МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Інститут прикладної математики та фундаментальних наук

Кафедра прикладної математики



Звіт

про виконання лабораторної роботи №1

з курсу “Чисельні методи частина 2”

на тему:

«ОДНОКРОКОВІ МЕТОДИ ЧИСЕЛЬНОГО РОЗВ’ЯЗУВАННЯ ЗАДАЧІ КОШІ ДЛЯ ЗВИЧАЙНИХ ДИФЕРЕНЦІАЛЬНИХ РІВНЯНЬ»

*Виконав:*

*студент гр. ПМ-31*

*Дах Н. І.*

*Прийняв:*

*доцент*

*Пізюр Я. В.*

# Львів-2016

**Зміст**

1. Постановка задачі
2. Аналітичний розв’язок задачі
3. Алгоритм для конкретного чисельного методу
4. Текст програми
5. Результати розв′язування на ЕОМ заданої задачі

Варіант 4

**Постановка задачі**





Використовуючи мову програмування C#, написав та відлагодив програму чисельного розв’язування задачі Коші для ЗДР методом Рунге-Кутта з автоматичним вибором кроку при заданій точності  та .

**Аналітичний розв’язок задачі**

**> restart;**

**> **

**> **





**> **



**> **



**> **



**> **



**> **



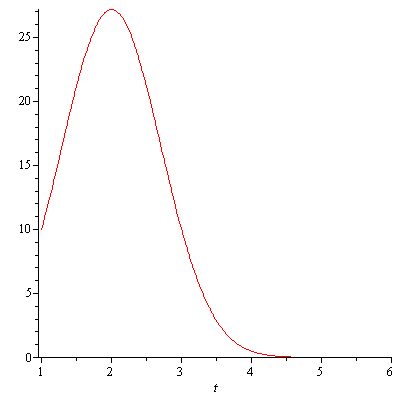
**> **



**> **



**> **



**> **



**> **



**> **



**> **



**Алгоритм для конкретного чисельного методу**

В своєму варіанті я використовував метод Рунге-Кута другого порядку точності, за ось таким алгоритмом:

1. **Ввести значення** , , , , , .

2**. Ініціалізувати змінні** ; ; ; .

**Вивести значення** ; ; ; .

3**. If**  **then go to** 13.

4**. If**  **then** .

5. ; .

6. .

7. **Обчислити** .

8. **Обчислити**

9. **If**  **then**

**begin** ; ; ; ; **go to 8 end**.

10. **If**  **then**

**begin** ; ; **go to 7 end**.

11. **Обчислити**

, .

12. **If**  **then**

**begin** ; ; обчислити точний розв’язок ; ;

вивести значення ; ; ; ;

**if**  **then** ;

**go to 3**

**end**;

**else**

**begin** ; ; ; **go to 6 end**.

13. **Вивести норму похибки** .

14. **End**.

**Текст програми**

using System;

namespace ConsoleApplication6

{

class Program

{

public static double t, Y, Thau, T, eps, Emax;

public static int N = 3;

public static double U(double t) {

double Ut = (-(Math.Exp(-4.0 \* t)) + (4000.0 + Math.Exp(-1.0)) / (Math.Exp(3.0))) \* Math.Exp(-t \* (t - 4.0))/400.0;

return Ut;

}

public static double g(double t) {

return -2.0 \* (t - 2.0);

}

public static double phi(double t) {

return Math.Exp(-Math.Pow(t,2.0));

}

public static double psi(double t) {

return 0.01;

}

public static void Initialization()

{

Console.Write("t0 = ");

t = Double.Parse(Console.ReadLine());

Console.Write("U0 = ");

Y = Double.Parse(Console.ReadLine());

Console.Write("Thau0 = ");

Thau = Double.Parse(Console.ReadLine());

Console.Write("T = ");

T = Double.Parse(Console.ReadLine());

Console.Write("eps = ");

eps = Double.Parse(Console.ReadLine());

Emax = 0;

}

public static double f(double t, double Y) {

return phi(t) \* psi(t) + g(t) \* Y;

}

static void Main(string[] args)

{

Initialization();

double k1=0.0, k2=0.0, w=0.0, nev=0.0, t1=0.0;

int kf = 1;

Console.WriteLine("\t\t" + "t" + "\t\t" + "y" + "\t\t" + "U(t)" + "\t\t" + "|y-U(t)|");

Console.WriteLine("\n");

Console.Write("\t " + t.ToString("#.#####") + "\t\t " + Y.ToString("#.#####") + "\t\t " + U(t).ToString("#.#####") + "\t\t" + (Math.Abs(U(t)-Y)).ToString("#.######") + "\n");

while ((T - t) > Emax)

{

if (kf == 1)

{

k1 = 0; k2 = 0; w = 0;

if ((t + Thau) > T) Thau = T - t;

nev = Y;

t1 = t;

}

for (int i = 0; i < 3; ++i)

{

if (i == 0 || i == 2)

{

k1 = f(t, Y);

}

k2 = f((t + (Thau \* 0.5)), (Y + (Thau \* k1 \* 0.5)));

Y = Y + Thau \* k2;

if (i == 0)

{

Thau = Thau \* 0.5; w = Y; Y = nev;

}

if (i == 1) t = t + Thau;

}

double E = Math.Abs(Y - w) / (3.0 \* (Math.Max(1.0, Math.Abs(Y))));

double ThauH = 2.0 \* Thau \* Math.Min(5.0, Math.Max(0.1, (0.9 \* Math.Pow((eps / E), (1.0 / 3.0)))));

if (E < eps)

{

t = t + Thau;

Y = Y + (Y - w) / 3.0;

double diff = Math.Abs(Y - U(t));

Thau = ThauH;

if (Emax < diff) Emax = diff;

Console.Write("\t " + t.ToString("#.######") + "\t " + Y.ToString("#.######") + "\t " + U(t).ToString("#.######") + "\t\t" + diff.ToString("#.#######") + "\n");

kf = 1;

}

else

{

Y = nev;

t = t1;

Thau = ThauH;

kf = 0;

}

}

Console.WriteLine("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ERROR\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

Console.WriteLine("\t\t\t\tEmax = " + Emax);

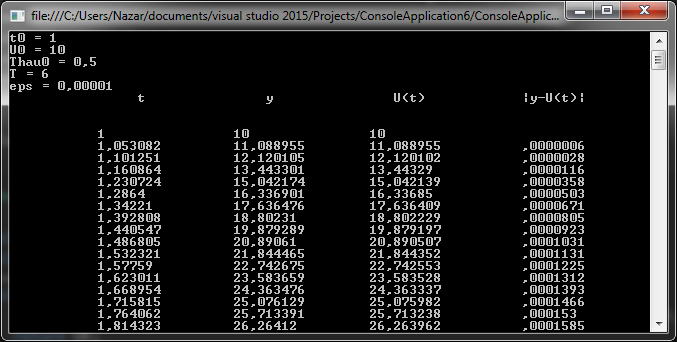
Console.ReadLine();

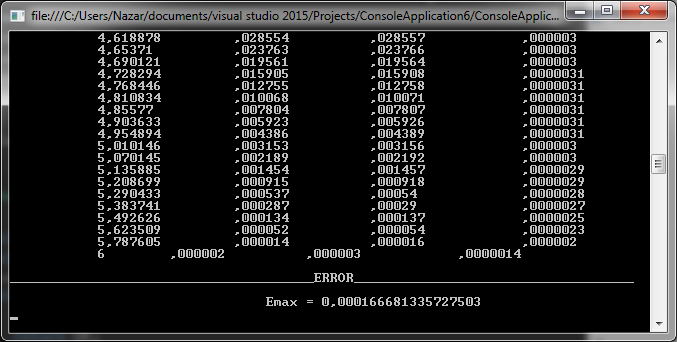
}

}

}

**Результат виконання програми**





**Висновок**

В цій лабораторній роботі я навчився: з допомогою мови програмування C# розв’язувати ЗДР методом Рунге-Кутта другого порядку точності, з автоматичним вибором довжини кроку, для заданої точності.