Numerical Methods

Lab 6 - Lagrange Interpolation, RK-2 and RK-4

FEB 2022

Submitted by

Aabhusan Aryal

076BCT001

Submitted to

Department of Electronics and Computer Engineering,

Pulchowk Campus, IOE

Source Code :- Lagrange's Method

```
#include <iostream>
using namespace std;

int main(){
    //Taking points as input
    int n;
    cout<<"Enter the number of points given: ";
    cin >> n;

    float x[n], y[n];
    for (int i=0; i<n; i++){
        cout<<"Enter x["<<i<<"] y["<<i<<"]: ";
        cin >> x[i] >> y[i];
    }

    float xp;
    cout<<"Enter the value of x at which y(x) is to
        be evaluated: ";
    cin >> xp;

    //Main logic
    float sum=0;
    for (int i=0; i<n; i++){
        float product=1;
        for (int j=0; j<n; j++){
            if(i!=j) product *= (xp-x[j])/(x[i]-x[j]);
        }
        sum += y[i] * product;
    }

    // Displaying result

    cout<<endl<<"RESULT : x["<<xp<<"] = "<<sum<<endl;
    return 0;
}
```

:- Source code :- RK2 Method

```
#include <iostream>
using namespace std;
float f(float x, float y) {
    return x*x - 2*x +5; // Function of slope
}
int main() {
  //Taking user inputs
   float x0, y0, xn;
   int n;

   cout <<" Enter x0, y0, xn and n: ";

   cin >> x0>>y0>>xn>> n;

   //Setting h= interval
   const float h= (xn-x0)/n;

   cout<< "The points on the required curve are:"<< endl;
   cout << "("<< x0 << "," << y0 << ")\n";

   for( int i=0; i<n; i++) {
       float m1 = f(x0, y0);
       float m2 = f(x0+h, y0+ m1*h);
       float m = (m1+m2)/2;
       y0 += h*m;
       x0 += h;
       cout<< "(" << x0 << "," << y0 <<")\n";
   }
   return 0;
}
```

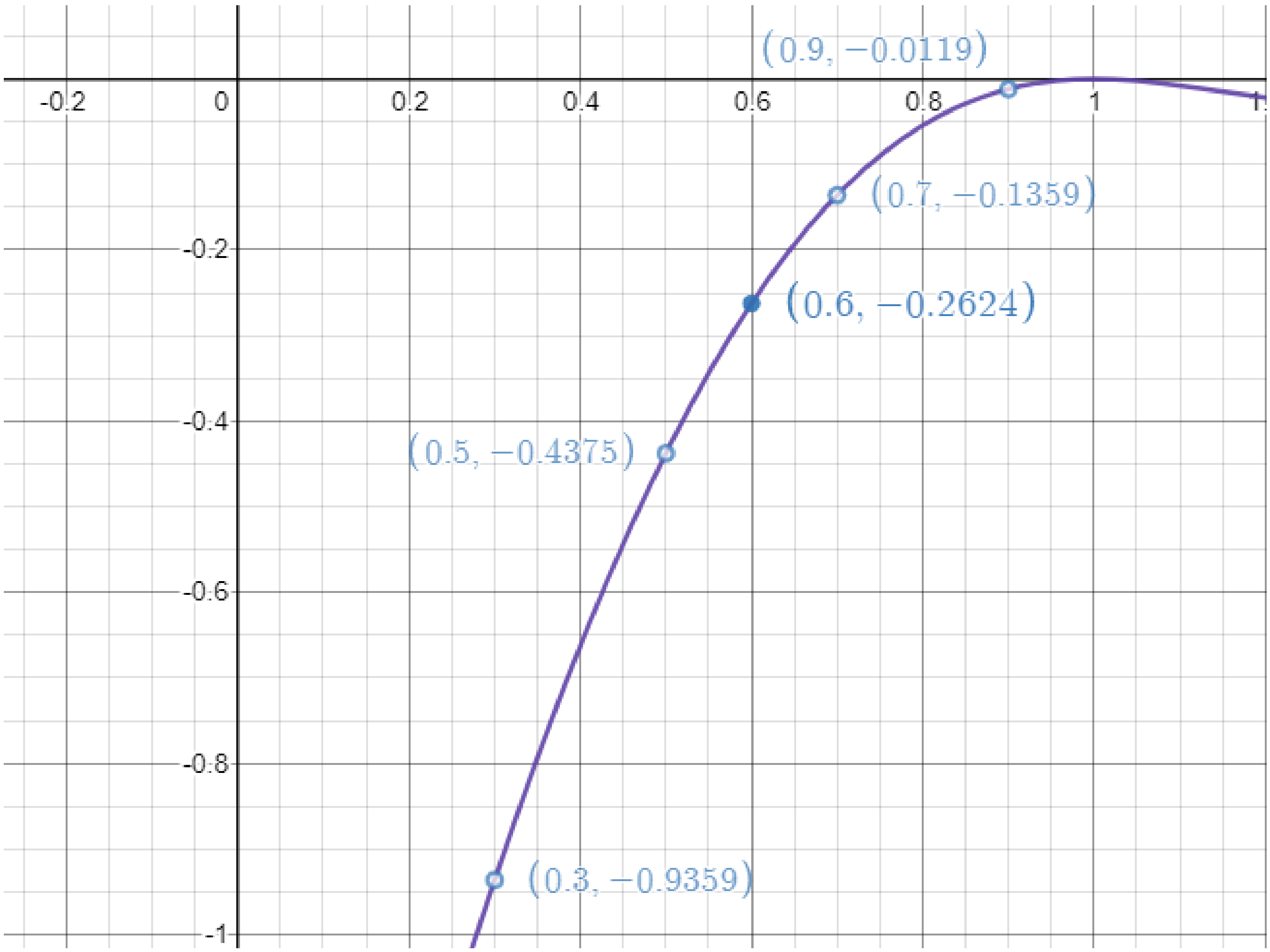
Source Code :- RK 4

```
#include <iostream>
using namespace std;

float f(float x, float y){
    return x*x - 2*x +5;
}

int main(){
    float x0, y0, xn;
    int n;
    cout<< " Enter
    cin >> x0 >> y0 >> xn >> n;
    const float h = (xn - x0)/n;
    cout << "The points on the required curve are : " << endl;
    cout<< "(" << x0<< "," << y0<< ")\n";
    for( int i=0; i<n; i++){
        float m1= f(x0, y0);
        float m2= f(x0+h/2, y0+m1*h/2);
        float m3 = f(x0+h/2, y0+m2*h/2);
        float m4 = f(x0+h, y0+ m3*h);
        float m = (m1+ 2*m2 + 2*m3+ m4)/6;
        y0 += h*m;
        x0 += h;

        cout<<"(" <<x0 <<"," <<y0 <<")\n";
    }
    return 0;
}
```

Output

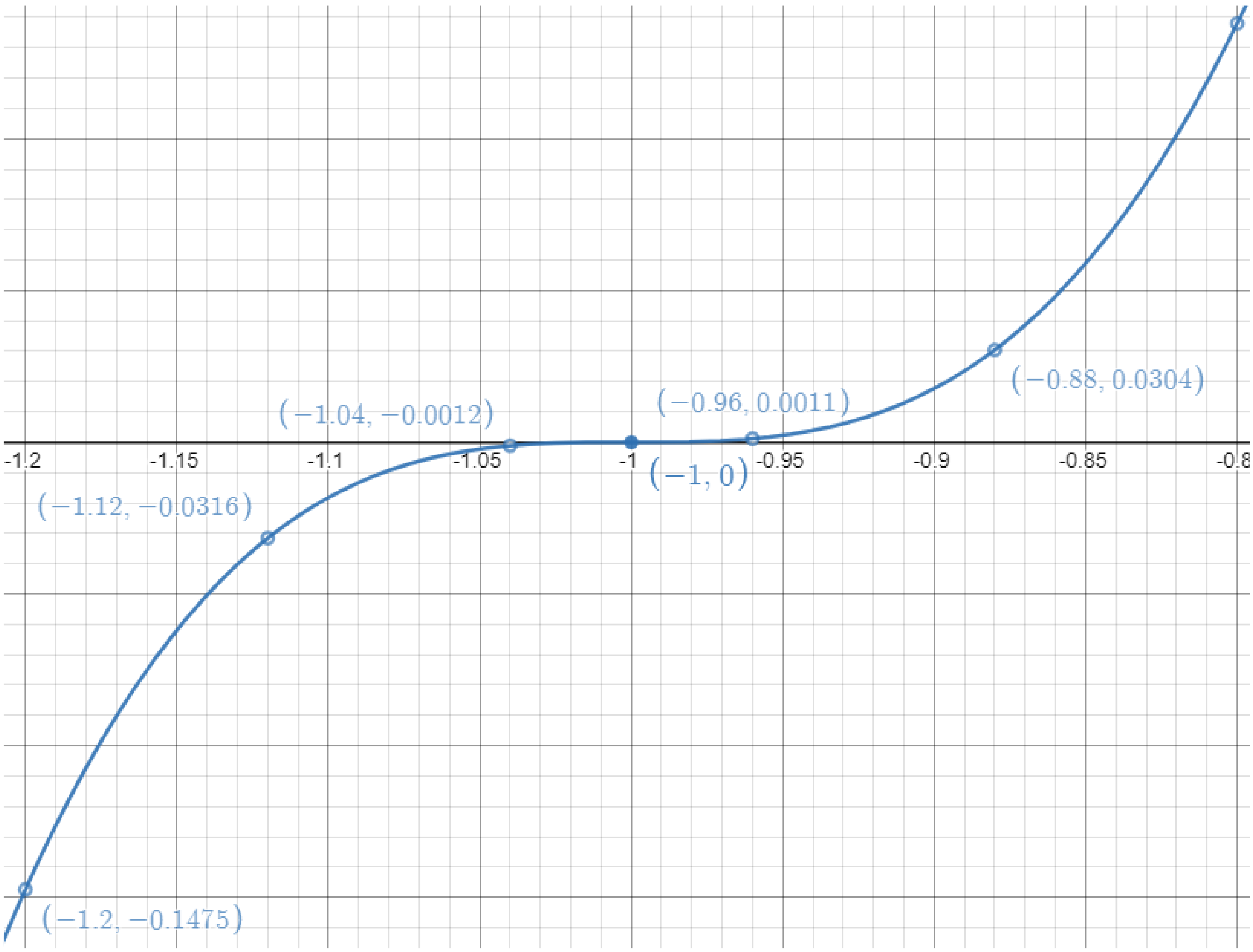
Lagrange’s Interpolation:

1. $y = x^4 - 2x^3 - x^2 + 4x^2$



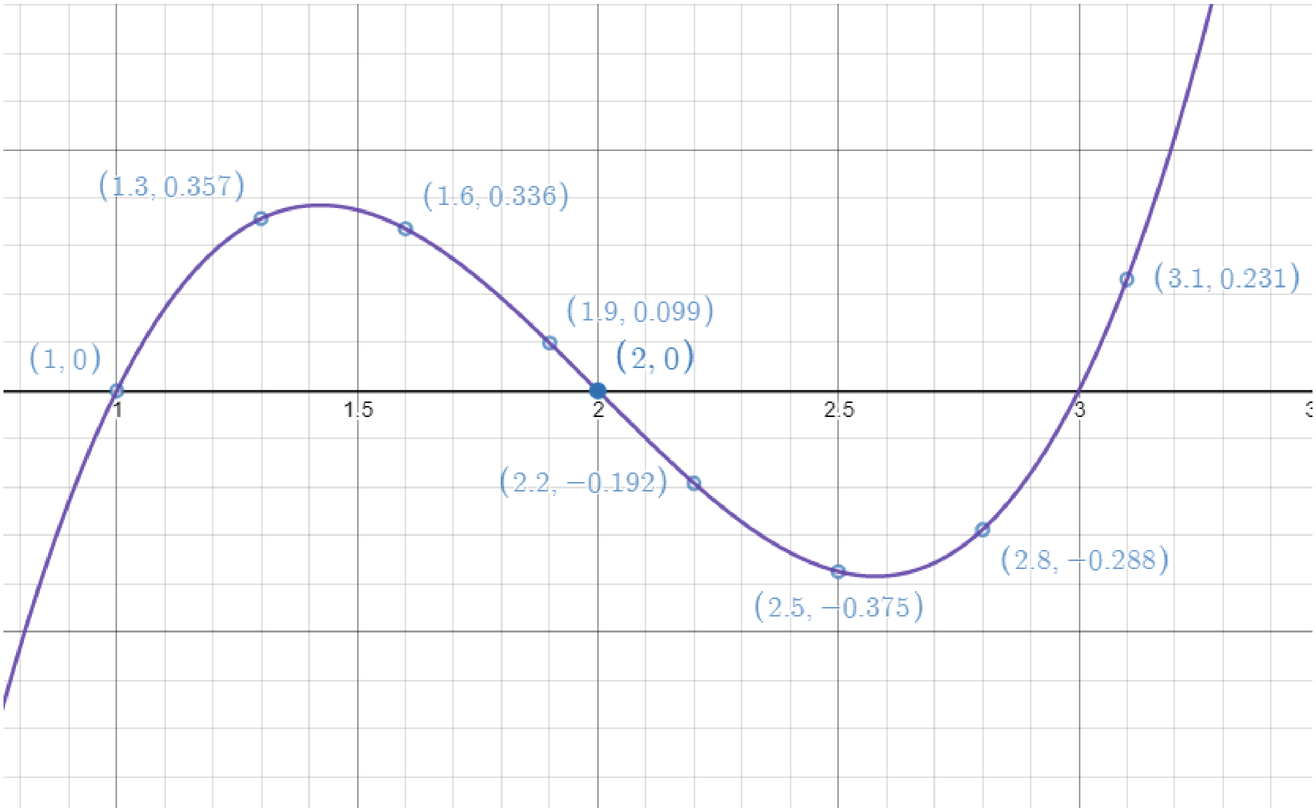

```
Enter the number of points given: 5
Enter x[0] y[0]: 0.1    -1.6119
Enter x[1] y[1]: 0.3    -0.9359
Enter x[2] y[2]: 0.5    -0.4375
Enter x[3] y[3]: 0.7    -0.1359
Enter x[4] y[4]: 0.9    -0.0119
Enter the value of x at which y(x) is to be evaluated: 0.6

RESULT: x[0.6] = -0.2624
```

2. $y = x^4 - 2x^3 - x^2 + 4x^2$




```
Enter the number of points given: 6
Enter x[0] y[0]: -1.2   -0.147456
Enter x[1] y[1]: -1.12  -0.031623561
Enter x[2] y[2]: -1.04  -0.001159266
Enter x[3] y[3]: -0.96  0.001143914
Enter x[4] y[4]: -0.88  0.030385373
Enter x[5] y[5]: -0.8   0.137984
Enter the value of x at which y(x) is to be evaluated: -1

RESULT: x[-1] = 0
```

3. $y = (x-1)(x-2)(x-3)$




```
Enter the number of points given: 8
Enter x[0] y[0]: 1      0
Enter x[1] y[1]: 1.3    0.357
Enter x[2] y[2]: 1.6    0.336
Enter x[3] y[3]: 1.9    0.099
Enter x[4] y[4]: 2.2    -0.192
Enter x[5] y[5]: 2.5    -0.375
Enter x[6] y[6]: 2.8    -0.288
Enter x[7] y[7]: 3.1    0.231
Enter the value of x at which y(x) is to be evaluated: 2

RESULT: x[2] = 0
```

Outputs for RK2 and RK4; and comparing them with Euler's method:

$\frac{dy}{dx} = 3x^2 + x + 1/50$ with initial point $(0, 0)$

$=> y = x^3 + 0.5x^2 + x/50$

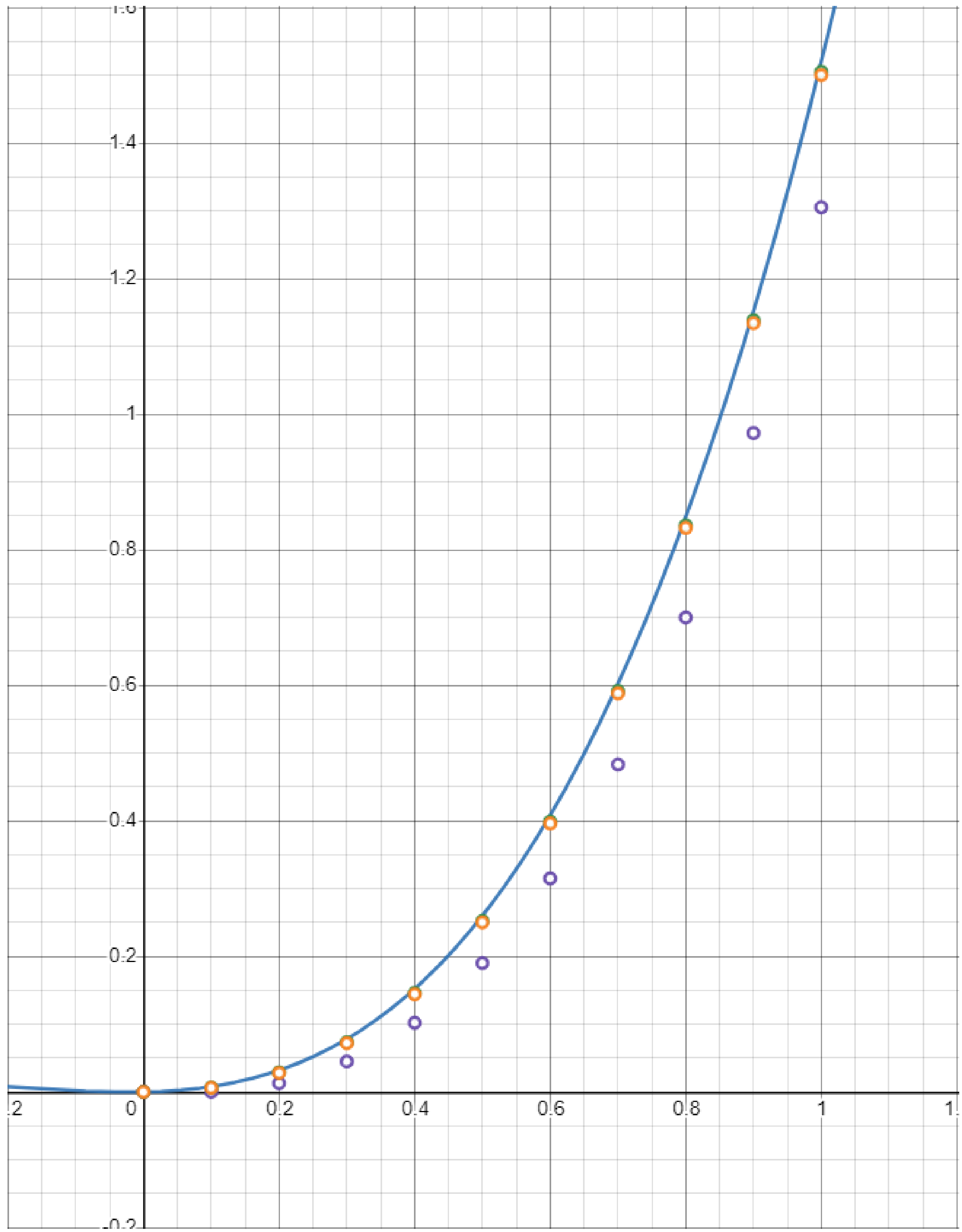
1. RK 2:

```
Enter x0, y0, xn and n: 0 0 1 10
The points on the required curve are:
(0,0)
(0.1,0.0065)
(0.2,0.029)
(0.3,0.0735)
(0.4,0.146)
(0.5,0.2525)
(0.6,0.399)
(0.7,0.5915)
(0.8,0.836)
(0.9,1.1385)
(1,1.505)
```

2. RK 4:

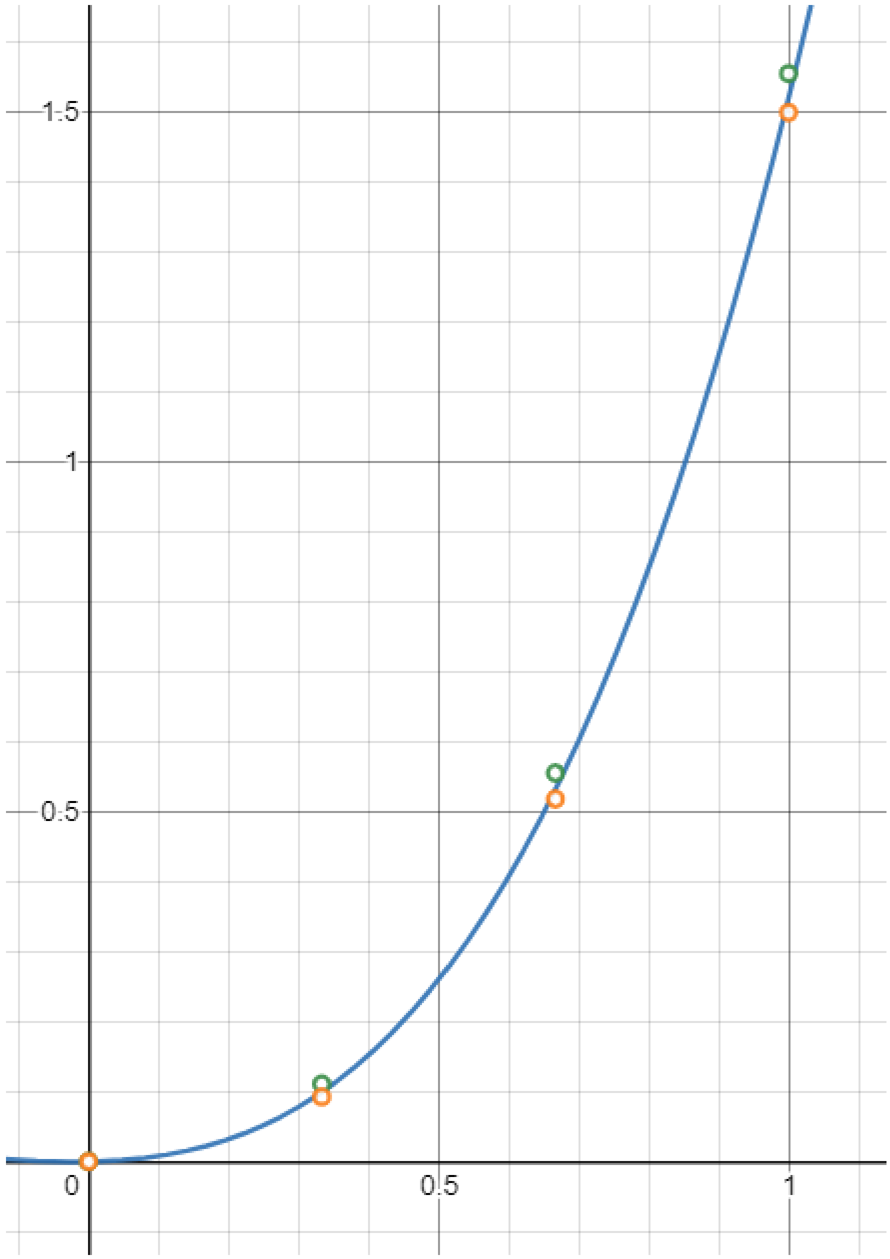
```
Enter x0, y0, xn and n: 0 0 1 10
The points on the required curve are:
(0,0)
(0.1,0.006)
(0.2,0.028)
(0.3,0.072)
(0.4,0.144)
(0.5,0.25)
(0.6,0.396)
(0.7,0.588)
(0.8,0.832)
(0.9,1.134)
(1,1.5)
```

Graph:



Here, the purple points are the result obtained from Euler's method; the green points are the result obtained from RK 2 method; and the orange points are the results obtained from RK 4 method.

We can clearly see that the results obtained from RK-2 and RK-4 methods are much more accurate than those obtained from the Euler's method. Also, it is seen that the orange and green points almost overlap each other, but if we decrease the number of intermediate points to just 3 we can clearly differentiate the accuracy of RK 2 and RK 4 methods, as seen from the graph below.



From this graph, it is clear that the accuracy of RK 4 is greater than that of Rk 2. Hence, in terms of accuracy,

RK 4 > RK 2 > Euler's Method