Homework 11

Problem 1

In [30]:
```
import numpy as np
import scipy as sc
```

In [56]:
```
def trapezoidal(f, a, b, n):
    h = float(b-a)/n
    result = 0.5*f(a) + 0.5*f(b)
    for i in range(1, n):
        result += f(a + i*h)
    result *= h
    return result

def simps(f, a, b, n):
  sum = 0
  w = (b - a) / n
  for i in range(n + 1):
    x = a + (i * w)
    if x==0:
       continue
    summand = f(x)
    if (i != 0) and (i != n):
      summand *= (2 + (2 * (i % 2)))
    sum += summand
  return ((b - a) / (3 * n)) * sum
#inspired by code found online
```

a)

In [11]:
```
f= lambda x: 1/(1+x**2)
a=-5
b=5
n=100
print(trapezoidal(f,a,b,n))
```

```
2.7467768808079054
```

In [12]:
```
print(simps(f,a,b,n))
```

```
2.7468015268957795
```

b)

In [ ]:
```
#accurate value of integral acheived through
#online integral calculator:

accurate = 2.74680153389
```

```
for i in range(1,n):
    tol = 1e-4
    test = 2*i
    trap_check = False
    simp_check = False
    trap = trapezoidal(f,a,b,test)
    simp = simps(f,a,b,test)
    if (abs(accurate-trap) <= tol) and (trap_check == False):
        print("Iterations required for trapezoidal = ", test)
        print("Trapezoidal approximation error = ", abs(accurate - trap) )
        trap_check = True
    if (abs(accurate-simp)<=tol) and (simp_check == False):
        print("Iterations required for simpsons = ", test)
        print("Simpsons approximation error = ", abs(accurate - simp) )
        simp_check = True
    if test == 2*n-2:
        print("uh oh")
    if trap_check & simp_check:
        break
```

```
Iterations required for simpsons =  32
Simpsons approximation error =  9.08168982540758e-05
Iterations required for simpsons =  34
Simpsons approximation error =  4.758839259011438e-05
Iterations required for simpsons =  36
Simpsons approximation error =  2.607496336315407e-05
Iterations required for simpsons =  38
Simpsons approximation error =  1.33579688084l4e-05
Iterations required for simpsons =  40
Simpsons approximation error =  7.575314374452802e-06
Iterations required for simpsons =  42
Simpsons approximation error =  3.673019449923487e-06
Iterations required for simpsons =  44
Simpsons approximation error =  2.2644333008692286e-06
Iterations required for simpsons =  46
Simpsons approximation error =  9.534706912894819e-07
Iterations required for simpsons =  48
Simpsons approximation error =  7.228677421089458e-07
Iterations required for trapezoidal =  50
Trapezoidal approximation error =  9.859134571188477e-05
Iterations required for simpsons =  50
Simpsons approximation error =  2.0411972823097813e-07
```

So the required iterations for simpsons rule is 32, and for trapezoidal is 50.

c)

In [37]:
```
simp = simps(f,a,b,n)
trap = trapezoidal(f,a,b,n)
scipysVersion4 = sc.integrate.quad(f,a,b, epsrel = 1e-4, full_output=1)
scipysVersion6 = sc.integrate.quad(f,a,b, epsrel = 1e-6, full_output=1)
```

In [49]:
```
print("Simpson = ", simp)
```

```
print("Trapezoidal = ", trap)
print("Scipy's tol - 1e-4 = ", scipysVersion4[0])
print("Scipy's n for tol 1e-4 = ", scipysVersion4[2]["neval"])
print("Scipy's tol - 1e-6 = ", scipysVersion6[0])
print("Scipy's n for tol 1e-6 = ", scipysVersion6[2]["neval"])
```

```
Simpson =  2.7468015268957795
Trapezoidal =  2.7467768808079054
Scipy's tol - 1e-4 =  2.746801533909586
Scipy's n for tol 1e-4 =  63
Scipy's tol - 1e-6 =  2.7468015338900327
Scipy's n for tol 1e-6 =  147
```

This number of iterations is much higher than the two methods outlined above. With the number required for 10^{-4} the same order of magnitude as for Simpsons and Trapezoidal.

Problem 2

Here the problem is estimated.

In [59]:
```
g = lambda t: -np.cos(1/t)*t
a2 = 1
b2 = 0

simp2 = simps(g,a2,b2,5)
print(-simp2)
```

```
-0.017740046481729693
```

This answer is similar to the answer found by an online integral solver, and so seems to be at least at the right order of magnitude and correct to 10^{-2}.

Problem 3

3)
$$I - I_n = \frac{C_1}{n\sqrt{n}} + \frac{C_2}{n^2} + \frac{C_3}{n^2\sqrt{n}} + \frac{C_4}{n^3} + \cdots$$

$$I_n = I - \left(\frac{C_1}{n\sqrt{n}} + \frac{C_2}{n^2} + \frac{C_3}{n^2\sqrt{n}} + \cdots\right)$$

so
$$I_{n/2} = I - \left(2\sqrt{2}\frac{C_1}{n\sqrt{n}} + 4\frac{C_2}{n^2} + 4\sqrt{2}\frac{C_3}{n^2\sqrt{n}} + \cdots\right)$$

$$I_{n/4} = I - \left(8\frac{C_1}{n\sqrt{n}} + 16\frac{C_2}{n^2} + 32\frac{C_3}{n^2\sqrt{n}} + \cdots\right)$$

Want: $I_n = \alpha I_n + \beta I_{n/2} + \gamma I_{n/4}$

so:
$$I_n = I - \left(\frac{C_1}{n\sqrt{n}} + \frac{C_2}{n^2} + O\left(\frac{1}{n^2\sqrt{n}}\right)\right)$$

$$I_{n/2} = I - \left(2\sqrt{2}\frac{C_1}{n\sqrt{n}} + 4\frac{C_2}{n^2} + O\left(\frac{1}{n^2\sqrt{n}}\right)\right)$$

$$I_{n/4} = I - \left(8\frac{C_1}{n\sqrt{n}} + 16\frac{C_2}{n^2} + O\left(\frac{1}{n^2\sqrt{n}}\right)\right)$$

so:
$$\alpha + \beta + \gamma = 1$$
$$\alpha + 2\sqrt{2}\beta + 8\gamma = 0$$
$$\alpha + 4\beta + 16\gamma = 0$$

so:
$$\alpha = 0$$
$$\beta = 4/3$$
$$\gamma = -1/3$$

eq: $I_n = \frac{4}{3} I_{n/2} - \frac{1}{3} I_{n/4}$