



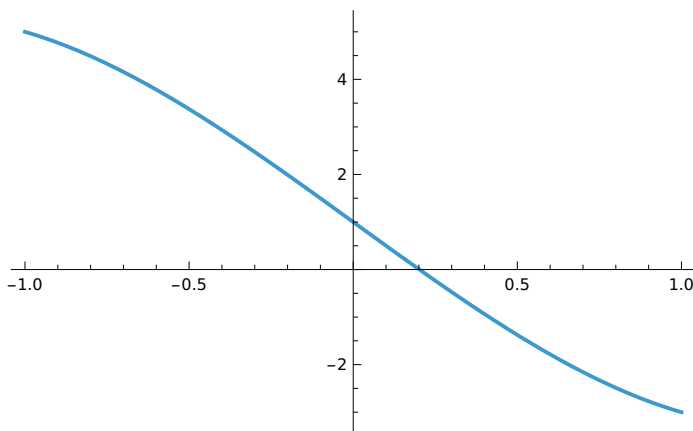
Made by *human* on Earth for Aliens

### Bisection Method

#### Code 1

```
In[1]:= f[x_] := x^3 - 5 x + 1;  
a = 0;  
b = 1;  
nmax = 15;  
Plot[f[x], {x, -1, 1}]  
For[i = 1, i ≤ nmax, i++, c = (a + b) / 2;  
Print["Root after iteration ", i, " is c = ", N[c, 6], " f(c) = ", N[f[c], 6], "\n"];  
If[f[a] * f[c] < 0, b = c, a = c];  
]
```

Out[5]=



Root after iteration 1 is  $c = 0.500000$   $f(c) = -1.37500$

Root after iteration 2 is  $c = 0.250000$   $f(c) = -0.234375$

Root after iteration 3 is  $c = 0.125000$   $f(c) = 0.376953$

Root after iteration 4 is  $c = 0.187500$   $f(c) = 0.0690918$

Root after iteration 5 is  $c = 0.218750$   $f(c) = -0.0832825$

Root after iteration 6 is  $c = 0.203125$   $f(c) = -0.00724411$

Root after iteration 7 is  $c = 0.195313$   $f(c) = 0.0308881$

Root after iteration 8 is  $c = 0.199219$   $f(c) = 0.0118129$

Root after iteration 9 is  $c = 0.201172$   $f(c) = 0.00228208$

Root after iteration 10 is  $c = 0.202148$   $f(c) = -0.00248160$

Root after iteration 11 is  $c = 0.201660$   $f(c) = -0.0000999043$

Root after iteration 12 is  $c = 0.201416$   $f(c) = 0.00109105$

Root after iteration 13 is  $c = 0.201538$   $f(c) = 0.000495564$

Root after iteration 14 is  $c = 0.201599$   $f(c) = 0.000197827$

Root after iteration 15 is  $c = 0.201630$   $f(c) = 0.0000489610$

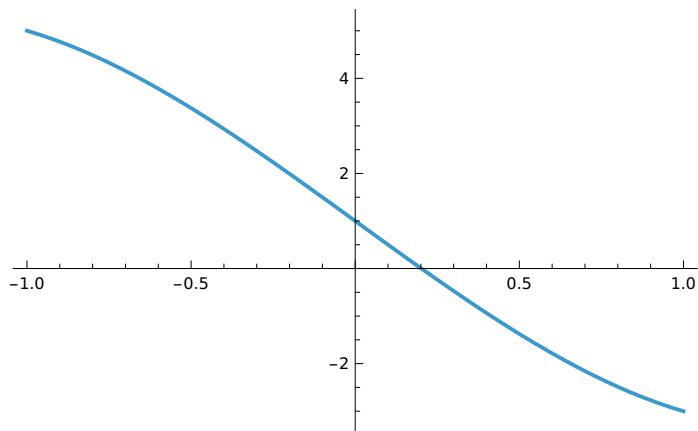
Code 2

```

In[13]:= f[x_] := x^3 - 5 x + 1;
a = 0;
b = 1;
nmax = 15;
Plot[f[x], {x, -1, 1}]
For[i = 1, i ≤ nmax, i++, c = (a + b) / 2;
Print[TableForm[{{i, N[c, 4], N[f[c], 4]}]];
If[f[a]*f[c] < 0, b = c, a = c];
]

```

Out[17]=



1	0.5000	-1.375
2	0.2500	-0.2344
3	0.1250	0.3770
4	0.1875	0.06909
5	0.2188	-0.08328
6	0.2031	-0.007244
7	0.1953	0.03089
8	0.1992	0.01181
9	0.2012	0.002282
10	0.2021	-0.002482
11	0.2017	-0.00009990
12	0.2014	0.001091
13	0.2015	0.0004956
14	0.2016	0.0001978
15	0.2016	0.00004896

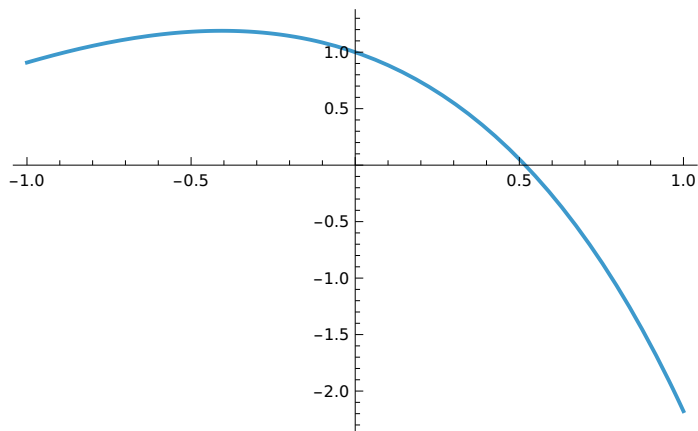
Code 3

```

In[19]:= f[x_] := Cos[x] - x * E ^ x;
a = 0;
b = 1;
nmax = 10;
Plot[f[x], {x, -1, 1}]
For[i = 1, i ≤ nmax, i++, c = (a + b) / 2;
Print[TableForm[{{i, N[c, 4], N[f[c], 4]}]];
If[f[a] * f[c] < 0, b = c, a = c];
]

```

Out[23]=



1	0.5000	0.05322
2	0.7500	-0.8561
3	0.6250	-0.3567
4	0.5625	-0.1413
5	0.5313	-0.04151
6	0.5156	0.006475
7	0.5234	-0.01736
8	0.5195	-0.005404
9	0.5176	0.0005452
10	0.5186	-0.002427

Secant Method

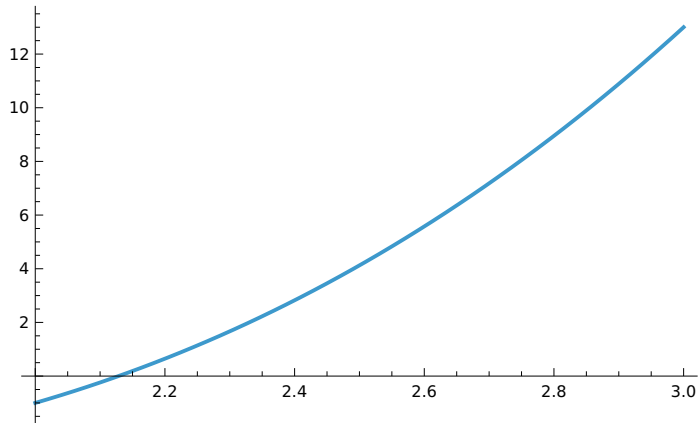
Code 1

```

In[7]:= f[x_] := x^3 - 5 x + 1;
x0 = 2;
x1 = 3;
nmax = 5;
Plot[f[x], {x, x0, x1}]
For[i = 1, i ≤ nmax, i++, x2 = N[x1 - ((x1 - x0) / (f[x1] - f[x0])) f[x1]];
Print["Root after iteration ", i, " is ", x2]; x0 = x1; x1 = x2;
]

```

Out[11]=



Root after iteration 1 is 2.07143

Root after iteration 2 is 2.10376

Root after iteration 3 is 2.12952

Root after iteration 4 is 2.1284

Root after iteration 5 is 2.12842

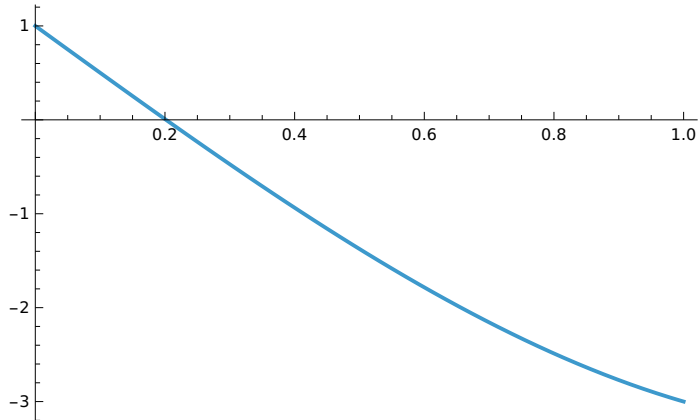
Code 2

```

In[31]:= f[x_] := x^3 - 5 x + 1;
x0 = 0;
x1 = 1;
nmax = 8;
Plot[f[x], {x, x0, x1}]
For[i = 1, i ≤ nmax, i++, x2 = N[x1 - ((x1 - x0) / (f[x1] - f[x0])) f[x1]];
Print[TableForm[{{i, N[x2], N[f[x2]]}}]]; x0 = x1; x1 = x2;
]

```

Out[35]=



1	0.25	-0.234375
2	0.186441	0.0742773
3	0.201736	-0.000471116
4	0.20164	$-8.64229 \times 10^{-7}$
5	0.20164	$1.03527 \times 10^{-11}$
6	0.20164	$-2.22045 \times 10^{-16}$
7	0.20164	$1.11022 \times 10^{-16}$
8	0.20164	$1.11022 \times 10^{-16}$

Regula Falsi Method

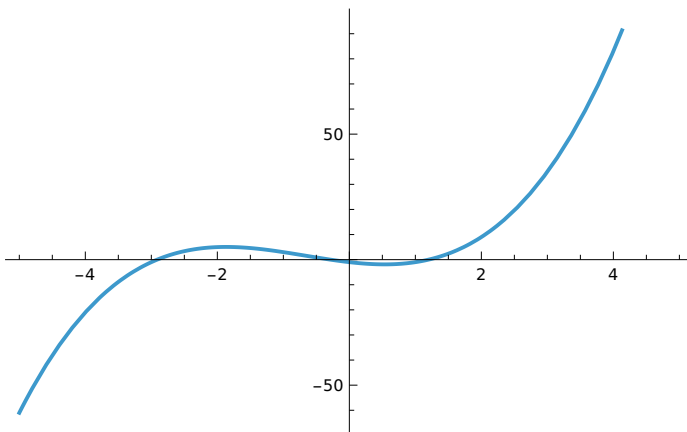
Code 1

```

In[67]:= f[x_] := x^3 + 2 x^2 - 3 x - 1;
x0 = 1;
x1 = 2;
nmax = 10;
Plot[f[x], {x, -5, 5}]
For[i = 1, i ≤ nmax, i++, x2 = N[x1 - ((x1 - x0) / (f[x1] - f[x0])) f[x1]];
Print["Root after iteration ", i, " is ", x2];
If[f[x0] * f[x2] < 0, x1 = x2, x0 = x2];
]

```

Out[71]=



Root after iteration 1 is 1.1

Root after iteration 2 is 1.15174

Root after iteration 3 is 1.17684

Root after iteration 4 is 1.18863

Root after iteration 5 is 1.19408

Root after iteration 6 is 1.19658

Root after iteration 7 is 1.19773

Root after iteration 8 is 1.19825

Root after iteration 9 is 1.19849

Root after iteration 10 is 1.1986

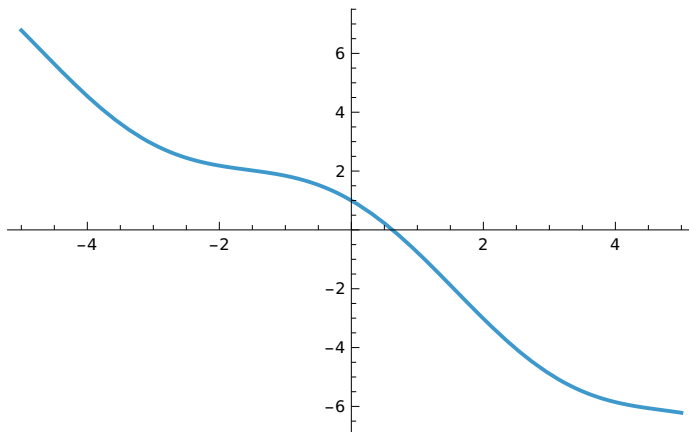
Code 2

```

In[73]:= f[x_] := Cos[x] - 1.3 x;
x0 = 1;
x1 = 2;
nmax = 10;
Plot[f[x], {x, -5, 5}]
For[i = 1, i ≤ nmax, i++, x2 = N[x1 - ((x1 - x0) / (f[x1] - f[x0])) f[x1]];
Print["Root after iteration ", i, " is ", x2];
If[f[x0]*f[x2] < 0, x1 = x2, x0 = x2];
]

```

Out[77]=



```

Root after iteration 1 is 0.663322
Root after iteration 2 is 0.629531
Root after iteration 3 is 0.624933
Root after iteration 4 is 0.62429
Root after iteration 5 is 0.624199
Root after iteration 6 is 0.624187
Root after iteration 7 is 0.624185
Root after iteration 8 is 0.624185
Root after iteration 9 is 0.624185
Root after iteration 10 is 0.624185

```

Newton Raphson Method

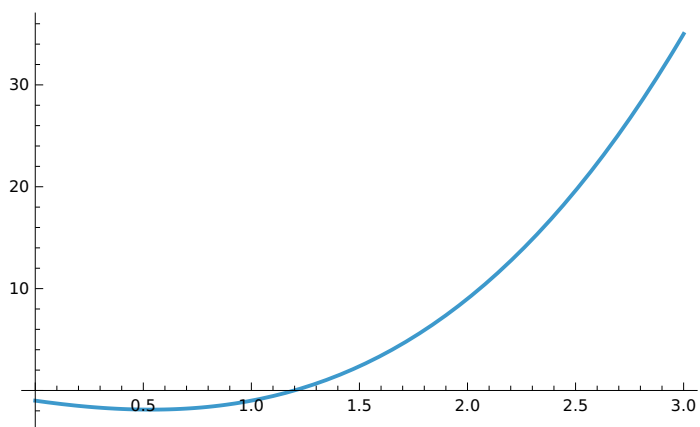
Code 1

```

In[79]:= f[x_] := x^3 + 2 x^2 - 3 x - 1;
a = 1;
b = 2;
nmax = 6;
Plot[f[x], {x, a - 1, b + 1}]
For[i = 1, i ≤ nmax, i++, c = N[b - (f[b] / f'[b])];
Print["The ", i, " th iteration is: ", c];
a = b; b = c;
]

```

Out[83]=



The 1 th iteration is: 1.47059

The 2 th iteration is: 1.24713

The 3 th iteration is: 1.2007

The 4 th iteration is: 1.19869

The 5 th iteration is: 1.19869

The 6 th iteration is: 1.19869

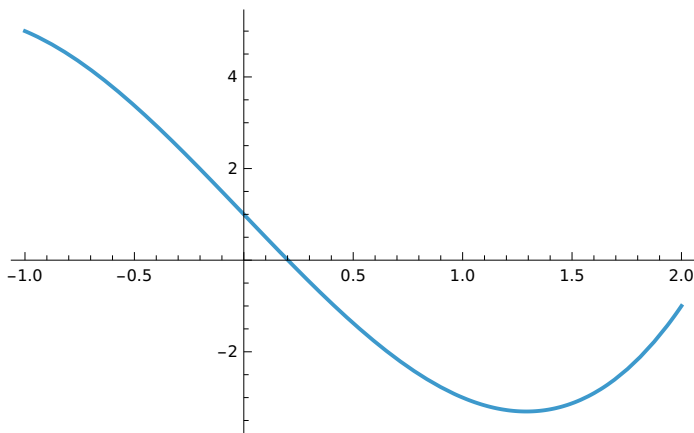
Code 2

```

In[85]:= f[x_] := x^3 - 5 x + 1;
a = 0;
b = 1;
nmax = 6;
l = List[];
Plot[f[x], {x, a - 1, b + 1}]
For[i = 1, i ≤ nmax, i++, c = N[b - (f[b] / f'[b])];
a = b; b = c;
seq = AppendTo[l, {N[c], N[f[c]]}]
TableForm[seq, TableHeadings → {"a", "b", "c", "d", "e", "f"}, {"i", "c", "f(c)"},
TableAlignments → Center]

```

Out[90]=



Out[92]//TableForm=

	i	c
a	-0.5	3.375
b	0.294118	-0.445146
c	0.200215	0.00695237
d	0.201639	$1.22213 \times 10^{-6}$
e	0.20164	$3.79696 \times 10^{-14}$
f	0.20164	$1.11022 \times 10^{-16}$

## Gauss Jacobi

Q. Solve the system of equations

$$8x + 3y + 4z = 15;$$

$$-2x + 5y - 2z = 1;$$

$$x + y + 3z = 5;$$

using gauss jacobi iteration method. Use initial approximation as  $[x, y, z] = [0, 0, 0]$ ;

Code and Output

```

In[1]:= x = 0;
        y = 0;
        z = 0;
        nmax = 10;
        For[i = 1, i ≤ nmax, i++, xnew = N[1/8 (15 - 3 y - 4 z), 5];
          ynew = N[1/5 (1 + 2 x + 2 z), 5]; znew = N[1/3 (5 - x - y), 5];
        Print["The ", i, " th iteration is xnew = ", xnew, " ynew = ", ynew, " znew = ", znew];
        x = xnew;
        y = ynew;
        z = znew
    ]

```

The 1 th iteration is xnew = 1.8750 ynew = 0.20000 znew = 1.6667  
 The 2 th iteration is xnew = 0.96667 ynew = 1.6167 znew = 0.97500  
 The 3 th iteration is xnew = 0.7813 ynew = 0.97667 znew = 0.80556  
 The 4 th iteration is xnew = 1.1060 ynew = 0.83472 znew = 1.0807  
 The 5 th iteration is xnew = 1.0216 ynew = 1.0747 znew = 1.0198  
 The 6 th iteration is xnew = 0.96212 ynew = 1.0166 znew = 0.96790  
 The 7 th iteration is xnew = 1.0098 ynew = 0.97201 znew = 1.0071  
 The 8 th iteration is xnew = 1.0069 ynew = 1.0068 znew = 1.0061  
 The 9 th iteration is xnew = 0.99443 ynew = 1.0052 znew = 0.99543  
 The 10 th iteration is xnew = 1.0003 ynew = 0.99594 znew = 1.0001

### Gauss Seidel

*Q. Solve the system of equations*

$$8x + 3y + 4z = 15;$$

$$-2x + 5y - 2z = 1;$$

$$x + y + 3z = 5;$$

*using gauss seidel iteration method. Use initial approximation as  $[x, y, z] = [0, 0, 0]$ ;*

Code and Output

```

In[11]:= x = 0;
y = 0;
z = 0;
nmax = 10;
For[i = 1, i ≤ nmax, i++, xnew = N[1/8 (15 - 3 y - 4 z), 6];
  ynew = N[1/5 (1 + 2 xnew + 2 z), 5]; znew = N[1/3 (5 - xnew - ynew), 5];
  Print["The ", i, " th iteration is xnew = ", xnew, " ynew = ", ynew, " znew = ", znew];
  x = xnew;
  y = ynew;
  z = znew;
]

```

The 1 th iteration is xnew = 1.87500 ynew = 0.95000 znew = 0.72500

The 2 th iteration is xnew = 1.15625 ynew = 0.95250 znew = 0.96375

The 3 th iteration is xnew = 1.03594 ynew = 0.99988 znew = 0.98806

The 4 th iteration is xnew = 1.00602 ynew = 0.99763 znew = 0.99878

The 5 th iteration is xnew = 1.00150 ynew = 1.0001 znew = 0.99946

The 6 th iteration is xnew = 1.00023 ynew = 0.99988 znew = 0.99997

The 7 th iteration is xnew = 1.00006 ynew = 1.0000 znew = 0.99997

The 8 th iteration is xnew = 1.00001 ynew = 0.99999 znew = 1.0000

The 9 th iteration is xnew = 1.00000 ynew = 1.0000 znew = 1.0000

The 10 th iteration is xnew = 1.00000 ynew = 1.0000 znew = 1.0000

### Module Command

```

In[16]:= abs[x0_] := Module[{x = x0}, If[x > 0, x = x0, x = -x0]]
abs[-16]

```

Out[17]=  
16

```

In[18]:= arearect[x0_, y0_] := Module[{x = x0, y = y0, Area, Per}, Area = x*y; Per = 2*(x + y);
  Print["Area = ", N[Area]];
  Print["Per = ", N[Per]]
]
arearect[2.4, 4]
Area = 9.6
Per = 12.8

```

```

In[24]:= fact[x0_] := Module[{n = x0}, factn = 1;
  For[i = 1, i ≤ n, i++, factn = factn*i];
  Print["Factorial of ", n, " is = ", factn];
]
fact[5]
Factorial of 5 is = 120

```

### Gauss Jacobi Method using Matrix Method

```

In[30]:= {{8, 3, 4}, {-2, 5, -2}, {1, 1, 3}} // MatrixForm

```

Out[30]//MatrixForm=

$$\begin{pmatrix} 8 & 3 & 4 \\ -2 & 5 & -2 \\ 1 & 1 & 3 \end{pmatrix}$$

```

In[29]:= B = {{15}, {1}, {5}} // MatrixForm

```

Out[29]//MatrixForm=

$$\begin{pmatrix} 15 \\ 1 \\ 5 \end{pmatrix}$$

Code and Output

```

In[61]:= A = {{8, 3, 4}, {-2, 5, -2}, {1, 1, 3}};
B = {{15}, {1}, {5}};
nmax = 10;

jacobi[x0_, y0_, z0_, iterations_] :=
Module[{x, y, z, list, i},
  x[0] = x0; y[0] = y0; z[0] = z0;
  list = {"Iteration", "x", "y", "z"}, {0, x[0], y[0], z[0]};
  For[i = 0, i < iterations, i++,
    x[i + 1] = N[(B[[1, 1]] - A[[1, 2]]*y[i] - A[[1, 3]]*z[i])/A[[1, 1]]];
    y[i + 1] = N[(B[[2, 1]] - A[[2, 1]]*x[i] - A[[2, 3]]*z[i])/A[[2, 2]]];
    z[i + 1] = N[(B[[3, 1]] - A[[3, 2]]*y[i] - A[[3, 1]]*x[i])/A[[3, 3]]];
    AppendTo[list, {i + 1, x[i + 1], y[i + 1], z[i + 1]}];
  ];
  TableForm[list]
]
jacobi[0, 0, 0, nmax]

```

Out[65]//TableForm=

Iteration	x	y	z
0	0	0	0
1	1.875	0.2	1.66667
2	0.966667	1.61667	0.975
3	0.78125	0.976667	0.805556
4	1.10597	0.834722	1.08069
5	1.02163	1.07467	1.01977
6	0.962116	1.01656	0.9679
7	1.00984	0.972006	1.00711
8	1.00694	1.00678	1.00605
9	0.994432	1.0052	0.995426
10	1.00034	0.995943	1.00012

## Indexing of a Number

In[93]:= X = {1, 5, 7}

X[[2]]

Out[93]=

{1, 5, 7}

Out[94]=

5

In[95]:= Sum[k, {k, 1, n}]

Out[95]=

$$\frac{1}{2} n (1 + n)$$

```
In[96]:= X = {1, 5, 7}
Product[x - X[[j]], {j, 1, 3}]
```

```
Out[96]:=
{1, 5, 7}
```

```
Out[97]:=
(-7 + x) (-5 + x) (-1 + x)
```

## Lagrange Interpolation

Q. Find the Lagrange interpolating polynomial for the following data  $(0,1) = (x_0, f(x_0))$ ,  $(1,3) = (x_1, f(x_1))$ ,  $(3,55) = (x_2, f(x_2))$

### Code 1

```
In[98]:= X = {0, 1, 3};
Y = {1, 3, 55};
n = 3;
For[k = 1, k ≤ n, k++,
L[n, k, x_] := Product[(x - X[[j]]) / (X[[k]] - X[[j]]), {j, 1, k - 1}] *
Product[(x - X[[j]]) / (X[[k]] - X[[j]]), {j, k + 1, n}]];
a = Sum[Y[[k]] * L[n, k, x], {k, 1, n}];
Print[a]
Simplify[a]

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

Out[104]:=
 $1 - 6x + 8x^2$ 
```

### Code 2

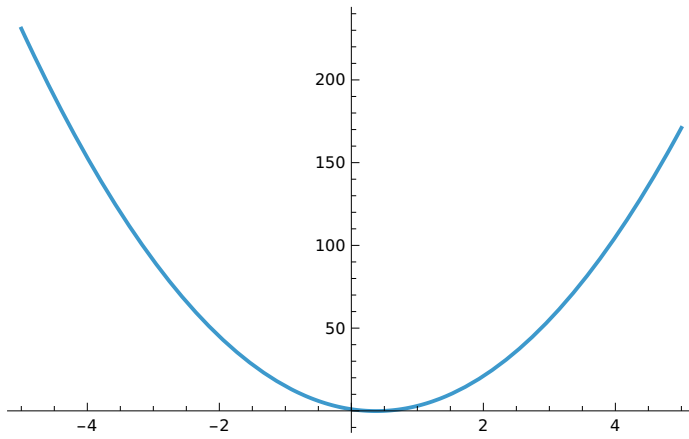
```
In[105]:= X = {0, 1, 3};
Y = {1, 3, 55};
n = 3;
For[k = 1, k ≤ n, k++,
L[n, k, x_] := Product[(x - X[[j]]) / (X[[k]] - X[[j]]), {j, 1, k - 1}] *
Product[(x - X[[j]]) / (X[[k]] - X[[j]]), {j, k + 1, n}]];
Simplify[Sum[Y[[k]] * L[n, k, x], {k, 1, n}]]

Out[109]:=
 $1 - 6x + 8x^2$ 
```

```
In[110]:=
```

```
Plot[%, {x, -5, 5}]
```

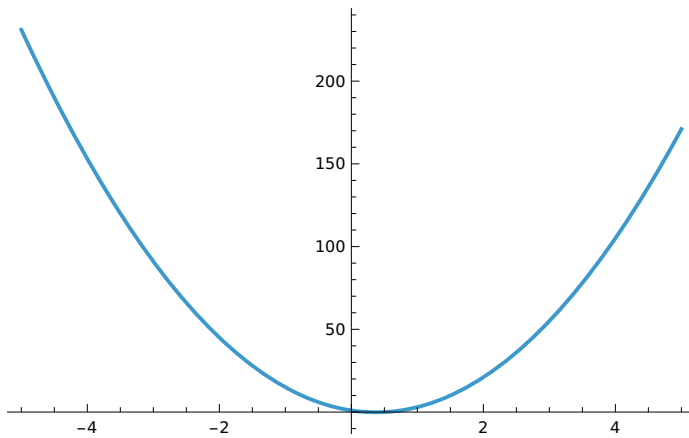
```
Out[110]=
```



```
In[111]:=
```

```
Plot[a, {x, -5, 5}]
```

```
Out[111]=
```



Code 3

In[143]:=

```

X = {-2, -1, 0, 1, 3, 4};
Y = {9, 16, 17, 18, 44, 81};
n = 6;
For[k = 1, k ≤ n, k++,
L[n, k, x_] := Product[(x - X[[j]]) / (X[[k]] - X[[j]]), {j, 1, k - 1}] *
Product[(x - X[[j]]) / (X[[k]] - X[[j]]), {j, k + 1, n}];
a = Sum[Y[[k]] * L[n, k, x], {k, 1, n}];
Print[a]
Simplify[a]

```

$$-\frac{1}{20}(-1-x)(1-x)(3-x)(4-x)x - \frac{2}{5}(1-x)(3-x)(4-x)x(2+x) + \frac{17}{24}(1-x)(3-x)(4-x)(1+x)(2+x) + \frac{1}{2}(3-x)(4-x)x(1+x)(2+x) + \frac{11}{30}(4-x)(-1+x)x(1+x)(2+x) + \frac{9}{40}(-3+x)(-1+x)x(1+x)(2+x)$$

Out[149]=

17 + x<sup>3</sup>

### Newton's Interpolation Method

#### Code and Output

In[119]:=

```

X = {0, 1, 3};
Y = {1, 3, 55};
n = 2;
d = Table["", {n + 1}, {n + 1}];
d[[All, 1]] = Y[[All]];
For[j = 1, j ≤ n, j++,
For[k = j, k ≤ n, k++,
d[[k + 1, j + 1]] = (d[[k + 1, j]] - d[[k, j]]) / (X[[k + 1]] - X[[k + 1 - j]]);
];
];
For[k = 0, k ≤ n, k++,
p[k + 1, x_] := Product[(x - X[[j]]), {j, 1, k}];
];
a = Sum[d[[k + 1, k + 1]] * p[k + 1, x], {k, 0, n}];
Print[a]
Simplify[a]

```

1 + 2 x + 8 (-1 + x) x

Out[128]=

1 - 6 x + 8 x<sup>2</sup>

### Trapezoidal Rule

Q. Find the approximate value of  $\int (x+1) dx$  from 0 to 1.

Code and Output

```

In[1]:= f[x_] := x + 1;
lowerlimit = a = 0;
upperlimit = b = 1;
fa = f[a];
fb = f[b];
h = b - a;
c = (h (fa + fb)) / 2;
Print["Integral = ", c]

Integral =  $\frac{3}{2}$ 

```

*Q. Find the approximate value of the integral  $\int (x^2 - x - 1) dx$  from 3 to 5 and compare it with exact value of the integral also.*

Code and Output

```

In[19]:= f[x_] := x^2 - x - 1;
lowerlimit = a = 3;
upperlimit = b = 5;
fa = f[a];
fb = f[b];
h = b - a;
c = (h (fa + fb)) / 2;
Print["The area approximate by the trapezoidal rule is ", c]
int = Integrate[f[x], {x, a, b}];
Print["Original value of the integral is ", N[int], " and the error is ", N[c] - N[int]]

The area approximate by the trapezoidal rule is 24
Original value of the integral is 22.6667 and the error is 1.33333

```

*Q. Find the approximate value of the integral  $\int (x^3 + 4x - 13) dx$  from 3 to 4 and compare it with exact value of the integral also.*

Code and Output

```

In[29]:= f[x_] := x^3 + 4 x - 13;
lowerlimit = a = 3;
upperlimit = b = 4;
fa = f[a];
fb = f[b];
h = b - a;
c = (h (fa + fb)) / 2;
Print["The area approximate by the trapezoidal rule is ", c]
int = Integrate[f[x], {x, a, b}];
Print["Original value of the integral is ", N[int], " and the error is ", N[c] - N[int]]

The area approximate by the trapezoidal rule is  $\frac{93}{2}$ 
Original value of the integral is 44.75 and the error is 1.75

```

### Composite Trapezoidal Rule

Q. Find the approximate value of  $\int (x+1) dx$  from 0 to 1 by using composite trapezoidal rule for  $n=10$ .  
 (/. x -> means replace x by a+i h)

#### Code and Output

```

In[66]:= f[x_] := x + 1;
lowerlimit = a = 0;
upperlimit = b = 1;
n = 10;
fa = f[a];
fb = f[b];
h = (b - a) / n;
sum = 0;
For[i = 1, i < n, i++, sum = sum + N[f[x] /. x -> (a + i h)]]
sum = (h / 2) (fa + 2 sum + fb);
Print["Integral = ", sum]

Integral = 1.5

```

### Simpson's Rule

Q. Find the approximate value of the integral  $\int (x+1) dx$  from 0 to 1 by using Simpson's rule.

#### Code and Output

```

In[85]:= f[x_] := x + 1;
lowerlimit = a = 0;
upperlimit = b = 1;
fa = f[a];
fb = f[b];
h = (b - a) / 2;
c = (h (fa + 4 f[a + h] + fb)) / 3;
Print["Integral = ", N[c]]

Integral = 1.5

```

Q. Find the approximate value of the integral  $\int (x^2 + 2x + 7) dx$  from 1 to 2 by using Simpson's rule and compare it with exact value of the integral also.

### Code and Output

```

In[104]:= f[x_] := x^2 + 2 x + 7;
lowerlimit = a = 1;
upperlimit = b = 2;
fa = f[a];
fb = f[b];
h = (b - a) / 2;
c = (h (fa + 4 f[a + h] + fb)) / 3;
Print["The area approximate by the simpson's rule is ", N[c]]
int = Integrate[f[x], {x, a, b}];
Print["Original value of the integral is ", N[int], " and the error is ", N[c] - N[int]]
byTrap = N[(b - a) / 2 * (fa + fb)];
Print["The area approximate by the trapezoidal rule is ", N[byTrap]]

The area approximate by the simpson's rule is 12.3333
Original value of the integral is 12.3333 and the error is 0.
The area approximate by the trapezoidal rule is 12.5

```

## Simpson's 3/8 Rule

Q. Approximate the value of the integral  $\int \sin(x)/(x^2 + 1) dx$  from 1 to 3.

Formulae =  $\int f(x) dx = (3h/8) * [f(a) + 3f(a+h) + 3f(a+2h) + f(b)]$  → Limits of integral is from a to b.

### Code and Output

```

In[1]:= f[x_] := Sin[x]/(x^2+1);
lowlimit = a = 1;
upplimit = b = 3;
fa = f[a];
fb = f[b];
h = (b - a)/3;
c = (3 h*(fa+3 f[a+h]+3 f[a+2 h]+fb))/8;
Print["The area approximated by the simpson's rule is ", N[c]]

The area approximated by the simpson's rule is 0.390482

```

### Composite Simpson's 1/3 rd Rule

Q. Find the approximate value of the integral  $\int e^x dx$  from 0 to 4 by using composite simpson's rule for  $n=20$ .

Formuale =  $\int f(x) dx =$   
 $(h/3)*[f(a)+4(f(x_1)+f(x_3)+...+f(x_{2n-1}))+2(f(x_2)+f(x_4)+...+f(x_{2n-2}))+f(b)]$  →  
 Limits of integral is from a to b.

### Code and Output

```

In[96]:= f[x_] := E^x;
lowlimit = a = 0;
upplimit = b = 4;
n = 20;
fa = f[a];
fb = f[b];
h = (b - a)/n;
sumodd = 0;
sumeven = 0;
For[i = 1, i < n, i += 2, sumodd = sumodd + N[f[x] /. x -> (a + i h)]]
For[i = 2, i < n, i += 2, sumeven = sumeven + N[f[x] /. x -> (a + i h)]]
sum = (h/3)(fa + 4 sumodd + 2 sumeven + fb);
Print["Integral = ", sum]
exact = N[Integrate[f[x], {x, a, b}]];
Print["Exact value of integral is ", exact]
error = N[sum] - N[exact];
Print["The error between exact and numerical value is: ", error]

Integral = 53.5986
Exact value of integral is 53.5982
The error between exact and numerical value is: 0.000474169

```

### Euler's Method

Q.  $dy/dx = 1+(y/x)$ ,  $y(1)=1$ ,  $1 \leq x \leq 6$ ,  $n=10$ .

### Code and Output

In[129]:=

```
f[x_, y_] := 1 + (y/x);
a = 1;
b = 6;
n = 10;
h = (b - a)/n;
y[1] = 1;
temp = y[1];
For[i = 0, i ≤ n - 1, i++, x[i] = a + i h;
y[i] = temp;
y[i + 1] = y[i] + h f[x[i], y[i]];
Print["The ", i + 1, " approximation is ", N[y[i + 1]]];
temp = y[i + 1]
];
```

The 1 approximation is 2.

The 2 approximation is 3.16667

The 3 approximation is 4.45833

The 4 approximation is 5.85

The 5 approximation is 7.325

The 6 approximation is 8.87143

The 7 approximation is 10.4804

The 8 approximation is 12.1448

The 9 approximation is 13.8593

The 10 approximation is 15.6193

Made by *Human* — tested by confusion, approved by deadlines.