# RAJSHAHI UNIVERSITY OF ENGINEERING AND TECHNOLOGY

**Lab report: 09**

**Date of Experiment: 22.04.2018**
**Date of Submission: 07.05.2018**


**Submitted to:**
Shyla Afroge
Assistant Professor,
Department of Computer Science and Engineering
Rajshahi University of Engineering and Technology

**Submitted by:**
Riyad Morshed Shoeb
Roll No: 1603013
Section: A
Department of Computer Science and Engineering
Rajshahi University of Engineering and Technology

**Name of the Experiment: Implementation of Numerical Solution of Ordinary Differential Equations by Taylor's Series.**

**Theory:**

Consider the differential equation,

$$y' = f(x, y)$$

with the initial condition,

$$y(x_0) = y_0$$

If y(x) is the exact solution of the equations above, then Taylor's series for y(x) around $x = x_0$ is given by,

$$y(x) = y_0 + (x - x_0)y_0' + \frac{(x - x_0)^2}{2!}y_0'' + \cdots$$

If the values of $y_0', y_0''$, ... are known, then the equation gives a power series for y. Using the formula for total derivatives, we can write,

$$y'' = f' = f_x + y'f_y = f_x + ff_y$$

where, the suffixes denote partial derivatives with respect to the variable concerned. Similarly, we obtain,

$$\begin{aligned} y''' = f'' &= f_{xx} + f_{xy}f + f\left(f_{yx} + f_{yy}f\right) + f_y\left(f_x + f_yf\right) \\ &= f_{xx} + 2ff_{xy} + f^2f_{xy} + f_xf_y + ff_y^2 \end{aligned}$$

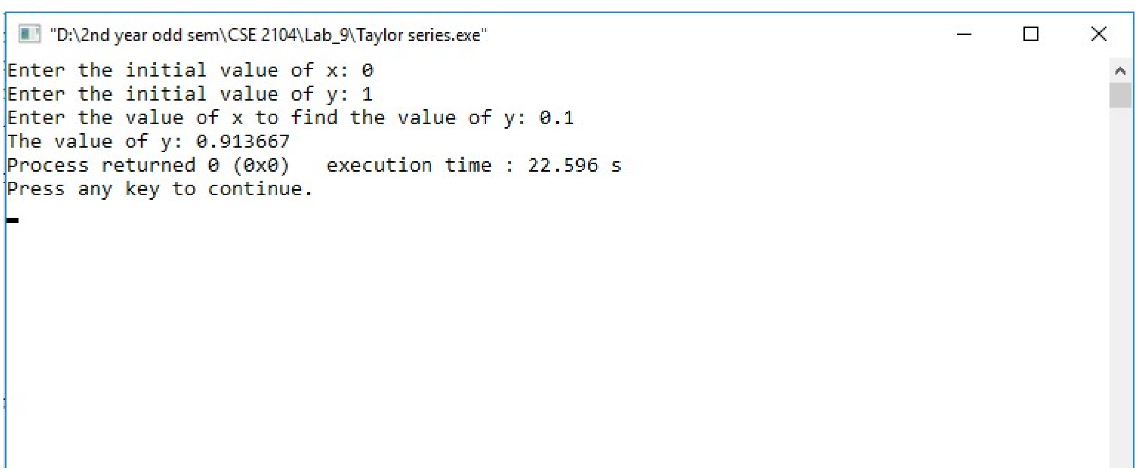
and other higher derivatives of y. The method can easily be extended to simultaneous and higher order differential equations.

**Code:**

```
#include<iostream>
#include<cstdio>
#include<cstdlib>
#include<cmath>
#include<vector>
using namespace std;

int fact(int n)
{
	if(n>1)
		return n*fact(n-1);
	else
		return 1;
}

double d_y(double x, double y)
{
	return (x-y*y);
}

double d2_y(double x, double y)
{
	return (1-2*y*d_y(x,y));
}

double d3_y(double x, double y)
{
	return (-2*y*d2_y(x,y)-2*d_y(x,y)*d_y(x,y));
```

```
}

int main(void)
{
	double x_initial;
	double y_initial;
	double find_x;
	double find_y;
	int i;
	vector<double>y;

	printf("Enter the initial value of x: ");
	cin>>x_initial;
	printf("Enter the initial value of y: ");
	cin>>y_initial;

	y.push_back(d_y(x_initial, y_initial));
	y.push_back(d2_y(x_initial, y_initial));
	y.push_back(d3_y(x_initial, y_initial));

	printf("Enter the value of x to find the value of y: ");
	cin>>find_x;

	find_y=1;
	for(i=0;i<3;i++)
	{
		find_y+=((pow(find_x,i+1)*y[i])/fact(i+1));
	}

	printf("The value of y: ");
	cout<<find_y;
}
```

**Output:**



**Name of the Experiment: Implementation of Numerical Solution of Ordinary Differential Equations by Euler's Method.**

**Theory:**

Suppose that we wish to solve $y' = f(x,y)$ for values of y at $x = x_r = x_0 + rh\ (r = 1,2,\ldots)$. Integrating the equation, we get,

$$y_1 = y_0 + \int_{x_0}^{x_1} f(x,y)dx$$

Assuming that $f(x,y) = f(x_0,y_0)\ in\ x_0 \le x \le x_1$, this gives Euler's formula

$$y_1 = y_0 + hf(x_0,y_0)$$

Similarly, for the range $x_0 \le x \le x_1$, we have,

$$y_2 = y_1 + \int_{x_1}^{x_2} f(x,y)dx$$

Substituting $f(x_1,y_1)\ for\ f(x,y)\ in\ x_0 \le x \le x_1$, we obtain,

$$y_2 = y_1 + hf(x_1,y_1)$$

Proceeding in this way, we obtain the general formula,

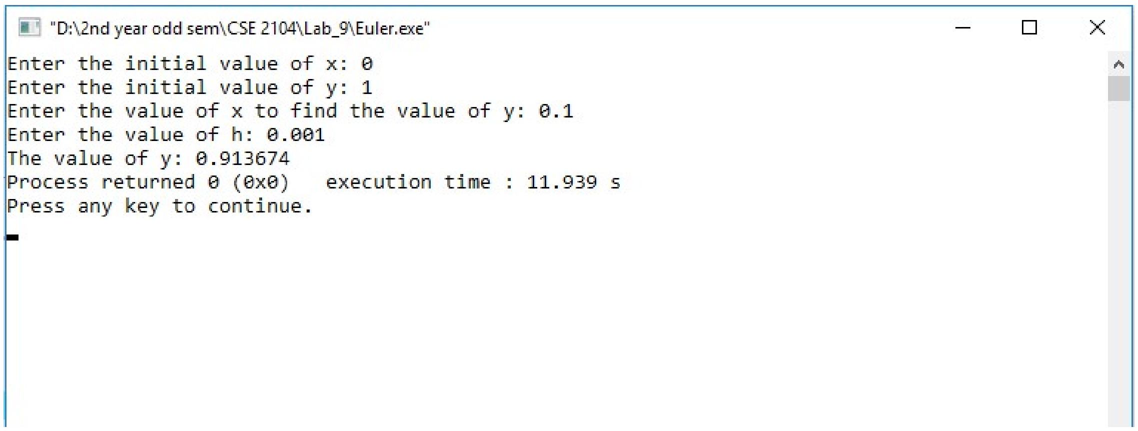
$$y_{n+1} = y_n + hf(x_n,y_n), \quad n = 0,1,2,\ldots$$

**Code:**

```
#include<iostream>
#include<cstdio>
#include<cstdlib>
#include<vector>
using namespace std;

double f(double x, double y)
{
      return (x-y*y);
}

int main(void)
{
      double x;
      vector<double>y;
      double find_x;
      double find_y;
      double h, temp;
      int i=0;

      printf("Enter the initial value of x: ");
      cin>>x;
      printf("Enter the initial value of y: ");
      cin>>temp;
      y.push_back(temp);
      printf("Enter the value of x to find the value of y: ");
      cin>>find_x;
      printf("Enter the value of h: ");
      cin>>h;

      while(x<=find_x)
      {
```

```
            temp=y[i]+h*f(x,y[i]);
            y.push_back(temp);
            x+=h;
            i++;
        }

        printf("The value of y: ");
        cout<<y[i];
}
```

**Output:**




**Discussion:**

Euler's method gives more accurate result then the Taylor's series. But the process is very slow and to obtain reasonable accuracy with Euler's method, we need to take a smaller value for h. Because of this restriction on h, the method is unusable for practical use.