# 4th Semester Notes (AG mam)

## Simpson's Method for Integrations

$$I = \int_a^b f(x)dx$$

For Simpson's rule, we divide the interval [a,b] into an even number of sub-intervals.

$$I = \frac{h}{3}[y_0 + 4(y_1 + y_3 + ... + y_{n-1}) + 2(y_2 + y_4 + ... + y_{n-2}) + y_n]$$

In [1]:
```
def f(x):
    return x**2
a,b,n = 1,2,10000  # upper and lower limits
h = (b-a)/n
s = f(a) + f(b)
h1 = 4
for i in range(1,n):
    s += h1*f(a+i*h)
    h1 = 6-h1
sm = s*h/3
print(sm)
```

```
2.3333333333333233
```

In [2]:
```
def f(x):
    return x**2
a,b,n = 1,2,10000  # upper and lower limits
h = (b-a)/n
s = f(a) + f(b)
s1 = 4* sum(f(a+i*h) for i in range(1,n,2))
s2 = 2* sum(f(a+i*h) for i in range(2,n,2))
sm = (s +s1 +s2)*h/3
sm
```

Out[2]:
```
2.3333333333333295
```

In [3]:
```
import numpy as np
def f(x):
    return x**2
a,b,n = 1,2,10000  # upper and lower limits
h = (b-a)/n
x0 = np.arange(a+h,b,2*h)
xe = np.arange(a+2*h,b,2*h)
val = h/3*(f(a) + 4*sum(f(x0)) + 2*sum(f(xe)) + f(b))
print(val)
```

```
2.3333333333331527
```

In [4]:
```
# verification
import sympy as smp
x = smp.symbols('x')
smp.integrate(x**2,(x,1,2)).evalf()
```

Out[4]: $2.33333333333333$

### Discrete values

In [5]:
```
x = [1,2,3,4,5]
y = [4,5,6,7,8]
a, b = 1, 5  # limits of integration
n = len(x)
h = (b-a)/n
s = y[0] + y[n-1]
h1 = 4
for i in range(1,n-1):
    s += h1*y[i]
    h1 = 6-h1
sm = s*h/3
print(sm)
```

```
19.2
```

In [6]:
```
# verification
import sympy as smp
x = smp.symbols('x')
smp.integrate(x+3, (x,1,5))
```

Out[6]: 24

In [7]:
```
import numpy as np
x = np.linspace(0,1,1000)
y = np.linspace(3,4,1000)
h = 0.001
h1 = 4
n = len(y)
s = y[0] + y[n-1]
for i in range(1,n-1):
    s += h1*y[i]
    h1 = 6-h1
s = h*s/3
print(s)
```

```
3.495166833500168
```

In [8]:
```
# verification
import sympy as smp
x = smp.symbols('x')
smp.integrate(x+3, (x,0,1)).evalf()
```

Out[8]: 3.5

In [ ]:

# Additional things

**(modify: code will take n=2,3,4 and give all the results in a single code)**

**(modify: code should give 2 roots. try to do it by using code of Question 8)**

In [9]:
```python
import numpy as np

# Values of x and y
x, y = -5, 12

a, b = float(x), float(y)
z2 = a**2 + b**2
r = (z2)**0.5
tn1 = np.arctan(b/a)
tn2 = tn1/2
rtr = (r)**0.5
sn = np.sin(tn2)
cs = np.cos(tn2)

rl = rtr * cs   # real part
img = rtr * sn  # imaginary part
print(rl, '+ j', img)
```

```
2.9999999999999996 + j -2.0
```

In [10]:
```python
# verification
import sympy as smp
from sympy import *
x = -5 +12*I
smp.sqrt(x).evalf()
```

Out[10]: $2.0 + 3.0i$

## Question-5:

In [ ]:
```python

```

## Question-6:

In [ ]:
```python

```