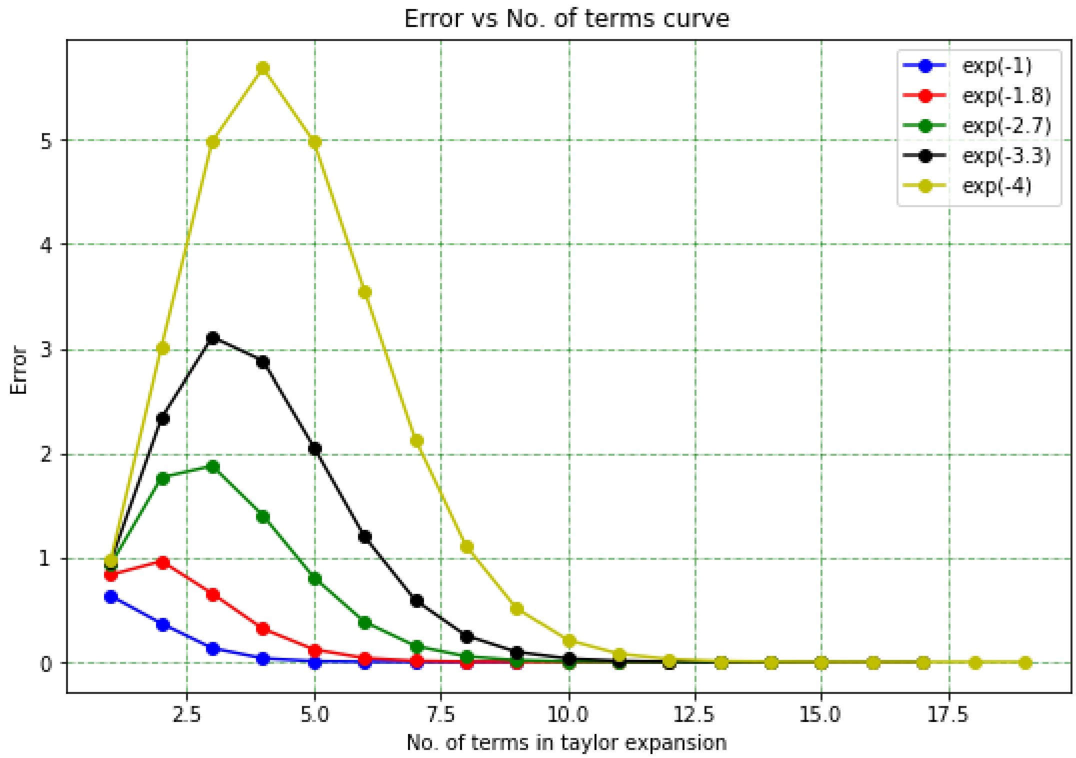
In [61]:

```
#####################################################################################

# Question 4(b): Plotting of error of Sine function

%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

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

eps=10**-6 # value of epsilon - decimal places upto which accuracy is desired

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)):
    # initializing two arrays to store indices and errors
    index=[]
    error=[]

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

    # the loop runs till the value doesn't match with
    # the actual value of sine and terminates as it matches
    # upto desired decimal places
    while abs(SINE(x,i)-math.sin(x))>eps:
        index.append(i)
        error.append(abs(SINE(x,i)-math.sin(x)))
        i+=1
    plt.plot(index, error, color[j], label='sin('+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
sin(1)
sin(1.7)
sin(2.7)
sin(3.1416)
sin(4)
Error
0
1
2
3
4
5
6
1
2
3
4
5
6
7
8
9
No. of terms in taylor expansion

In [62]:

```python
################################################################################

# Question 4(b): Plotting of error of Sine function

%run CPL_Library.ipynb # Running the entire library in one line because
                        # importing does not work with JupyterLab

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

eps=10**-6 # value of epsilon - decimal places upto which accuracy is desired

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

argument=[1, 1.8, 2.7, 3.3, 4] # array of arguments given for comparison

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

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

    # the loop runs till the value doesn't match with
    # the actual value of sine and terminates as it matches
    # upto desired decimal places
    while abs(EXP(x,i)-math.exp(-x))>eps:
        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()

################################################################################
```


Error vs No. of terms curve
exp(-1)
exp(-1.8)
exp(-2.7)
exp(-3.3)
exp(-4)
Error
No. of terms in taylor expansion
0
1
2
3
4
5
2.5
5.0
7.5
10.0
12.5
15.0
17.5

In [ ]: