Ratnesh Kumar
GE Practical

Practical : 1 Bisection Method

Question : 1

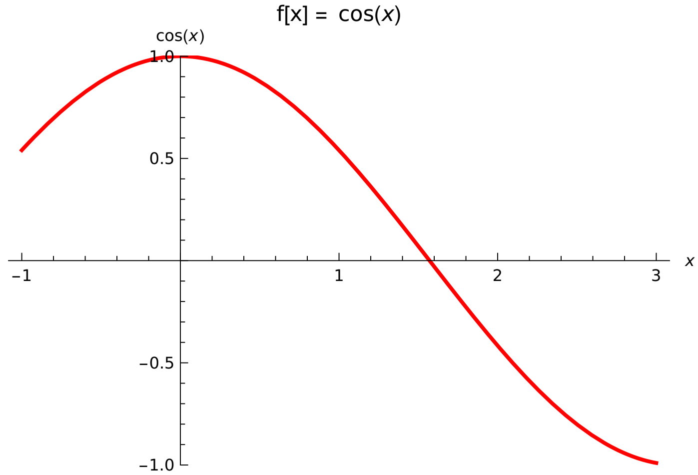
In[158]:=

```
x0 = 0;
x1 = 2.0;
Nmax = 20;
eps = 0.0001;
f[x_] := Cos[x];
If[N[f[x0]* f[x1]]>0,
    Print["Your values do not satisfy the IVP, so change the value."],
For [i=1, i ≤ Nmax, i++,
    m =(x0 + x1)/2;
    If[Abs[(x1-x0)/2]< eps, Return[m],
        Print[i,"th iteration value is :",m];
        Print["Estimated error in ",i," th iteration is : ",(x1 - x0)/2];
    If[f[m]* f[x1] > 0, x1 = m, x0 = m]]];
    Print["Root is: ",m]
    Print["Estimated error in", i," th iteration is : ",(x1 - x0)/2]]
    Plot[f[x],{x,-1,3},
    PlotRange → {-1,1},
    PlotStyle → {Red, Thick},
    PlotLabel → "f[x] = "f [x],
    AxesLabel → {x,f[x]}]
```

```
1th iteration value is :1.
Estimated error in 1 th iteration is : 1.
2th iteration value is :1.5
Estimated error in 2 th iteration is : 0.5
3th iteration value is :1.75
Estimated error in 3 th iteration is : 0.25
4th iteration value is :1.625
Estimated error in 4 th iteration is : 0.125
5th iteration value is :1.5625
Estimated error in 5 th iteration is : 0.0625
6th iteration value is :1.59375
Estimated error in 6 th iteration is : 0.03125
7th iteration value is :1.57813
Estimated error in 7 th iteration is : 0.015625
8th iteration value is :1.57031
Estimated error in 8 th iteration is : 0.0078125
9th iteration value is :1.57422
Estimated error in 9 th iteration is : 0.00390625
10th iteration value is :1.57227
Estimated error in 10 th iteration is : 0.00195313
11th iteration value is :1.57129
Estimated error in 11 th iteration is : 0.000976563
12th iteration value is :1.5708
Estimated error in 12 th iteration is : 0.000488281
13th iteration value is :1.57056
Estimated error in 13 th iteration is : 0.000244141
14th iteration value is :1.57068
Estimated error in 14 th iteration is : 0.00012207
```

Out[6]= 1.57074

Out[7]=



Question : 2

In[165]:=

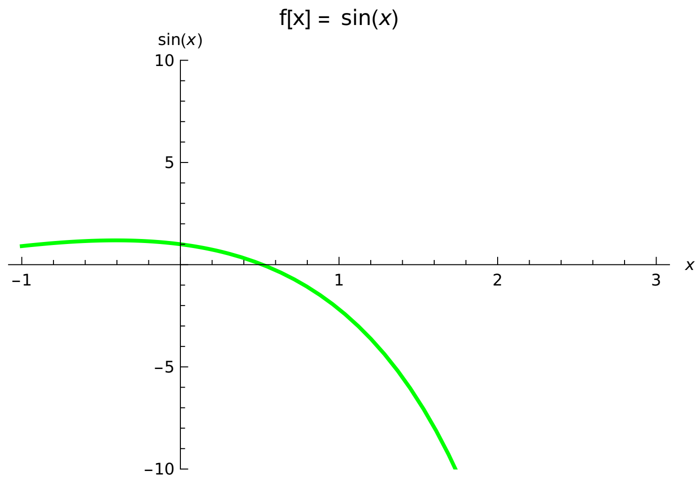
```
x0 = 0;
x1 = 2.0;
Nmax = 20;
eps = 0.00001;
f[x_] := Cos[x]-x(E^x);
If[N[f[x0]* f[x1]]>0,
    Print["Your values do not satisfy the IVP, so change the value."],
For [i=1, i ≤ Nmax, i++,
    m =(x0 + x1)/2;
    If[Abs[(x1-x0)/2]< eps, Return[m],
        Print[i,"th iteration value is :",m];
        Print["Estimated error in ",i," th iteration is : ",(x1 - x0)/2];
    If[f[m]* f[x1] > 0, x1 = m, x0 = m]]];
    Print["Root is: ",m]
    Print["Estimated error in", i," th iteration is : ",(x1 - x0)/2]]
    Plot[f[x],{x,-1,3},
    PlotRange → {-10,10},
    PlotStyle → {Green, Thick},
    PlotLabel → "f[x] = "f [x],
    AxesLabel → {x,f[x]}]
```

```
1th iteration value is :1.
Estimated error in 1 th iteration is : 1.
2th iteration value is :0.5
Estimated error in 2 th iteration is : 0.5
3th iteration value is :0.75
Estimated error in 3 th iteration is : 0.25
4th iteration value is :0.625
Estimated error in 4 th iteration is : 0.125
5th iteration value is :0.5625
Estimated error in 5 th iteration is : 0.0625
6th iteration value is :0.53125
Estimated error in 6 th iteration is : 0.03125
7th iteration value is :0.515625
Estimated error in 7 th iteration is : 0.015625
8th iteration value is :0.523438
Estimated error in 8 th iteration is : 0.0078125
9th iteration value is :0.519531
Estimated error in 9 th iteration is : 0.00390625
10th iteration value is :0.517578
Estimated error in 10 th iteration is : 0.00195313
11th iteration value is :0.518555
Estimated error in 11 th iteration is : 0.000976563
12th iteration value is :0.518066
Estimated error in 12 th iteration is : 0.000488281
13th iteration value is :0.517822
Estimated error in 13 th iteration is : 0.000244141
14th iteration value is :0.5177
Estimated error in 14 th iteration is : 0.00012207
15th iteration value is :0.517761
Estimated error in 15 th iteration is : 0.0000610352
16th iteration value is :0.517731
Estimated error in 16 th iteration is : 0.0000305176
17th iteration value is :0.517746
Estimated error in 17 th iteration is : 0.0000152588
```

Out[170]=

```
0.517754
```

Out[171]=



Question : 3

In[431]:=

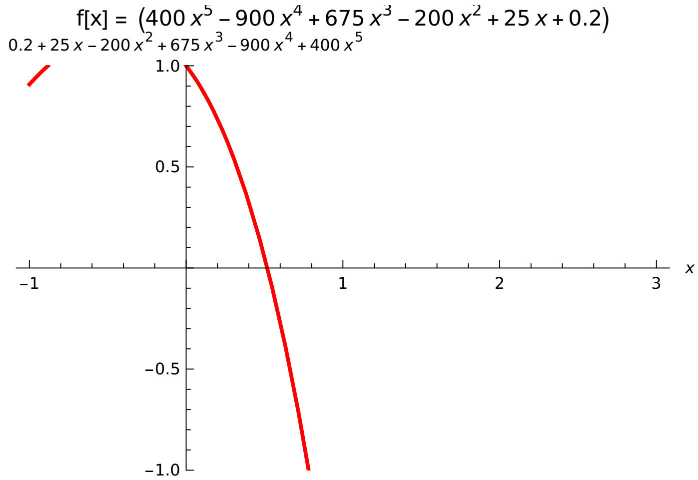
```
x0 = Input["Enter first guess"];
x1 = Input["Enter Second guess"];
Nmax = Input["Enter Nmax guess"];
eps = Input["Enter approx error"];
f[x_] := Cos[x]-x(E^x);
If[N[f[x0]* f[x1]]>0,
    Print["Your values do not satisfy the IVP, so change the value."],
For [i=1, i ≤ Nmax, i++,
    m =(x0 + x1)/2;
    If[Abs[(x1-x0)/2]< eps, Return[m],
        Print[i,"th iteration value is :",m];
        Print["Estimated error in ",i," th iteration is : ",(x1 - x0)/2];
    If[f[m]* f[x1] > 0, x1 = m, x0 = m]]];
    Print["Root is: ",m]
    Print["Estimated error in", i," th iteration is : ",(x1 - x0)/2]]
    Plot[f[x],{x,-1,3},
    PlotRange → {-1,1},
    PlotStyle → {Red, Thick},
    PlotLabel → "f[x] = "f [x],
    AxesLabel → {x,f[x]}]
```

```
1th iteration value is :1.
Estimated error in 1 th iteration is : 1.
2th iteration value is :0.5
Estimated error in 2 th iteration is : 0.5
3th iteration value is :0.75
Estimated error in 3 th iteration is : 0.25
4th iteration value is :0.625
Estimated error in 4 th iteration is : 0.125
5th iteration value is :0.5625
Estimated error in 5 th iteration is : 0.0625
6th iteration value is :0.53125
Estimated error in 6 th iteration is : 0.03125
7th iteration value is :0.515625
Estimated error in 7 th iteration is : 0.015625
8th iteration value is :0.523438
Estimated error in 8 th iteration is : 0.0078125
9th iteration value is :0.519531
Estimated error in 9 th iteration is : 0.00390625
10th iteration value is :0.517578
Estimated error in 10 th iteration is : 0.00195313
11th iteration value is :0.518555
Estimated error in 11 th iteration is : 0.000976563
12th iteration value is :0.518066
Estimated error in 12 th iteration is : 0.000488281
13th iteration value is :0.517822
Estimated error in 13 th iteration is : 0.000244141
14th iteration value is :0.5177
Estimated error in 14 th iteration is : 0.00012207
```

Out[436]=

```
0.517761
```

Out[437]=



Secant Method

Question : 1

In[179]:=

```
(*x0 = Input["Enter first guess"];
x1 = Input["Enter Second guess"];
Nmax = Input["Enter Nmax guess"];
eps = Input["Enter approx error"];
f[x_] = Input["Enter Function error"];*)
x0 = 0;
x1 = 1.0;
Nmax = 20;
eps = 0.00001;
f[x_] := Cos[x];
For [i=1, i ≤ Nmax, i++,
    x2 =x1-((f[x1]*(x1 - x0))/(f[x1]- f[x0]));
    If[Abs[(x1-x2)]/2 < eps, Return[x2],x0=x1;x1=x2];
        Print[i,"th iteration value is :",x2];
        Print["Estimated error in ",i," th iteration is : ",Abs[x1 - x0]]];
    Print["Root is :",x2];
    Print["Estimated error in ",Abs[x2 - x1]];
    Plot[f[x],{x,-1,3},
    PlotRange → {-2,2},
    PlotStyle → {Red, Thick},
    PlotLabel → "f[x] = "f [x],
    AxesLabel → {x,f[x]}]
```

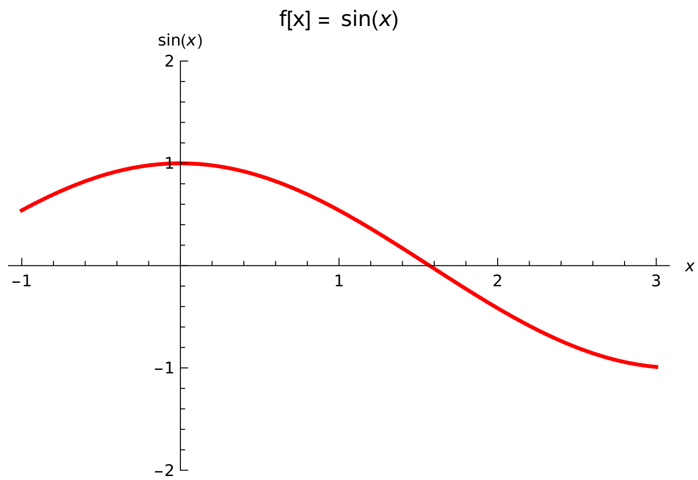
```
1th iteration value is :2.17534
Estimated error in 1 th iteration is : 1.17534
2th iteration value is :1.57278
Estimated error in 2 th iteration is : 0.602559
3th iteration value is :1.57067
Estimated error in 3 th iteration is : 0.00211435
4th iteration value is :1.5708
Estimated error in 4 th iteration is : 0.000126873
```

Out[184]=

```
1.5708
Root is :1.5708
```

Estimated error in 7.81941×10^{-11}

Out[187]=



Question : 2

In[188]:=

```
x0 = 0;
x1 = 1.0;
Nmax = 20;
eps = 0.0001;
f[x_] := Cos[x]-x(E^x);
For [i=1, i ≤ Nmax, i++,
    x2 =x1-((f[x1]*(x1 - x0))/(f[x1]- f[x0]));
    If[Abs[(x1-x2)]/2 < eps, Return[x2],x0=x1;x1=x2];
        Print[i,"th iteration value is :",x2];
        Print["Estimated error in ",i," th iteration is : ",Abs[x1 - x0]]];
    Print["Root is :",x2];
    Print["Estimated error in ",Abs[x2 - x1]];
    Plot[f[x],{x,-1,3},
    PlotRange → {-2,2},
    PlotStyle → {Red, Thick},
    PlotLabel → "f[x] = "f [x],
    AxesLabel → {x,f[x]}]
```

```
1th iteration value is :0.314665
Estimated error in 1 th iteration is : 0.685335
2th iteration value is :0.446728
Estimated error in 2 th iteration is : 0.132063
3th iteration value is :0.531706
Estimated error in 3 th iteration is : 0.0849777
4th iteration value is :0.516904
Estimated error in 4 th iteration is : 0.0148014
5th iteration value is :0.517747
Estimated error in 5 th iteration is : 0.000842998
```

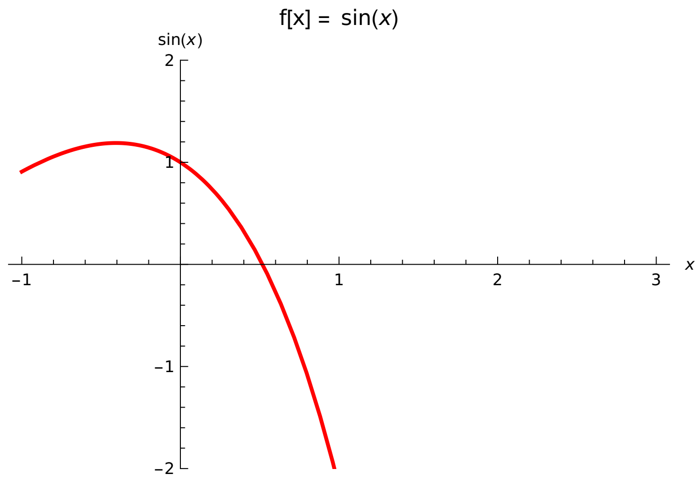
Out[193]=

0.517757

Root is :0.517757

Estimated error in 9.90548×10^{-6}

Out[196]=



Regular Falsi

Question : 1

In[197]:=

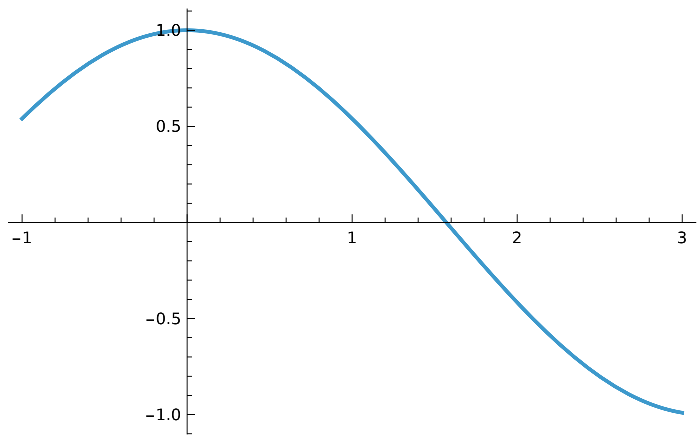
```
x0 = 0;
x1 = 2.0;
Nmax = 20;
eps = 0.0001;
f[x_] := Cos[x];
If[N[f[x0]]*N[f[x1]]> 0,
    Print["These values do not satisfy the IVP so change the value."],
    For [i=1, i ≤ Nmax, i++,
        x2 = N[x1-f[x1]*(x1 - x0)/(f[x1]- f[x0])];
        If [Abs[x1-x0]<eps,Return[N[x2]],
            Print[i,"th iterations value is: ", N[x2]];
            Print["Estimated error in ",i," th iteration is : ",N[x1 - x0]]];
    If[f[x2]*f[x1]>0,x1=x2,x0=x2]];
        Print["Root is :",N[x2]];
        Print["Estimated error in ",i," th iteration is : ",N[x1 - x0]];
        If[N[f[x0]]*N[f[x1]]< 0,Plot[f[x],{x,-1,3}]]
```

```
1th iterations value is: 1.41228
Estimated error in 1 th iteration is : 2.
2th iterations value is: 1.57391
Estimated error in 2 th iteration is : 0.587717
3th iterations value is: 1.57078
Estimated error in 3 th iteration is : 0.161623
4th iterations value is: 1.5708
Estimated error in 4 th iteration is : 0.0031228
```

Out[202]=

```
1.5708
```

Out[203]=



Question : 2

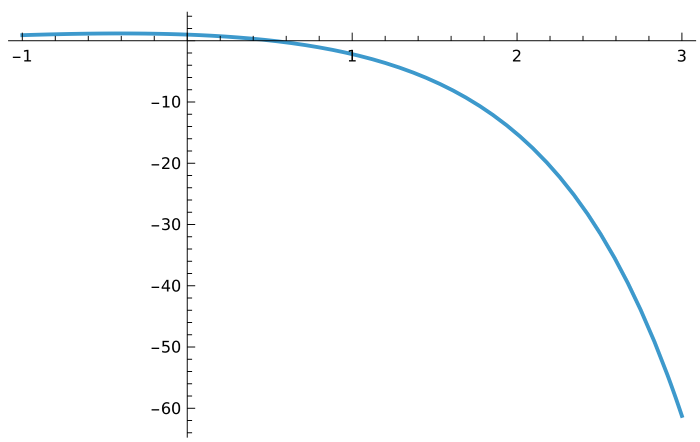
In[204]:=

```
x0 = 0;
x1 = 1.0;
Nmax = 20;
eps = 0.0001;
f[x_] := Cos[x]-x(E^x) ;
If[N[f[x0]]*N[f[x1]]> 0,
    Print["These values do not satisfy the IVP so change the value."],
    For [i=1, i ≤ Nmax, i++,
        x2 = N[x1-f[x1]*(x1 - x0)/(f[x1]- f[x0])];
        If [Abs[x1-x0]<eps,Return[N[x2]],
            Print[i,"th iterations value is: ", N[x2]];
            Print["Estimated error in ",i," th iteration is : ",N[x1 - x0]]];
    If[f[x2]*f[x1]>0,x1=x2,x0=x2]];
        Print["Root is :",N[x2]];
        Print["Estimated error in ",i," th iteration is : ",N[x1 - x0]]];
        If[N[f[x0]]*N[f[x1]]< 0,Plot[f[x],{x,-1,3}]]
```

```
1th iterations value is: 0.314665
Estimated error in 1 th iteration is : 1.
2th iterations value is: 0.446728
Estimated error in 2 th iteration is : 0.685335
3th iterations value is: 0.494015
Estimated error in 3 th iteration is : 0.553272
4th iterations value is: 0.509946
Estimated error in 4 th iteration is : 0.505985
5th iterations value is: 0.515201
Estimated error in 5 th iteration is : 0.490054
6th iterations value is: 0.516922
Estimated error in 6 th iteration is : 0.484799
7th iterations value is: 0.517485
Estimated error in 7 th iteration is : 0.483078
8th iterations value is: 0.517668
Estimated error in 8 th iteration is : 0.482515
9th iterations value is: 0.517728
Estimated error in 9 th iteration is : 0.482332
10th iterations value is: 0.517748
Estimated error in 10 th iteration is : 0.482272
```

```
11th iterations value is: 0.517754
Estimated error in 11 th iteration is : 0.482252
12th iterations value is: 0.517756
Estimated error in 12 th iteration is : 0.482246
13th iterations value is: 0.517757
Estimated error in 13 th iteration is : 0.482244
14th iterations value is: 0.517757
Estimated error in 14 th iteration is : 0.482243
15th iterations value is: 0.517757
Estimated error in 15 th iteration is : 0.482243
16th iterations value is: 0.517757
Estimated error in 16 th iteration is : 0.482243
17th iterations value is: 0.517757
Estimated error in 17 th iteration is : 0.482243
18th iterations value is: 0.517757
Estimated error in 18 th iteration is : 0.482243
19th iterations value is: 0.517757
Estimated error in 19 th iteration is : 0.482243
20th iterations value is: 0.517757
Estimated error in 20 th iteration is : 0.482243
Root is :0.517757
Estimated error in 21 th iteration is : 0.482243
```

Out[210]=



Newton-Raphson Method

Question : 1

In[211]:=

```
x0 = 1;
Nmax = 20;
eps = 0.0001;
f[x_] := Cos[x];
For [i=1, i ≤ Nmax, i++,
    x1 = N[x0-(f[x]/.x→ x0)/(D[f[x],x]/.x→ x0)];
    If [Abs[x1-x0]< eps, Return[x1],x0p=x0;x0=x1];
    Print["In ",i,"th Number of iterations the approximation to root is:", x1];
    Print["Estimated error in ",Abs[x1 - x0p]]];
Print["The Final approximation of root is:", x1];
Print["Estimated error in ",Abs[x1 - x0]];
Plot[f[x],{x,-1,3}]
```

In 1th Number of iterations the approximation to root is:-0.557408

Estimated error in 1.55741

In 2th Number of iterations the approximation to root is:0.0659365

Estimated error in 0.623344

In 3th Number of iterations the approximation to root is:-0.0000957219

Estimated error in 0.0660322

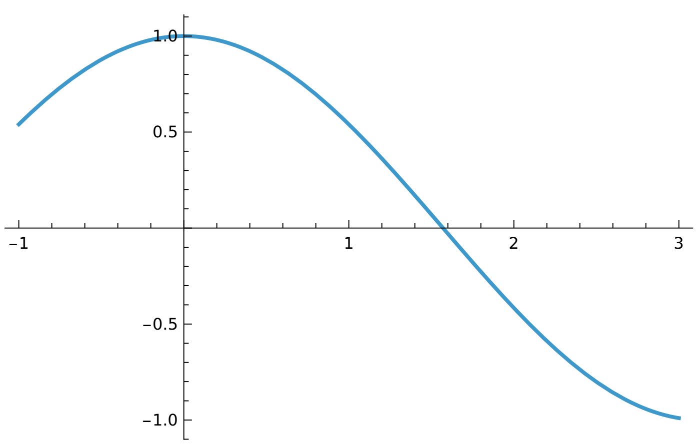
Out[215]=

2.92357×10^{-13}

The Final approximation of root is:2.92357×10^{-13}

Estimated error in 0.0000957219

Out[218]=



In[219]:=

Question : 2

Out[219]=

Question : 2

In[220]:=

```
x0 = 0.5;
Nmax = 20;
eps = 0.0001;
f[x_] := x^3-5*x+1;
For [i=1, i ≤ Nmax, i++,
    x1 = N[x0-(f[x]/.x→ x0)/(D[f[x],x]/.x→ x0)];
    If [Abs[x1-x0]< eps, Return[x1],x0p=x0;x0=x1];
    Print["In ",i,"th Number of iterations the approximation to root is:", x1];
    Print["Estimated error in ",Abs[x1 - x0p]]];
Print["The Final approximation of root is:", x1];
Print["Estimated error in ",Abs[x1 - x0]];
Plot[f[x],{x,-1,3}]
```

In 1th Number of iterations the approximation to root is:-0.0463025

Estimated error in 0.546302

In 2th Number of iterations the approximation to root is:0.000033118

Estimated error in 0.0463356

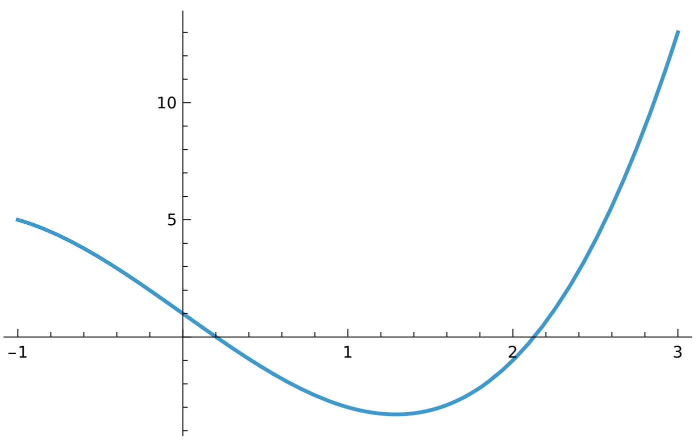
Out[224]=

-1.2108×10^{-14}

The Final approximation of root is:-1.2108×10^{-14}

Estimated error in 0.000033118

Out[227]=



Jacobi Method

Question : 1

In[228]:=

```
GaussJacobi[A0_, b0_, x0_, maxiter_] :=
 Module[{A = N[A0], b = N[b0], xk = x0, xk1, i, j, k = 0, n, m,
   OutputDetails},
  size = Dimensions[A];
  n = size[[1]];
  m = size[[2]];
  If[n ≠ m,
   Print["Not a square matrix, cannot proceed with Gauss-Jacobi method"];
   Return[]
   ];
  OutputDetails = {xk};
  xk1 = Table[0, {n}];
  While[k < maxiter,
   For[i = 1, i ≤ n, i++,
    xk1[[i]] = (1/A[[i, i]])*(b[[i]] -
        Sum[A[[i, j]]*xk[[j]], {j, 1, i - 1}] -
        Sum[A[[i, j]]*xk[[j]], {j, i + 1, n}])
    ];
   k++;
   OutputDetails = Append[OutputDetails, xk1];
   xk = xk1;
   ];
  colHeading = Table[X[s], {s, 1, n}];
  Print[NumberForm[
    TableForm[OutputDetails,
     TableHeadings → {None, colHeading}], 6]];
  Print["No. of iterations performed: ", maxiter];
  ];
A = {{5, 1, 2}, {-3, 9, 4}, {1, 2, -7}};
b = {10, -14, -33};
X0 = {0, 0, 0};
GaussJacobi[A, b, X0, 15]
```

X[1]	X[2]	X[3]
0	0	0
2.	-1.55556	4.71429
0.425397	-2.98413	4.55556
0.774603	-3.43845	3.92245
1.11871	-3.04067	3.84253
1.07112	-2.89044	4.00534
0.975953	-2.97867	4.04146
0.979148	-3.02644	4.00266
1.00422	-3.00813	3.98947
1.00584	-2.99391	3.99828
0.99947	-2.99729	4.00257
0.998428	-3.00132	4.0007
0.999985	-3.00083	3.9994
1.00041	-2.99974	3.99976
1.00004	-2.99976	4.00013
0.999898	-3.00004	4.00008

No. of iterations performed: 15

Question : 2

In[233]:=

```
GaussJacobi[A0_, b0_, x0_, maxiter_] :=
 Module[{A = N[A0], b = N[b0], xk = x0, xk1, i, j, k = 0, n, m,
   OutputDetails},
  size = Dimensions[A];
  n = size[[1]];
  m = size[[2]];
  If[n ≠ m,
   Print["Not a square matrix, cannot proceed with Gauss-Jacobi method"];
   Return[]
   ];
  OutputDetails = {xk};
  xk1 = Table[0, {n}];
  While[k < maxiter,
   For[i = 1, i ≤ n, i++,
    xk1[[i]] = (1/A[[i, i]])*(b[[i]] -
        Sum[A[[i, j]]*xk[[j]], {j, 1, i - 1}] -
        Sum[A[[i, j]]*xk[[j]], {j, i + 1, n}])
    ];
   k++;
   OutputDetails = Append[OutputDetails, xk1];
   xk = xk1;
   ];
  colHeading = Table[X[s], {s, 1, n}];
  Print[NumberForm[
    TableForm[OutputDetails,
     TableHeadings → {None, colHeading}], 6]];
  Print["No. of iterations performed: ", maxiter];
  ];
A = {{4, 1, 1}, {1, 5, 2}, {1, 2, 3}};
b = {2, -6, -4};
X0 = {0.5, -0.5, -0.5};
GaussJacobi[A, b, X0, 15]
```

$x_{[1]}$	$x_{[2]}$	$x_{[3]}$
0.5	-0.5	-0.5
0.75	-1.1	-1.16667
1.06667	-0.883333	-0.85
0.933333	-1.07333	-1.1
1.04333	-0.946667	-0.928889
0.968889	-1.03711	-1.05
1.02178	-0.973778	-0.964889
0.984667	-1.0184	-1.02474
1.01079	-0.987037	-0.982622
0.992415	-1.00911	-1.01224
1.00534	-0.993588	-0.9914
0.996247	-1.00451	-1.00605
1.00264	-0.996828	-0.995744
0.998143	-1.00223	-1.00299
1.00131	-0.998431	-0.997894
0.999081	-1.0011	-1.00148

No. of iterations performed: 15

Question : 3

In[238]:=

```
GaussJacobiMatrixForm[A0_, b0_, x0_, maxiter_] :=
  Module[{A = N[A0], b = N[b0], xk = x0, k = 0, D, R, Dinv,
    OutputDetails}, D = DiagonalMatrix[Diagonal[A]]; R = A - D;
   Dinv = Inverse[D]; OutputDetails = {xk};
   While[k < maxiter, xk = Dinv.(b - R.xk);
    OutputDetails = Append[OutputDetails, xk];
    k++;];
   colHeading = Table[Subscript[x, s], {s, 1, Length[x0]}];
   Print[NumberForm[
     TableForm[OutputDetails, TableHeadings → {None, colHeading}],
     6]];
   Print["No. of iterations performed: ", maxiter];];
A = {{5, 1, 2}, {-3, 9, 4}, {1, 2, -7}};
b = {10, -14, -33};
X0 = {0, 0, 0};

GaussJacobiMatrixForm[A, b, X0, 15]
```

x_1	x_2	x_3
0	0	0
2.	-1.55556	4.71429
0.425397	-2.98413	4.55556
0.774603	-3.43845	3.92245
1.11871	-3.04067	3.84253
1.07112	-2.89044	4.00534
0.975953	-2.97867	4.04146
0.979148	-3.02644	4.00266
1.00422	-3.00813	3.98947
1.00584	-2.99391	3.99828
0.99947	-2.99729	4.00257
0.998428	-3.00132	4.0007
0.999985	-3.00083	3.9994
1.00041	-2.99974	3.99976
1.00004	-2.99976	4.00013
0.999898	-3.00004	4.00008

No. of iterations performed: 15

Gauss Seidel Method

Question : 1

In[243]:=

```
GaussSeidel[A0_, b0_, x0_, maxiter_] :=
  Module[{A = N[A0], b = N[b0], xk = x0, xk1, i, j, k = 0, n, m,
    OutputDetails, size, colHeading}, size = Dimensions[A];
   n = size[[1]];
   m = size[[2]];
   If[n ≠ m,
    Print["Not a square matrix, cannot proceed with Gauss-Seidel method"];
    Return[]];
   OutputDetails = {xk};
   xk1 = Table[0, {n}];
   While[k < maxiter,
    For[i = 1, i ≤ n, i++,
     xk1[[i]] = (1/A[[i, i]])*(b[[i]] -
         Sum[A[[i, j]]*xk1[[j]], {j, 1, i - 1}] -
         Sum[A[[i, j]]*xk[[j]], {j, i + 1, n}])];
    xk = xk1; OutputDetails = Append[OutputDetails, xk];
    k++;];
   colHeading = Table[Subscript[x, s], {s, 1, n}];
   Print[NumberForm[
     TableForm[OutputDetails, TableHeadings → {None, colHeading}],
     6]];
   Print["No. of iterations performed: ", maxiter];];
A = {{5, 1, 2}, {-3, 9, 4}, {1, 2, -7}};
b = {10, -14, -33};
X0 = {0, 0, 0};
GaussSeidel[A, b, X0, 15];
```

x_1	x_2	x_3
0	0	0
2.	-0.888889	4.74603
0.279365	-3.57178	3.73369
1.22088	-2.80801	4.08641
0.927039	-3.06272	3.97166
1.02388	-2.97944	4.00929
0.992174	-3.00674	3.99696
1.00256	-2.99779	4.001
0.99916	-3.00072	3.99967
1.00028	-2.99976	4.00011
0.99991	-3.00008	3.99996
1.00003	-2.99997	4.00001
0.99999	-3.00001	4.
1.	-3.	4.
0.999999	-3.	4.
1.	-3.	4.

No. of iterations performed: 15

Question : 2

In[248]:=

```
GaussSeidelMatrixForm[A0_, b0_, x0_, maxiter_] :=
  Module[{A = N[A0], b = N[b0], xk = x0, k = 0, D, L, U, DLinv,
    OutputDetails}, D = DiagonalMatrix[Diagonal[A]];
   L = LowerTriangularize[A, -1];
   U = UpperTriangularize[A, 1];
   DLinv = Inverse[D + L];
   OutputDetails = {xk};
   While[k < maxiter, xk = -DLinv.U.xk + DLinv.b;
    OutputDetails = Append[OutputDetails, xk];
    k++;];
   colHeading = Table[Subscript[x, s], {s, 1, Length[x0]}];
   Print[NumberForm[
     TableForm[OutputDetails, TableHeadings → {None, colHeading}],
     6]];
   Print["No. of iterations performed: ", maxiter];];
A = {{5, 1, 2}, {-3, 9, 4}, {1, 2, -7}};
b = {10, -14, -33};
X0 = {0, 0, 0};
GaussSeidelMatrixForm[A, b, X0, 15]
```

x_1	x_2	x_3
0	0	0
2.	-0.888889	4.74603
0.279365	-3.57178	3.73369
1.22088	-2.80801	4.08641
0.927039	-3.06272	3.97166
1.02388	-2.97944	4.00929
0.992174	-3.00674	3.99696
1.00256	-2.99779	4.001
0.99916	-3.00072	3.99967
1.00028	-2.99976	4.00011
0.99991	-3.00008	3.99996
1.00003	-2.99997	4.00001
0.99999	-3.00001	4.
1.	-3.	4.
0.999999	-3.	4.
1.	-3.	4.

No. of iterations performed: 15

Lagrange Interpolation

Question : 1

In[421]:=

```
LagrangePolynomial[x0_, f0_] :=
 Module[{xi = x0, fi = f0, n, m, polynomial},
  n = Length[xi];
  m = Length[fi];
  If[n ≠ m,
   Print["List of points and function's values are not of same size"];
   Return[];];
  For[i = 1, i ≤ n, i++,
   L[i, x_] = (Product[(x - xi[[j]])/(xi[[i]] - xi[[j]]), {j, 1, i - 1}]) *
               (Product[(x - xi[[j]])/(xi[[i]] - xi[[j]]), {j, i + 1, n}]);];
  polynomial[x_] := Sum[L[k, x]*fi[[k]], {k, 1, n}];
  Return[polynomial[x]];]
nodes = {0, 1, 3};
values = {1, 3, 55};
lagrangePolynomial[x_] = LagrangePolynomial[nodes, values]
Expand[%]
nodes = {1, 3, 5, 7, 9};
values = {N[Log[1]], N[Log[3]], N[Log[5]], N[Log[7]], N[Log[9]]};
lagrangePolynomial[x_] = LagrangePolynomial[nodes, values]
Simplify[%]
Plot[{lagrangePolynomial[x], Log[x]}, {x,1,10}, Ticks→ {Range[0,10]}, PlotLegends → "Express
```

Out[424]=

$$\frac{1}{3}(1-x)(3-x)+\frac{3}{2}(3-x)x+\frac{55}{6}(-1+x)x$$

Out[425]=

$$1-6x+8x^2$$

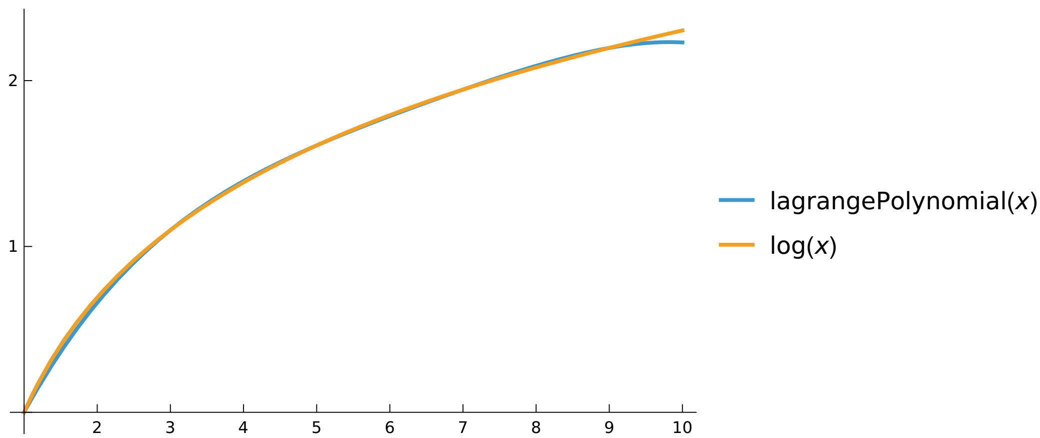
Out[428]=

$$0.+0.0114439(5-x)(7-x)(9-x)(-1+x)+0.0251475(7-x)(9-x)(-3+x)(-1+x)+0.0202699(9-x)(-5+x)(-3+x)(-1+x)+0.00572194(-7+x)(-5+x)(-3+x)(-1+x)$$

Out[429]=

$$-0.987583+1.18991x-0.223608x^2+0.0221231x^3-0.000844369x^4$$

Out[430]=



Newton Interpolation

In[115]:=

```
newtonDividedDifference[x_List, y_List] :=
 Module[{n = Length[x], dd, i, j},dd = Table[0, {n}, {n}];
  Do[dd[[i, 1]] = y[[i]],{i, 1, n}];
  For[j = 2, j ≤ n, j++, For[i = j, i ≤ n, i++,
    dd[[i, j]] = (dd[[i, j - 1]] - dd[[i - 1, j - 1]])/(x[[i]] - x[[i - j + 1]]);];];dd]



newtonPolynomial[x_List, y_List, var_Symbol] :=
 Module[{dd = newtonDividedDifference[x, y], n = Length[x], poly},poly = dd[[1, 1]];
  Do[poly = poly + dd[[i, i]] * Product[var - x[[k]], {k, 1, i - 1}],{i, 2, n}];
  Expand[poly]]



xVals = {0.5, 1.5, 3, 5, 6.5, 8};
yVals = {1.625, 5.875, 31, 131, 282.125, 521};
P = newtonPolynomial[xVals, yVals, x]
f7 = P /. x → 7
```

Out[119]=

$1. + 1.\,x + 1.\,x^3$

Out[120]=

351.

Question : 1

In[133]:=

```
NDD[x0_, f0_, startindex_, endindex_] :=
 Module[{x = x0, f = f0, i = startindex, j = endindex, answer},
  If[i == j,Return[f[[i]]],answer = (NDD[x, f, i + 1, j] - NDD[x, f, i, j - 1])/(x[[j]] - x[[i]]);
   Return[answer]];];
x = {0.5, 1.5, 3, 5, 6.5, 8};
f = {1.625, 5.875, 31, 131, 282.125, 521};
NDD[x, f, 1, 2]
```

Out[136]=

4.25

Question : 2

In[137]:=

```
NDDP[x0_, f0_] := Module[{x1 = x0, f = f0, n, newtonPolynomial, k, j},
  n = Length[x1];
  newtonPolynomial[y_] = 0;
  For[i = 1, i ≤ n, i++, prod[y_] = 1;
   For[k = 1, k ≤ i - 1, k++, prod[y_] = prod[y] * (y - x1[[k]])];
   newtonPolynomial[y_] =
    newtonPolynomial[y] + NDD[x1, f, 1, i] * prod[y]];
  Return[newtonPolynomial[y]];];
nodes = {0, 1, 3};
values = {1, 3, 55};
NDDP[nodes, values]
```

Out[140]=

$1 + 2 y + 8 (-1 + y) y$

Trapezoidal Rule Method

Question : 1

In[376]:=

```
(*a = Input[];
b = Input[];
n = Input[];*)
a = 1;
b = 2;
n = 10;
h = (b - a)/n;
y = Table[a + i*h, {i, 1, n}];
f[x] := Log[x];
sumodd = 0;
sumeven = 0;
For[i = 1, i < n, i += 2, sumodd += 2*f[x] /. x → y[[i]]];
For[i = 2, i < n, i += 2, sumodd += 2*f[x] /. x → y[[i]]];
Tn = (h/2)*((f[x] /. x → a) + N[sumodd] +
     N[sumeven] + (f[x] /. x → b));
Print["For n=", n, ",Trapezoidal estimate is:", Tn]
in = Integrate[Log[x], {x, 4, 5.2}];
Print["True value is ", in]
Print["Absolute error is ", Abs[Tn - in]]
```

```
For n=10,Trapezoidal estimate is:0.385878
True value is 1.82785
Absolute error is 1.44197
```

Question : 2

In[406]:=

```
a = Input["Enter the left end point"];
b = Input["Enter the right end point"];
n = Input["Enter the number of sub intervals to be formed"];
h = (b - a)/n;
y = Table[a + i*h, {i, 1, n}];
f[x] := 0.2+25*x-200*x^2+675*x^3-900*x^4+400*x^5;
sumodd = 0;
sumeven = 0;
For[i = 1, i < n, i += 2, sumodd += 2*f[x] /. x → y[[i]]];
For[i = 2, i < n, i += 2, sumodd += 2*f[x] /. x → y[[i]]];
Tn = (h/2)*((f[x] /. x → a) + N[sumodd] +
     N[sumeven] + (f[x] /. x → b));
Print["For n=", n, ",Trapezoidal estimate is:", Tn]
in = Integrate[Sin[x], {x, 0, Pi/2}];
Print["True value is ", in]
Print["Absolute error is ", Abs[Tn - in]]
```

```
For n=8,Trapezoidal estimate is:1.6008
True value is 1
Absolute error is 0.6008
```

Euler Method

Question : 1

In[293]:=

```
EulerMethod[a0_, b0_, n0_, f_, alpha_] :=
  Module[{a = a0, b = b0, n = n0, h, ti}, h = (b - a)/n;
   ti = Table[a + (j - 1) h, {j, 1, n + 1}];
   ui = Table[0, {n + 1}];
   ui[[1]] = alpha;
   OutputDetails = {{0, ti[[1]], alpha}};
   For[i = 1, i ≤ n, i++,
    ui[[i + 1]] = ui[[i]] + h*f[ti[[i]], ui[[i]]];
    OutputDetails =
     Append[OutputDetails, {i, N[ti[[i + 1]]], N[ui[[i + 1]]]}];];
   Print[NumberForm[
     TableForm[OutputDetails,
      TableHeadings → {None, {"i", "ti", "ui"}}], 6]];
   Print["Subinterval size h used= ", h];];
f[t_, w_] := 1 + w/t;
a = 1;
b = 6;
n = 10;
alpha = 1;
EulerMethod[a, b, 10, f, alpha];
```

i	ti	ui
0	1	1
1	1.5	2.
2	2.	3.16667
3	2.5	4.45833
4	3.	5.85
5	3.5	7.325
6	4.	8.87143
7	4.5	10.4804
8	5.	12.1448
9	5.5	13.8593
10	6.	15.6193

Subinterval size h used= $\frac{1}{2}$

Question : 2

In[300]:=

```
EulerMethodwithH[a0_, b0_, h0_, f_, alpha_] :=
  Module[{a = a0, b = b0, h = h0, n, ti},
   n = (b - a)/h;
   ti = Table[a + (j - 1) h, {j, 1, n + 1}];
   ui = Table[0, {n + 1}];
   ui[[1]] = alpha;
   OutputDetails = {{0, ti[[1]], alpha}};
   For[i = 1, i ≤ n, i++,
    ui[[i + 1]] = ui[[i]] + h*f[ti[[i]], ui[[i]]];
    OutputDetails = Append[OutputDetails,
      {i, N[ti[[i + 1]]], N[ui[[i + 1]]]}];];
   Print[NumberForm[
     TableForm[OutputDetails,
      TableHeadings → {None, {"i", "ti", "ui"}}], 6]];
   Print["Subinterval size h used= ", h];];
g[t_, w_] := 1 + w/t;
a = 1;
b = 6;
h = .2;
alpha = 1;
EulerMethodwithH[a, b, h, g, alpha];
```

i	ti	ui
0	1.	1
1	1.2	1.4
2	1.4	1.83333
3	1.6	2.29524
4	1.8	2.78214
5	2.	3.29127
6	2.2	3.8204
7	2.4	4.36771
8	2.6	4.93168
9	2.8	5.51104
10	3.	6.10469
11	3.2	6.71167
12	3.4	7.33115
13	3.6	7.96239
14	3.8	8.60474
15	4.	9.25763
16	4.2	9.92051
17	4.4	10.5929
18	4.6	11.2744
19	4.8	11.9646
20	5.	12.6631
21	5.2	13.3696
22	5.4	14.0839
23	5.6	14.8055
24	5.8	15.5343
25	6.	16.2699

Subinterval size h used= 0.2

2nd Order Runge Kutta Method (Modified Euler Method)

Question : 1

In[307]:=

```
ModifiedEulerMethod[a0_, b0_, n0_, f_, alpha_, actualSolution_] :=
 Module[{a=a0, b=b0, n=n0, h, ti, K1, K2},
 h = (b-a)/n;
 ti=Table[a+(j-1) h,{j,1,n+1}];
wi=Table[0,{n+1}];wi[[1]] =alpha;
actualSol = actualSolution[ti[[1]]];
difference = Abs[actualSol - wi[[1]]];
OutputDetails={{0,ti[[1]],alpha,actualSol, difference}};
For[i=1,i≤n,i++,
K1 = h f[ti[[i]],wi[[i]]];
K2 = h f[ti[[i]] +h/2,wi[[i]]+ K1/2];
wi[[i+1]] =wi[[i]]+K2;
actualSol = actualSolution[ti[[i+1]]];
difference =Abs[actualSol - wi[[i+1]]];
OutputDetails=Append[OutputDetails,{i,N[ti[[i+1]]],N[wi[[i+1]]],N[actualSol],N[difference]}];];
Print[NumberForm[TableForm[OutputDetails,TableHeadings→{None,{"i","ti","wi","actSol(ti)",'
f[t_,x_] := 1+x/t;
actualSolution[t_]:=t(1+Log[t]);
ModifiedEulerMethod[1, 6, 5,f,1,actualSolution]
```

i	ti	wi	actSol(ti)	Abs(wi-actSol(ti))
0	1	1	1	0
1	2.	3.33333	3.38629	0.052961
2	3.	6.2	6.29584	0.0958369
3	4.	9.40952	9.54518	0.135654
4	5.	12.873	13.0472	0.174174
5	6.	16.5385	16.7506	0.212029

2nd Order Runge Kutta Method (Heun Method)

Question : 1

In[312]:=

```
HeunMethod[a0_, b0_, n0_, f_, alpha_, actualSolution_] :=
 Module[{a=a0, b=b0, n=n0, h, ti, K1, K2},
 h = (b-a)/n;
 ti=Table[a+(j-1) h,{j,1,n+1}];
wi=Table[0,{n+1}];wi[[1]] =alpha;
actualSol = actualSolution[ti[[1]]];
difference = Abs[actualSol - wi[[1]]];
OutputDetails={{0,ti[[1]],alpha,actualSol, difference}};
For[i=1,i≤n,i++,
K1 = (h/2) * f[ti[[i]],wi[[i]]];
K2 = (h/2) * f[ti[[i]] +h,wi[[i]]+ (2*K1)];
wi[[i+1]] =wi[[i]]+K1+K2;
actualSol = actualSolution[ti[[i+1]]];
difference =Abs[actualSol - wi[[i+1]]];
OutputDetails=Append[OutputDetails,{i,N[ti[[i+1]]],N[wi[[i+1]]],N[actualSol],N[difference]}];];
Print[NumberForm[TableForm[OutputDetails,TableHeadings→{None,{"i","ti","wi","actSol(ti)",'
f[t_,x_] := 1+x/t;
actualSolution[t_]:=t(1+Log[t]);
HeunMethod[1, 6, 10,f,1,actualSolution]
```

i	ti	wi	actSol(ti)	Abs(wi-actSol(ti))
0	1	1	1	0
1	1.5	2.08333	2.1082	0.0248643
2	2.	3.34028	3.38629	0.0460166
3	2.5	4.72535	4.79073	0.0653796
4	3.	6.21208	6.29584	0.0837535
5	3.5	7.78314	7.88467	0.101526
6	4.	9.42627	9.54518	0.118905
7	4.5	11.1323	11.2683	0.136014
8	5.	12.8943	13.0472	0.152929
9	5.5	14.7064	14.8761	0.169701
10	6.	16.5642	16.7506	0.186363

Simpson's Rule

Question : 1

```
Simpson[f_, a_, b_, n_] := Module[{h = (b - a)/n},
  If[OddQ[n], Return["n must be even"]];
  (h/3) * (f[a] + f[b] +
     4*Sum[f[a + i*h], {i, 1, n - 1, 2}] +
     2*Sum[f[a + i*h], {i, 2, n - 2, 2}])]
f[x_] := 1/(1 + x)
Simpson[f, 0, 1, 2]
```

Question : 2

In[497]:=

```
Simpson[f_, a_, b_, n_] := Module[{h = (b - a)/n},
  If[OddQ[n], Return["n must be even"]];
  (h/3) * (f[a] + f[b] +
     4*Sum[f[a + i*h], {i, 1, n - 1, 2}] +
     2*Sum[f[a + i*h], {i, 2, n - 2, 2}])]
f[x_] := 1/(1 + x)
Simpson[f, 0, 1, 3]
```

Out[499]=

```
n must be even
```

Question : 3

In[500]:=

```
SimpsonOneThird[f_, a_, b_, n_] :=
 Module[{h, sum1, sum2},
  If[OddQ[n], Return["n must be even"]];
  h = (b - a)/n;
  sum1 = Sum[f[a + i*h], {i, 1, n - 1, 2}];
  sum2 = Sum[f[a + i*h], {i, 2, n - 2, 2}];
  (h/3)*(f[a] + 4*sum1 + 2*sum2 + f[b])]
f[x_] := 1/(1 + x)
SimpsonOneThird[f, 0, 1, 2]
```

Out[502]=

$$\frac{25}{36}$$