НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА СИСТЕМНОГО ПРОГРАМУВАННЯ ТА СПЕЦІАЛІЗОВАНИХ КОМП’ЮТЕРНИХ СИСТЕМ

**Лабораторна робота №4  
з дисципліни «Алгоритми та методи обчислення»  
на тему: «ЧИСЕЛЬНЕ ІНТЕГРУВАННЯ»**

**Варіант 21**

Виконав:  
студент 3-го курсу,  
гр. КВ-41,  
Яковенко Максим

Київ – 2016

**Завдання для лабораторної роботи**

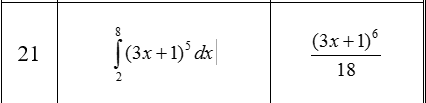
1. Відповідно до варіанту (табл. 4.2) за допомогою залишкового члена АНАЛІТИЧНО визначити крок інтегрування, що забезпечує необхідну точність  (визначається у тексті програми). Обчислити інтеграл  з кроком *h* і визначити абсолютну похибку Δ, прийнявши за точне значення, обчислене за формулою Ньютона-Лейбніца.

2. За допомогою методу подвійного перерахунку досягти тієї ж похибки Δ, що й у п.1. Вивести значення отриманого кроку й абсолютної похибки.

Звіт про лабораторну роботу має містити вихідний текст програми, таблицю з результатами та висновки.

Примітка. У варіантах з парним номером необхідно використовувати узагальнену формулу трапецій, у варіантах з непарним номероам - узагальнену формулу Сімпсона.

**Варіант 21:**



**Текст програми**:

#include <cmath>

#include <stdio.h>

double func(double x)

{

return pow((3\*x+1),5);

}

double funcFour(double x)

{

return 9720\*(3\*x+1);

}

double funcPerv(double x)

{

return pow((3 \* x + 1), 6)/18;

}

double Integrate\_Simpson(double n, double a, double b)

{

int i;

double h;

double sig1 = 0.0;

double sig2 = 0.0;

n = 2\*ceil(n/2);

h = (b - a)/n;

for (i = 1; i < n; i++)

{

if (i%2 == 0)

sig2 += func(a + i\*h);

else

sig1 += func(a + i\*h);

}

return (h/3)\*(func(a) + func(b) + 4\*sig1 + 2\*sig2);

}

int main()

{

double n;

double a = 2.0, b = 8.0;

double e = 0.0001;

double r, h, max;

double I0, In, I2n;

double x, delta;

I0 = funcPerv(b) - funcPerv(a);

printf("[a, b] = [%.1f, %.1f]\n", a, b);

printf("I = F(b) - F(a) = %f\n\n", I0);

printf("======================================================\n");

printf("| e | h | I | delta |\n");

printf("======================================================\n");

x = a;

max = funcFour(x);

while (x < b)

{

if (funcFour(x) > max) max = funcFour(x);

x += e;

}

h = pow(180\*e/(b - a)/max, 0.25);

n = 2\*ceil(0.5\*(b - a)/h);

h = (b - a)/n;

delta = fabs(I0 - Integrate\_Simpson(n, a, b));

printf("| %f | %.3f | %.8f | %.9f |\n", e, h, Integrate\_Simpson(n, a, b), delta);

printf("======================================================\n");

printf("=====================================\n");

printf("| delta | h | Abs |\n");

printf("=====================================\n");

n = ceil(1/pow(e, 0.25));

In = Integrate\_Simpson(n, a, b);

I2n = Integrate\_Simpson(2\*n, a, b);

r = fabs(In - I2n)/3;

while (r > delta)

{

n \*= 2;

In = Integrate\_Simpson(n, a, b);

I2n = Integrate\_Simpson(2\*n, a, b);

r = fabs(In - I2n)/3;

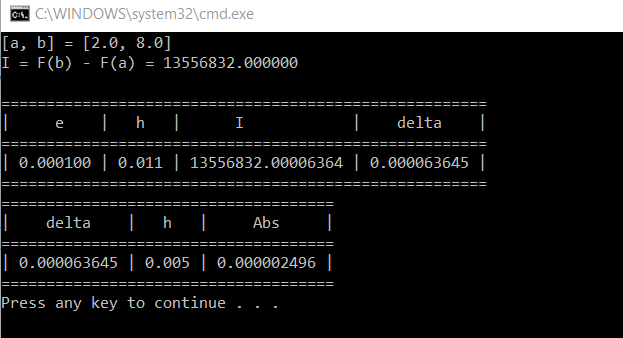
}

printf("| %.9f | %.3f | %.9f |\n", delta, 0.5\*(b - a)/n, fabs(I0 - I2n));

printf("=====================================\n");

}

**Таблиці результатів:**

****

**Висновки:**

В ході виконання лабораторної роботи ми досліджували методи числового інтегрування, а саме метод Сімпсона, та метод подвійного перерахунку.   
 До чисельного інтегрування звертаються коли підінтегральна функціє задана таблично чи відшукання первісної є занадто складною задачею. Розрізняють інтерполяційні та неінтерполяційні методи інтегрування. Оскільки інтерполяційний метод доцільно використовувати в разі невеликих проміжків інтегрування, то на практиці найчастіше використовуються узагальнені формули чисельного інтегрування.   
 Одним з цих методів є метод Сімпсона, принцип роботи якого заключається в тому, щоб поділити проміжок інтегрування на m=2n (парне число) і визначити значення на кожному з малих проміжків. Цей метод є доволі точним, хоч і вимагає для оцінки точності використовувати 4ту похідну функції, що іноді є доволі складною задачею.

Тому часто використовують метод подвійного перерахунку(принцип Рунге), який дозволяє значно простіше отримати задану точність. Його принцип полягає в тому, що спочатку береться якесь початкове n і оцінується похибка. Якщо похибка більша від заданої, то n=2n.