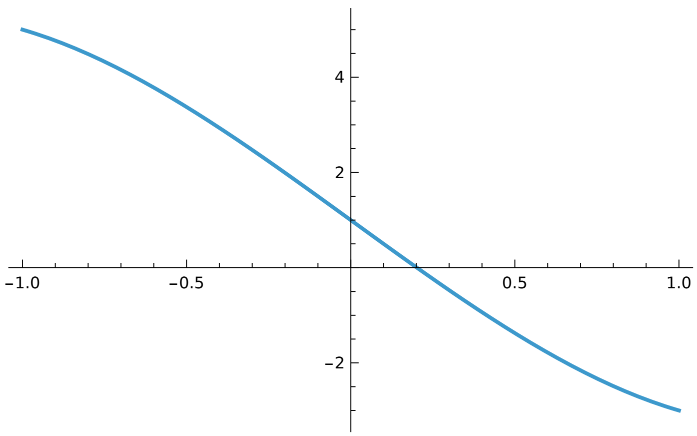
Bisection Method

In[32]:=
```
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[36]=



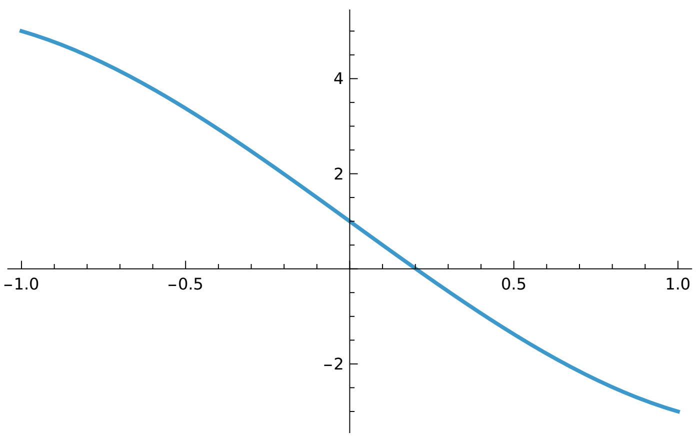
```
root after iteration 1 is c =  0.500000f[c] =  -1.37500

root after iteration 2 is c =  0.250000f[c] =  -0.234375

root after iteration 3 is c =  0.125000f[c] =  0.376953

root after iteration 4 is c =  0.187500f[c] =  0.0690918

root after iteration 5 is c =  0.218750f[c] =  -0.0832825

root after iteration 6 is c =  0.203125f[c] =  -0.00724411

root after iteration 7 is c =  0.195313f[c] =  0.0308881

root after iteration 8 is c =  0.199219f[c] =  0.0118129

root after iteration 9 is c =  0.201172f[c] =  0.00228208

root after iteration 10 is c =  0.202148f[c] =  -0.00248160

root after iteration 11 is c =  0.201660f[c] =  -0.0000999043

root after iteration 12 is c =  0.201416f[c] =  0.00109105

root after iteration 13 is c =  0.201538f[c] =  0.000495564

root after iteration 14 is c =  0.201599f[c] =  0.000197827

root after iteration 15 is c =  0.201630f[c] =  0.0000489610
```

Tabular form

In[44]:=

```
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, 6], N[f[c], 6]}}]];
If[f[a] * f[c] < 0, b = c, a = c];
]
```

Out[48]=

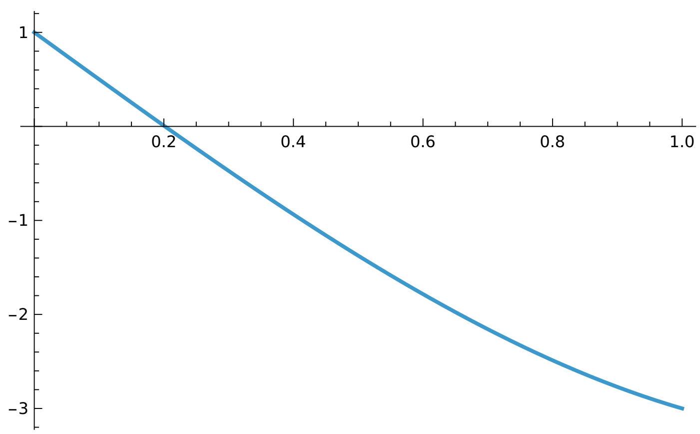


1	0.500000	-1.37500
2	0.250000	-0.234375
3	0.125000	0.376953
4	0.187500	0.0690918
5	0.218750	-0.0832825
6	0.203125	-0.00724411
7	0.195313	0.0308881
8	0.199219	0.0118129
9	0.201172	0.00228208
10	0.202148	-0.00248160
11	0.201660	-0.0000999043
12	0.201416	0.00109105
13	0.201538	0.000495564
14	0.201599	0.000197827
15	0.201630	0.0000489610

Secant Method

In[13]:=
```
f[x_] := x^3 - 5 x + 1;
x0 = 0;
x1 = 1;
nmax = 5;
Plot[f[x], {x, x0, x1}]
For[i = 1, i ≤ nmax, i++, x2 = N[x1 - ((x1 - x0) f[x1] / (f[x1] - f[x0]))];
Print[TableForm[{{i, N[x2, 6], N[f[x2], 6]}}]]; x0 = x1; x1 = x2;
]
```

Out[17]=

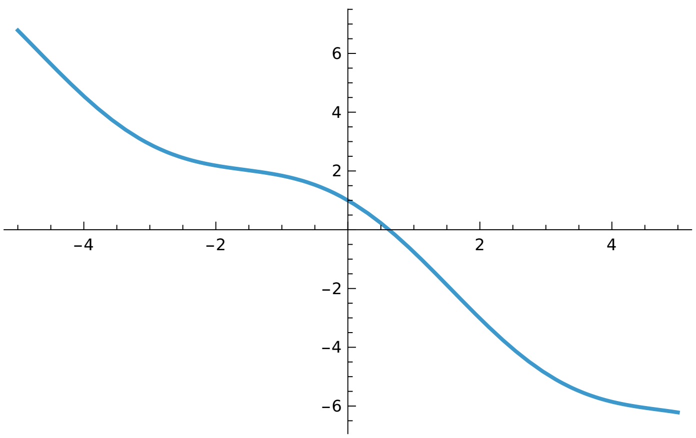


1	0.25	-0.234375
2	0.186441	0.0742773
3	0.201736	-0.000471116
4	0.20164	-8.64229×10^{-7}
5	0.20164	1.03527×10^{-11}

Regula Falsi

In[61]:=
```
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[x1] - f[x0]))];
Print[TableForm[{{i, N[x2, 5], N[f[x2], 5]}}]];
If[f[x0] * f[x2] < 0, x1 = x2, x0 = x2];
]
```

Out[65]=

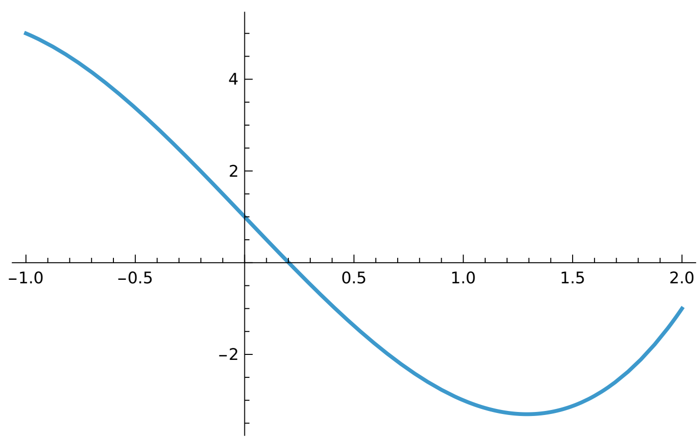


1	0.663322	-0.0743667
2	0.629531	-0.0100866
3	0.624933	-0.00140967
4	0.62429	-0.000197808
5	0.624199	-0.0000277726
6	0.624187	-3.89963×10^{-6}
7	0.624185	-5.47563×10^{-7}
8	0.624185	-7.68859×10^{-8}
9	0.624185	-1.07959×10^{-8}
10	0.624185	-1.5159×10^{-9}

Newton Raphson

In[19]:=
```
f[x_] := x^3 - 5 x + 1;
a = 0;
b = 1;
nmax = 5;
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, {i, N[c, 5], N[f[c], 5]}]];
TableForm[seq, TableHeadings → {{"1", "2", "3", "4", "5"}, {"i", "c", "f[c]"}},
TableAlignments → Center]
```

Out[24]=



Out[26]//TableForm=

	i	c	f[c]
1	1	-0.5	3.375
2	2	0.294118	-0.445146
3	3	0.200215	0.00695237
4	4	0.201639	1.22213×10^{-6}
5	5	0.20164	3.79696×10^{-14}

Gauss Seidel

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]];

In[27]:=
```
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 xnew + 2 z), 5];
 znew = N[(1/3) (5 - xnew - ynew), 5];

 Print["The ", i, " th iteration is:  x = ", xnew, "   y = ", ynew, "   z = ", znew];
 x = xnew;
 y = ynew;
 z = znew;
]
```

```
The 1 th iteration is:  x = 1.8750   y = 0.95000   z = 0.72500
The 2 th iteration is:  x = 1.1563   y = 0.95250   z = 0.96375
The 3 th iteration is:  x = 1.0359   y = 0.99988   z = 0.98806
The 4 th iteration is:  x = 1.0060   y = 0.99763   z = 0.99878
The 5 th iteration is:  x = 1.0015   y = 1.0001   z = 0.99946
The 6 th iteration is:  x = 1.0002   y = 0.99988   z = 0.99997
The 7 th iteration is:  x = 1.0001   y = 1.0000   z = 0.99997
The 8 th iteration is:  x = 1.0000   y = 0.99999   z = 1.0000
The 9 th iteration is:  x = 1.0000   y = 1.0000   z = 1.0000
The 10 th iteration is:  x = 1.0000   y = 1.0000   z = 1.0000
```

Gauss jacobi method

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]];

In[32]:=
```
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:  x = ", xnew, "   y = ", ynew, "   z = ", znew];
 x = xnew;
 y = ynew;
 z = znew;
]
```

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

Module Command

In[37]:=
```
abs[x0_] := Module[{x = x0}, If[x > 0, x = x0, x = -x0]]
abs[-16]
```

Out[38]=
```
16
```

In[39]:=
```
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[41]:=
```
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 using Matrix Method

In[48]:=
```
{{8, 3, 4}, {-2, 5, -2}, {1, 1, 3}} // MatrixForm
```

Out[48]//MatrixForm=

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

In[49]:=
```
{{8, 3, 4}, {-2, 5, -2}, {1, 1, 3}}^-1
```

Out[49]=

$$\left\{\left\{\frac{17}{120}, -\frac{1}{24}, -\frac{13}{60}\right\}, \left\{\frac{1}{30}, \frac{1}{6}, \frac{1}{15}\right\}, \left\{-\frac{7}{120}, -\frac{1}{24}, \frac{23}{60}\right\}\right\}$$

In[50]:=
```
B = {{15}, {1}, {5}} // MatrixForm
```

Out[50]//MatrixForm=

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

```
In[43]:= 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},

  (* initial values *)
  x[0] = x0;
  y[0] = y0;
  z[0] = z0;

  (* table list *)
  list = {{"Iteration", "x", "y", "z"}, {0, x[0], y[0], z[0]}};

  (* Jacobi loop *)
  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[47]//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

In[51]:=
```
X = {1, 5, 7}
X[[2]]
```

Out[51]=
```
{1, 5, 7}
```

Out[52]=
```
5
```

In[56]:=
```
Sum[k, {k, 1, n}]
```

Out[56]=

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

In[61]:=
```
X = {1, 5, 7}; Product[x - X[[j]], {j, 1, 3}]
```

Out[61]=
```
0.008
```

Lagrange Interpolation

In[125]:=
```
Quit[]
```

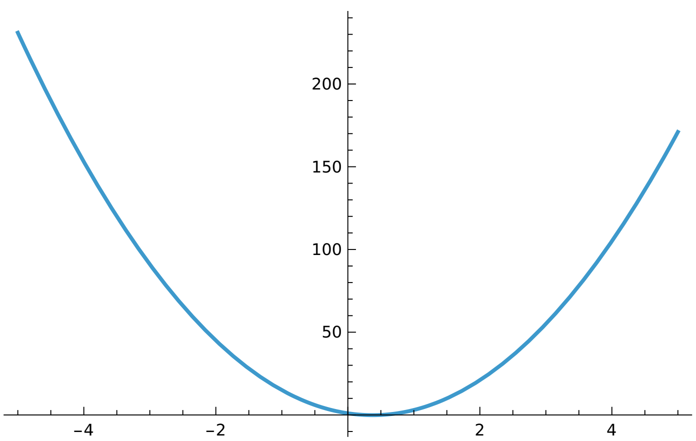
In[1]:=
```
X = {0, 1, 3};
Y = {1, 3, 55};
n = 3;

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[7]= $1 - 6x + 8x^2$

In[8]:=
```
Plot[a, {x, -5, 5}]
```

Out[8]=



In[15]:=
```
Quit[]
```

In[6]:=
```
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[12]=

$17 + x^3$

Newton's Interpolation method

In[13]:=
```
X = {0, 1, 3};
Y = {1, 3, 55};
n = 2;
d = Table["", {n + 1}, {n + 1}];
d[[All, 1]] = Y[[All]];
(* Construct divided difference table *)
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]])]];
(* p[k+1, x] polynomial term *)
For[k = 0, k ≤ n, k++,
 p[k + 1, x_] := Product[(x - X[[j]]), {j, 1, k}];
];
(* Final polynomial *)
a = Sum[d[[k + 1, k + 1]] * p[k + 1, x], {k, 0, n}];
Print[a]
Simplify[a]
```

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

Out[22]=

$1 - 6x + 8x^2$

Trapezoidal rule

In[26]:=
```
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 = $\frac{3}{2}$

In[42]:=
```
f[x_] := x ^ 2 - x - 1;
lowerlimit = a = 3;
upperlimit = b = 5;
fa = f[a];
fb = f[b];
h = b - a;
(* Trapezoidal rule *)
c = h * (fa + fb) / 2;
Print["The area approximate by the trapezoidal rule is ", c];
(* Exact integral *)
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
```

Composite Trapezoidal

In[52]:=
```
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[a + i * h]]];
sum = (h / 2) * (fa + 2 * sum + fb);
Print["Integral = ", sum];
```

```
Integral = 1.5
```

Simpsons 1/3 rule

In[63]:=
```
f[x_] := x + 1;
lowerlimit = a = 0;
upperlimit = b = 1;
fa = f[a];
fb = f[b];
h = (b - a) / 2;
c = (h / 3) * (fa + 4 * f[a + h] + fb);
Print["Integral = ", N[c]];
```

```
Integral = 1.5
```

In[71]:=
```
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 / 3) * (fa + 4 * f[a + h] + fb);
Print["The area approximate by the Simpson's rule is ", N[c]];
(* Exact Integral *)
int = Integrate[f[x], {x, a, b}];
Print["Original value of the integral is ",
  N[int], " and the error is ", N[c] - N[int]];
(* Trapezoidal Rule *)
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
```

Composite simpson's 1/3

In[83]:=
```
f[x_] := Exp[x];
lowlimit = a = 0;
upplimit = b = 4;
n = 20;
fa = f[a];
fb = f[b];
h = (b - a) / n;
sumodd = 0;
sumeven = 0;
(* odd terms *)
For[i = 1, i < n, i += 2, sumodd = sumodd + N[f[a + i * h]]];
(* even terms *)
For[i = 2, i < n, i += 2, sumeven = sumeven + N[f[a + i * h]]];
(* Simpson 1/3 composite formula *)
sum = (h / 3) * (fa + 4 * sumodd + 2 * sumeven + fb);
Print["Integral = ", sum];
(* exact integral *)
exact = N[Integrate[f[x], {x, a, b}]];
Print["Exact value of integral is ", exact];
(* error *)
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
```

Simpson's 3/8 rule

In[100]:=

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

The area approximated by Simpson's rule is 0.390482
```

Euler Method

In[108]:=

```
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
```