Numerical Method: Integration

Q.find the integration of 1/(1+x) lowerlim a=0 ,upperlimb=1

1)Simpson's rule

In [1]:
```
import numpy as np
```

In [2]:
```
def f(x):
    return(1/(1+x))
a=0;b=1;n=6;h=(b-a)/n;h
```

Out[2]:
```
0.16666666666666666
```

In [3]:
```
x=np.arange(a,b+0.1,h)
x
```

Out[3]:
```
array([0.        , 0.16666667, 0.33333333, 0.5       , 0.66666667,
       0.83333333, 1.        ])
```

In [4]:
```
y=f(x);y
```

Out[4]:
```
array([1.        , 0.85714286, 0.75      , 0.66666667, 0.6       ,
       0.54545455, 0.5       ])
```

In [5]:
```
s1=(h/2)*(y[0]+y[n]+2*sum(y[np.arange(1,n)]))
s1
```

Out[5]:
```
0.6948773448773449
```

2)Simpson's 1/3rd rule

In [6]:
```
s2=(h/3)*(y[0]+y[n]+4*sum(y[np.arange(1,n,2)])+2*sum(y[np.arange(2,n,2)]))
s2
```

Out[6]:
```
0.6931697931697932
```

3)Simpson's 3/8th rule

In [7]:
```
s3=(h*3/8)*(y[0]+y[n]+3*sum(y[np.arange(1,n)])-sum(y[np.arange(3,n,3)]))
s3
```

Out[7]:
```
0.6931953463203463
```

Iterative method

It is the mathematical procedure that use an initial value to generate a sequence of improving approximate solution.

Q1) f(x)=x^3-x-1=0 find the root where a=1,b=2

1)Bisection Method (finding the root of polynomial)

In [8]:
```
import pandas as pd
```

In [9]:
```
def f(x):
    return(pow(x,3)-x-1)
a=1;b=2;x=(a+b)/2
l=[]
for i in range(5):
    if(f(a)*f(x)>0):
        a=x
    else:
        b=x
    x=(a+b)/2
    l2=[a,b,x,f(a),f(x),f(a)*f(x)]
    l.append(l2)
df=pd.DataFrame(l)
d1=df.rename(columns={0:'a',1:'b',2:'x',3:'f(a)',4:'f(x)',5:'f(a)*f(x)'})
d1
```

Out[9]:

	a	b	x	f(a)	f(x)	f(a)*f(x)
0	1.0000	1.50000	1.250000	-1.000000	-0.296875	0.296875
1	1.2500	1.50000	1.375000	-0.296875	0.224609	-0.066681
2	1.2500	1.37500	1.312500	-0.296875	-0.051514	0.015293
3	1.3125	1.37500	1.343750	-0.051514	0.082611	-0.004256
4	1.3125	1.34375	1.328125	-0.051514	0.014576	-0.000751

Ans: The root of given equation if approximate 1.328125

2) Regula falsi method or linear interpolation or secant method or choid

Q2) f(x)=x^3+x-1=0 find the root where x0=0,x1=1

In [10]:
```
def f(x):
    return(x**3+x-1)
x0=0
x1=1
x2=x1-((x1-x0)/(f(x1)-f(x0)))*f(x1)
l=[]
l1=[x0,x1,f(x0),f(x1),x2]
l.append(l1)
e=1
while(e>0.01):
    e=abs(x1-x2)
    x0=x1
    x1=x2
    x2=x1-((x1-x0)/(f(x1)-f(x0)))*f(x1)
    l2=[x0,x1,f(x0),f(x1),x2]
    l.append(l2)
df=pd.DataFrame(l)
df.rename(columns={0:'x0',1:'x1',2:'f(x0)',3:'f(x1)',4:'x2'})
```

Out[10]:

	x0	x1	f(x0)	f(x1)	x2
0	0.000000	1.000000	-1.000000	1.000000	0.500000
1	1.000000	0.500000	1.000000	-0.375000	0.636364
2	0.500000	0.636364	-0.375000	-0.105935	0.690052
3	0.636364	0.690052	-0.105935	0.018636	0.682020
4	0.690052	0.682020	0.018636	-0.000737	0.682326

Ans:The root of given equation is 0.682326

3)Newton Raphson Method

Q3) f(x)=x^3-x-1=0 find the solution of given equation

In [11]:
```
def f(x):
    return(x**3-x-1)
def f1(x):
    return(3*x**2-1)
x1=1
e=1
x2=x1-(f(x1)/f1(x1))
l=[]
l1=[x1,f(x1),f1(x1),x2]
l.append(l1)
while(e>.01):
    e=abs(x1-x2)
    x1=x2
    x2=x1-(f(x1)/f1(x1))
    l2=[x1,f(x1),f1(x1),x2]
    l.append(l2)
df=pd.DataFrame(l)
df1=df.rename(columns={0:'x1',1:'f(x1)',2:'f1(x1)',3:'x2'})
df1
```

Out[11]:

	x1	f(x1)	f1(x1)	x2
0	1.000000	-1.000000e+00	2.000000	1.500000
1	1.500000	8.750000e-01	5.750000	1.347826
2	1.347826	1.006822e-01	4.449905	1.325200
3	1.325200	2.058362e-03	4.268468	1.324718
4	1.324718	9.243778e-07	4.264635	1.324718

In [12]:
```
sol=df1.x2[4]
round(f(sol),3)
print(sol)
```

```
1.3247179572447898
```

Ans:The solution of given equation is 1.3247179572447898

In [13]:
```
pwd
```

Out[13]:
```
'C:\\Users\\DELL\\AppData\\Local\\Packages\\PythonSoftwareFoundation.Python.3.7_qbz5n2kfra8p0\\LocalCache\\local-packages\\Python37\\Scripts'
```

In [ ]:
```

```