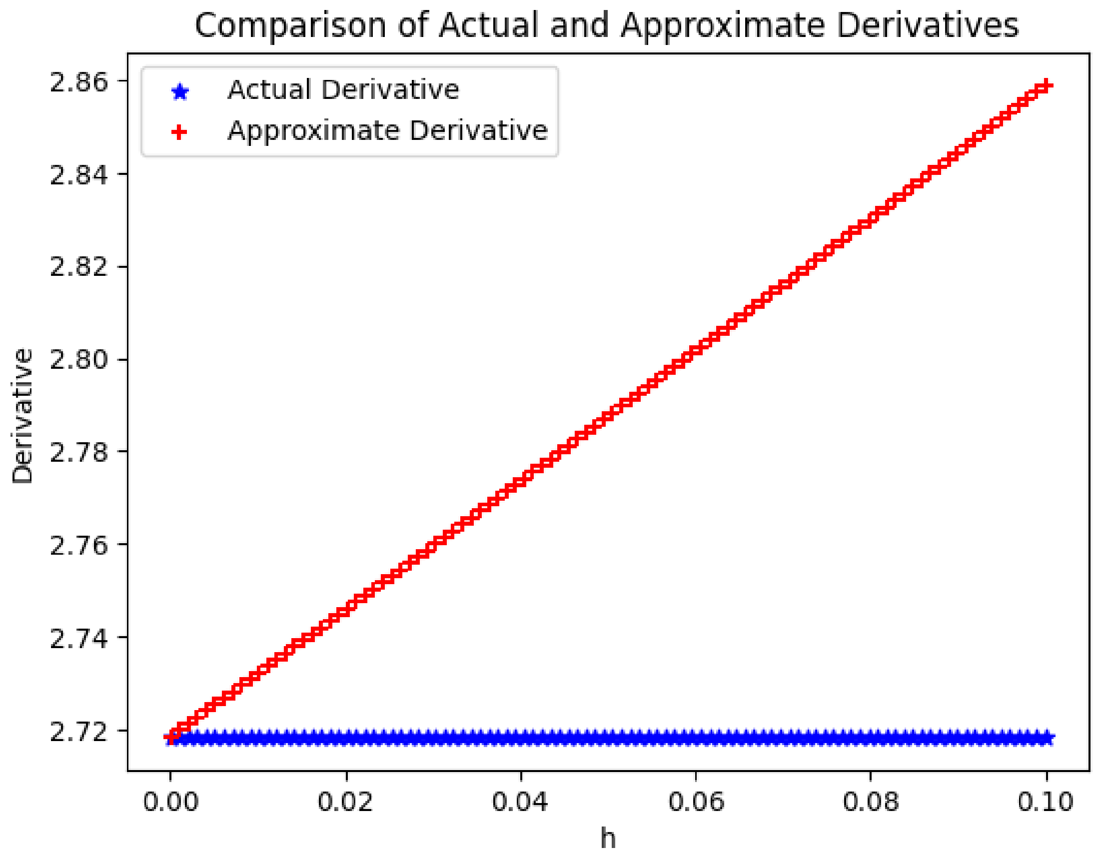
```python
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

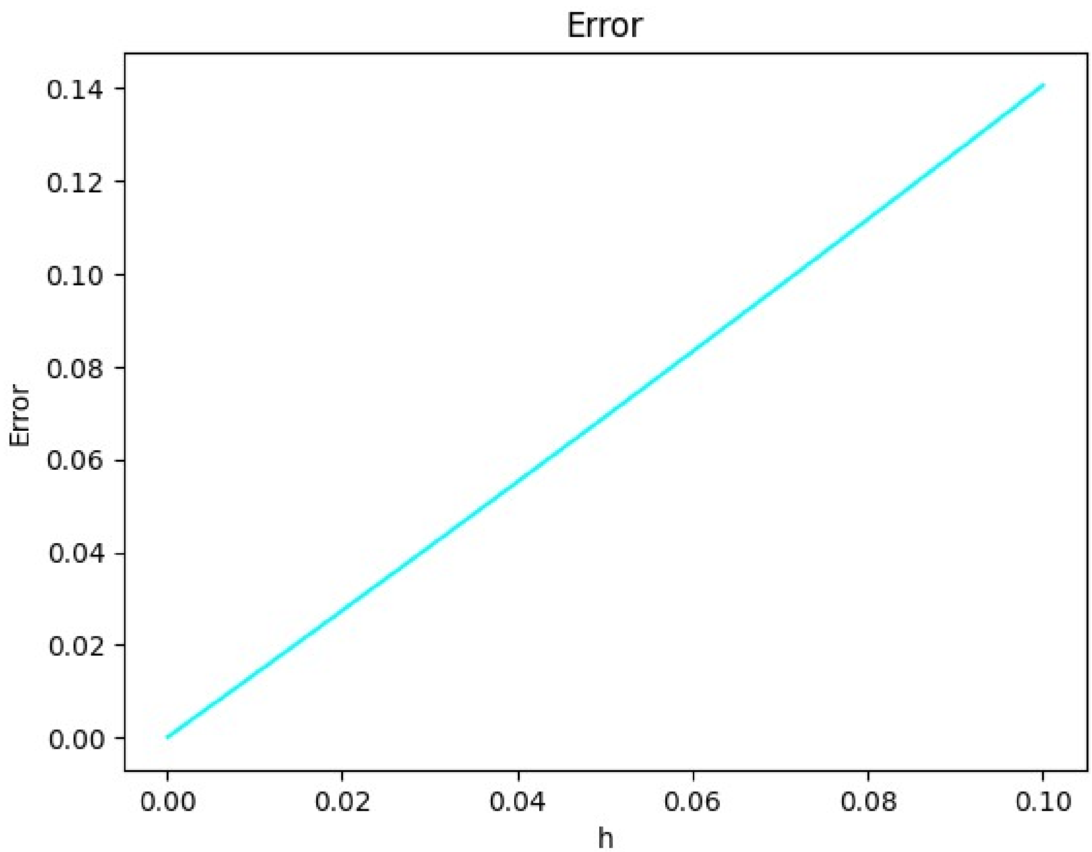
Question 1

```python
x=1
h=np.linspace(0.1,0.0001,100)

def function(x):
  return np.exp(x)
def derivative(x):
  return np.exp(x)
def approx(x,h):
  return (function(x+h)-function(x))/h
def error(x,h):
  return abs(derivative(x)-approx(x,h))

print("Actual derivative value at 1 =",derivative(x))
plt.scatter(h, [derivative(1)] * len(h), label='Actual Derivative',
color='blue', marker='*')
plt.scatter(h, approx(1,h), label='Approximate Derivative',
color='red', marker='+')
plt.xlabel('h')
plt.ylabel('Derivative')
plt.title('Comparison of Actual and Approximate Derivatives')
plt.legend()
plt.show()
```

```
Actual derivative value at 1 = 2.718281828459045
```



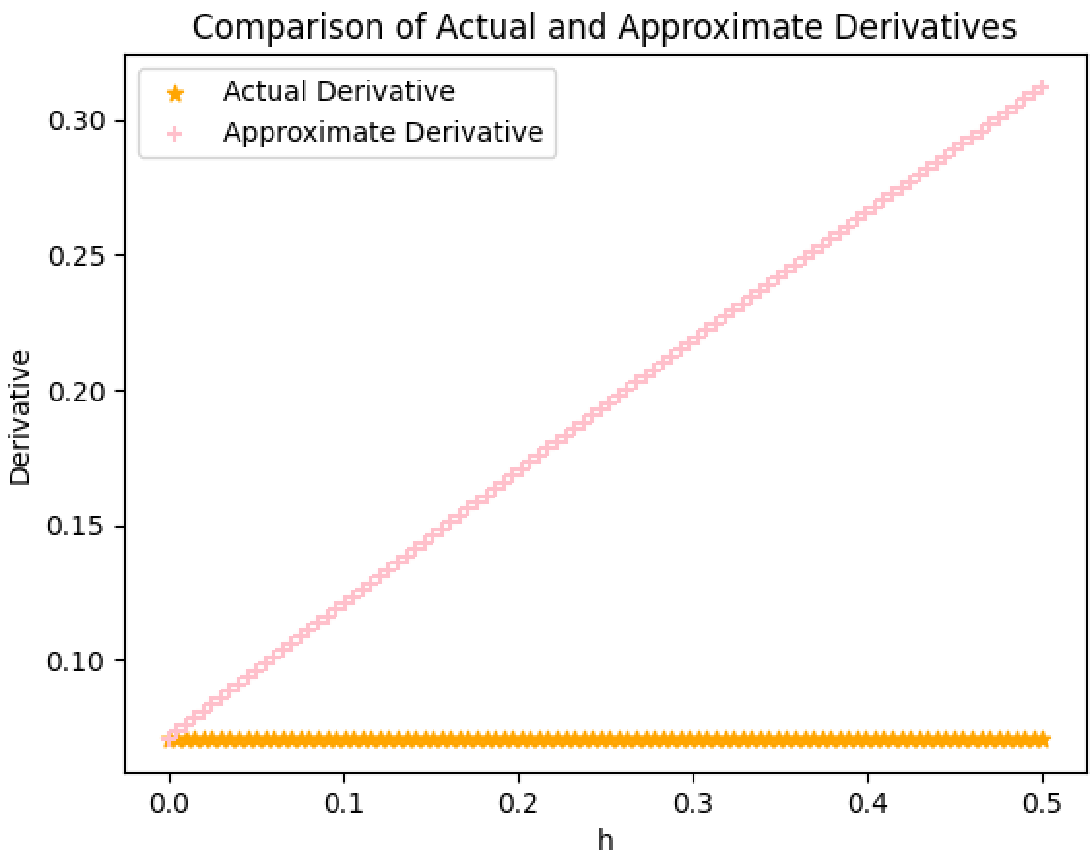

```
plt.plot(h, error(1,h), label='Error', color='cyan')
plt.title("Error")
plt.xlabel("h")
plt.ylabel("Error")
plt.show()
```



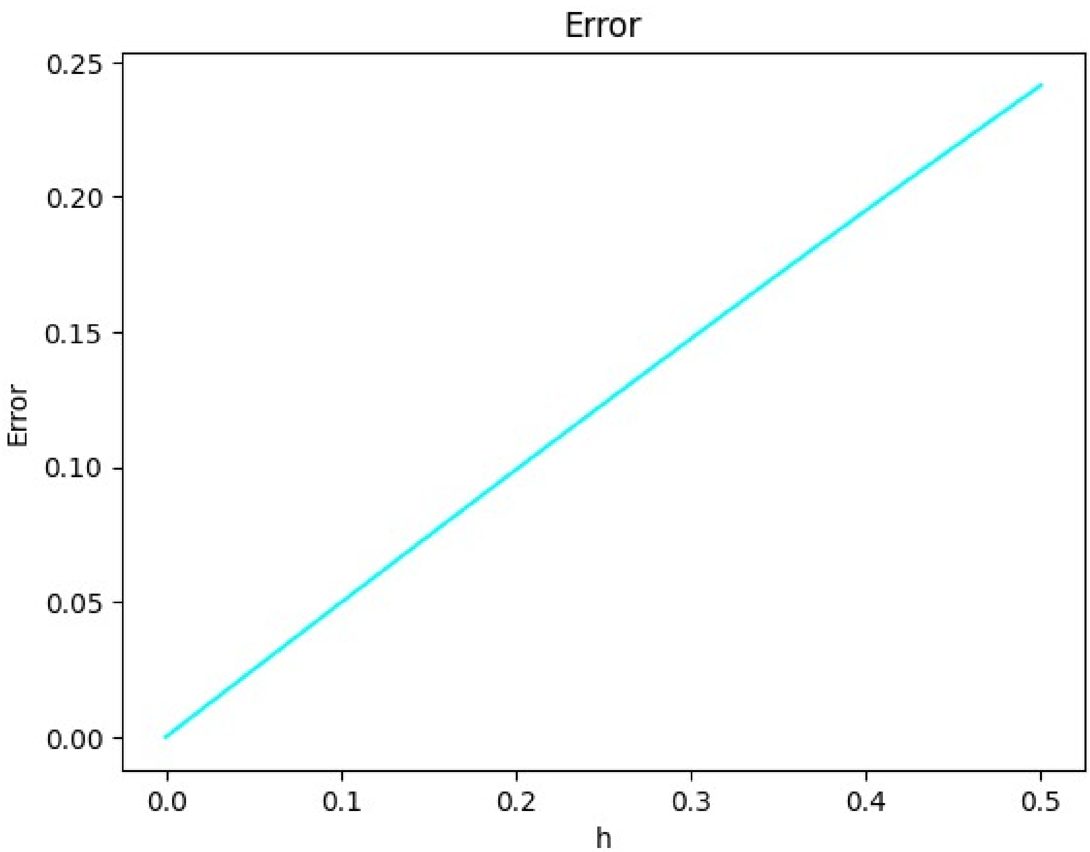
Question 2

```
p=1.5
h=np.linspace(0.5,0.000005,100)

def function(p):
  return np.sin(p)
def derivative(p):
  return np.cos(p)
def approx(p,h):
  return (function(p)-function(p-h))/h
def error(p,h):
  return abs(derivative(p)-approx(p,h))

print("Actual derivative value at 1.5 =",derivative(p))
plt.scatter(h, [derivative(p)] * len(h), label='Actual Derivative',
color='orange', marker='*')
plt.scatter(h, approx(p,h), label='Approximate Derivative',
color='pink', marker='+')
plt.xlabel('h')
plt.ylabel('Derivative')
plt.title('Comparison of Actual and Approximate Derivatives')
```

```
plt.legend()
plt.show()

Actual derivative value at 1.5 = 0.0707372016677029
```




```
plt.plot(h, error(p,h), label='Error', color='cyan')
plt.title("Error")
plt.xlabel("h")
plt.ylabel("Error")
plt.show()
```



Question 3

```
p=2
h=0.1

def function(p):
  return np.log(p)
def derivative(p):
  return 1/p
def approx(p,h):
  return (function(p+h)-function(p-h))/(2*h)
def error(p,h):
  return abs(derivative(p)-approx(p,h))

print("Actual derivative value at 2 =",7
```

```
Actual derivative value at 2 = 0.5
Approximate derivative value at 2 = 0.5004172927849132
Error = 0.00041729278491320354
```

Question 4

```
p=1
h=0.1

def function(p):
  return p**3
def derivative(p):
  return 3*(p**2)
def second_derivative(p):
  return 6*p
def approx(p,h):
  return (function(p+h)-2*(function(p))+function(p-h))/(h**2)
def error(p,h):
  return abs(second_derivative(p)-approx(p,h))

print("Actual second derivative value at 1 =",second_derivative(p))
print("Approximate second derivative value at 1 =",approx(p,h))
print("Error =",error(p,h))
```

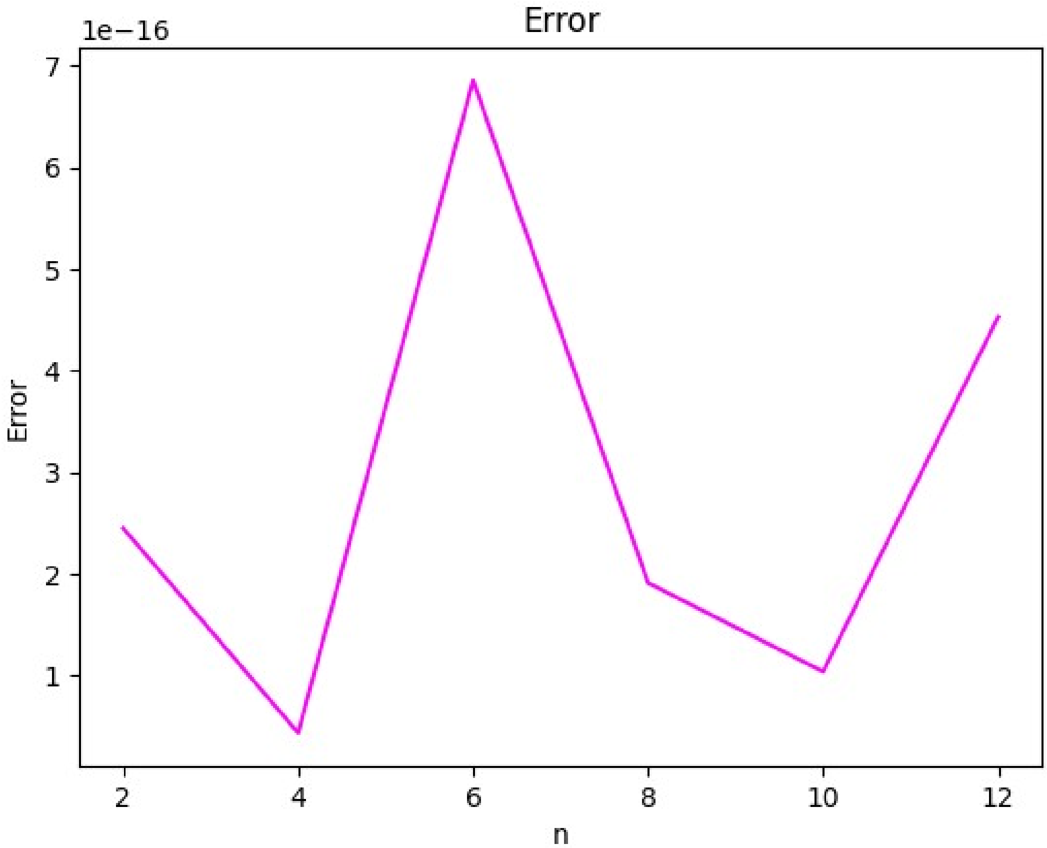
```
Actual second derivative value at 1 = 6
Approximate second derivative value at 1 = 6.000000000000049
Error = 4.884981308350689e-14
```

Question 5

```
a=0
b=2*(np.pi)
n=[2,4,6,8,10,12]
error_in_integral=[]
def function(x):
  return np.cos(x)
def exact_integral(a,b):
  return np.sin(b)-np.sin(a)

print("Exact integral = ",exact_integral(a,b))

for n_val in n:
  h=(b-a)/n_val
  sum=0
  f_x0=function(a)
  f_xn=function(b)
  sum=sum+f_x0+f_xn
  for i in range(1,n_val):
    sum=sum+(2*function(a+i*h))
  sum=sum*(h/2)
  print("Approximate integral for n =",n_val,"is",sum)
  print("\n")
```

```
    print("Error for n =",n_val,"is",abs(exact_integral(a,b)-sum))
    error=abs(exact_integral(a,b)-sum)
    error_in_integral.append(error)

plt.plot(n,error_in_integral, label='Error', color='magenta')
plt.title("Error")
plt.xlabel("n")
plt.ylabel("Error")
plt.show()
```

```
Exact integral =  -2.4492935982947064e-16
Approximate integral for n = 2 is 0.0


Error for n = 2 is 2.4492935982947064e-16
Approximate integral for n = 4 is -2.8855060405826847e-16


Error for n = 4 is 4.3621244228797825e-17
Approximate integral for n = 6 is -9.30098266135635e-16


Error for n = 6 is 6.851689063061644e-16
Approximate integral for n = 8 is -4.3598356225107897e-16


Error for n = 8 is 1.9105420242160833e-16
Approximate integral for n = 10 is -3.487868498008632e-16


Error for n = 10 is 1.0385748997139253e-16
Approximate integral for n = 12 is -6.975736996017264e-16


Error for n = 12 is 4.526443397722557e-16
```



Question 6

```
a=0
b=1
h=(b-a)/6

def function(x):
  return 1/(1+x**2)
def exact_integral(a,b):
  return np.arctan(b)-np.arctan(a)

print("Exact integral = ",exact_integral(a,b))

sum =0;
f_x0=function(a)
f_xn=function(b)
sum=sum+f_x0+f_xn
for i in range(1,6):
  if i%2==0:
    sum=sum+(2*function(a+i*h))
  else:
    sum=sum+(4*function(a+i*h))
```

```
sum=sum*(h/3)
print("Approximate integral for n =",6,"is",sum)
print("\n")
print("Error for n =",6,"is",abs(exact_integral(a,b)-sum))
```

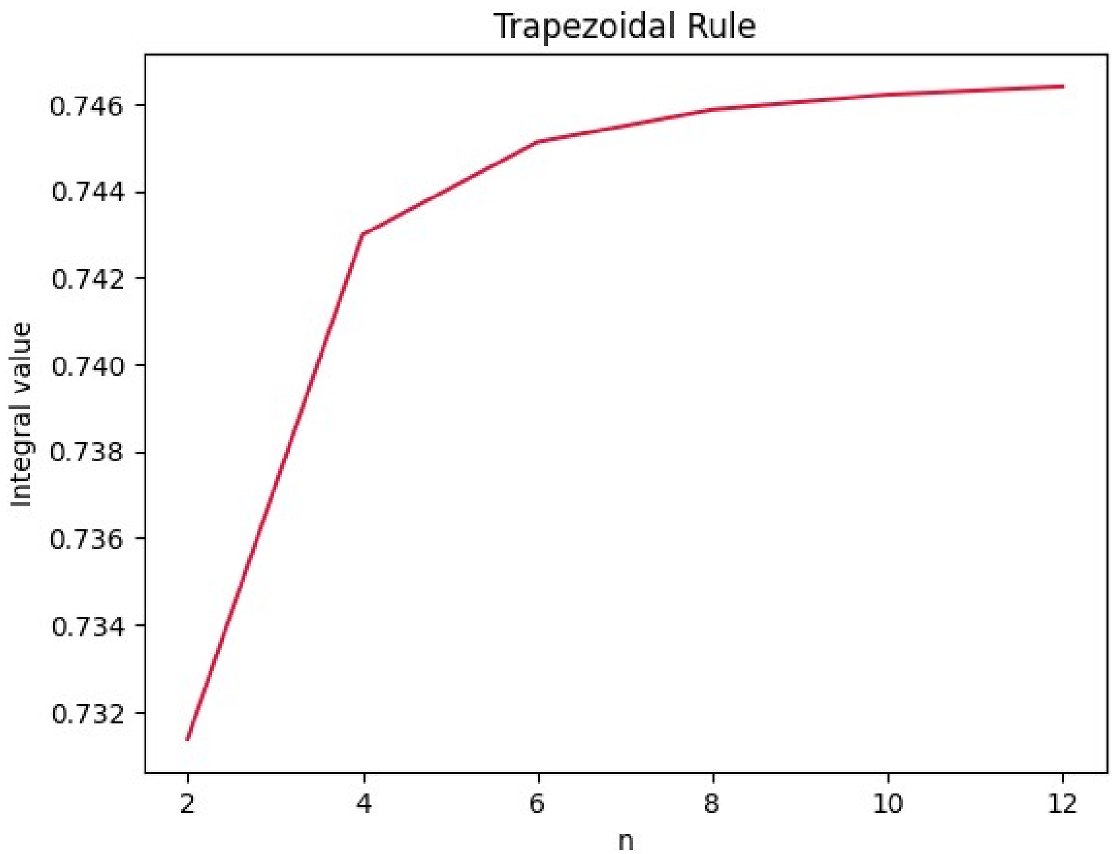
```
Exact integral =  0.7853981633974483
Approximate integral for n = 6 is 0.7853979452340107


Error for n = 6 is 2.1816343753755518e-07
```

Question 7

```
a=0
b=1
n=[2,4,6,8,10,12]
integral_trap=[]
def function(x):
  return np.exp((-1)*(x**2))

def trapezoidal(a,b,n):
  for n_val in n:
    h=(b-a)/n_val
    sum=0
    f_x0=function(a)
    f_xn=function(b)
    sum=sum+f_x0+f_xn
    for i in range(1,n_val):
      sum=sum+(2*function(a+i*h))
    sum=sum*(h/2)
    integral_trap.append(sum)
    print("Approximate integral for n =",n_val,"is ",sum)
    print("\n")




trapezoidal(a,b,n)
plt.plot(n,integral_trap, label='Trapezoidal', color='crimson')
plt.title("Trapezoidal Rule")
plt.xlabel("n")
plt.ylabel("Integral value")
plt.show()
```

```
Approximate integral for n = 2 is  0.731370251828563


Approximate integral for n = 4 is  0.7429840978003812
```

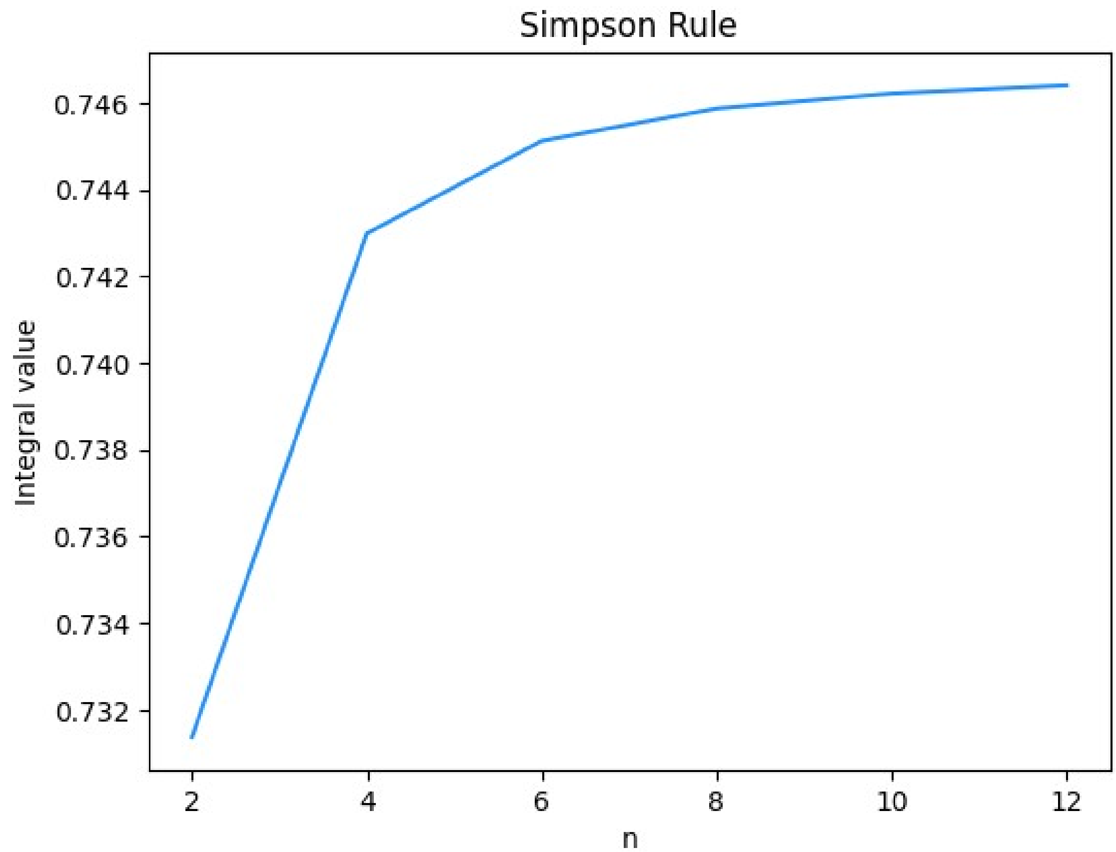
```
Approximate integral for n = 6 is  0.7451194124361793

Approximate integral for n = 8 is  0.7458656148456951

Approximate integral for n = 10 is  0.7462107961317495

Approximate integral for n = 12 is  0.7463982478934403
```




```
a=0
b=1
n=[2,4,6,8,10,12]
simpson_=[]
def simpson(a,b,n):
  for n_val in n:
    h=(b-a)/n_val
    sum=0;
    f_x0=function(a)
```

```
    f_xn=function(b)
    sum=sum+f_x0+f_xn
    for i in range(1,n_val):
      if i%2==0:
        sum=sum+(2*function(a+i*h))
      else:
        sum=sum+(4*function(a+i*h))
    sum=sum*(h/3)
    simpson_.append(sum)
    print("Approximate integral for n =",n_val,"is ",sum)
    print("\n")


simpson(a,b,n)
plt.plot(n,integral_trap, label='Simspon', color='dodgerblue')
plt.title("Simpson Rule")
plt.xlabel("n")
plt.ylabel("Integral value")
plt.show()
```

```
Approximate integral for n = 2 is  0.7471804289095103


Approximate integral for n = 4 is  0.7468553797909873


Approximate integral for n = 6 is  0.7468303914893449


Approximate integral for n = 8 is  0.7468261205274664


Approximate integral for n = 10 is  0.7468249482544436


Approximate integral for n = 12 is  0.7468245263791943
```




```
a=0
b=1
n=[2,4,6,8,10,12]
integral_trap=[]
def function(x):
  return np.exp((-1)*(x**2))

def trapezoidal(a,b,n):
  for n_val in n:
    h=(b-a)/n_val
    sum=0
    f_x0=function(a)
    f_xn=function(b)
    sum=sum+f_x0+f_xn
    for i in range(1,n_val):
      sum=sum+(2*function(a+i*h))
    sum=sum*(h/2)
    integral_trap.append(sum)
    print("Approximate integral for n =",n_val,"is ",sum)
    print("\n")
```

```
trapezoidal(a,b,n)

a=0
b=1
n=[2,4,6,8,10,12]
simpson_=[]
def simpson(a,b,n):
  for n_val in n:
    h=(b-a)/n_val
    sum=0;
    f_x0=function(a)
    f_xn=function(b)
    sum=sum+f_x0+f_xn
    for i in range(1,n_val):
      if i%2==0:
        sum=sum+(2*function(a+i*h))
      else:
        sum=sum+(4*function(a+i*h))
    sum=sum*(h/3)
    simpson_.append(sum)
    print("Approximate integral for n =",n_val,"is ",sum)
    print("\n")


simpson(a,b,n)

plt.plot(n, integral_trap, label='Trapezoidal', color='crimson')
plt.plot(n, simpson_, label='Simpson', color='dodgerblue')
plt.xlabel('n')
plt.ylabel('Approximate Integral')
plt.title('Comparison of Trapezoidal and Simpson Methods')
plt.legend()
plt.show()
```

```
Approximate integral for n = 2 is  0.731370251828563


Approximate integral for n = 4 is  0.7429840978003812


Approximate integral for n = 6 is  0.7451194124361793


Approximate integral for n = 8 is  0.7458656148456951


Approximate integral for n = 10 is  0.7462107961317495


Approximate integral for n = 12 is  0.7463982478934403
```

```
Approximate integral for n = 2 is  0.7471804289095103

Approximate integral for n = 4 is  0.7468553797909873

Approximate integral for n = 6 is  0.7468303914893449

Approximate integral for n = 8 is  0.7468261205274664

Approximate integral for n = 10 is  0.7468249482544436

Approximate integral for n = 12 is  0.7468245263791943
```