МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра КИТ

**Звіт  
по навчальній практиці**

Виконав: студент гр. ПЗ1911

Сіньков Г.О.

Дніпро, 2019

**Постановка задачі**

Програма повинна забезпечувати введення наступних даних: границь проміжку інтегрування функції, кількість відрізків для обчислення інтеграла, вибір підінтегральної функції та метода її інтегрування. Програма обчислює значення інтегралу; значення виводяться з точністю до шести знаків. В доповнення може будуватись графік функції та графічне представлення методу інтегрування.

Перелік методів чисельного інтегрування функцій:

* метод лівих прямокутників;
* метод правих прямокутників;
* метод середніх прямокутників;
* метод трапецій;
* метод парабол (метод Сімпсона).

|  |  |
| --- | --- |
| Вимоги до оформлення програми | Вимоги до змісту завдання |
| Консольний додаток з контролем вхідної інформації. Методи та функції реалізовані у вигляді окремих функцій та знаходяться в окремому файлі. | Для набору мінімум з трьох різних функцій повинні бути реалізовані всі методи. |

**Зовнішні специфікації**

Формат вхідних даних

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Найменування даних | Умовне позначення | Вимоги до даних | Приклад |
| 1 | Нижня межа інтегрування | a | Дійсне число з області визначення функції | 2 або 2,56 |
| 2 | Верхня межа інтегрування | b | Дійсне число з області визначення функції | 4 або 2,56 |
| 3 | Метод рішення | number\_metod | Ціле позитивне число в діапазоні від 1 до 3 | 1 |
| 4 | Номер функції | number\_fun | Ціле позитивне число в діапазоні від 1 до 3 | 2 |
| 5 | Підінтегральна функція | F(x) | Рядок, довжиною не більше 10 латинського алфавіту символів |  |
| 6 | Кількість розбиття | n | Натуральне число | 100 |
| 7 | Точність обчислення інтеграла | epsilon | Дійсне число на інтервалі від 0 до 1, виключаючи межі інтервалу | 0.001 |

При завданні виду підінтегральної функції можуть використовуватися знаки арифметичних операцій (+, -, \*, /), операція піднесення до степеня ^, круглі дужки, позначення елементарних математичних функцій (sin, cos, ln і т.д.). В якості аргументу функції використовується х. Складати функцію необхідно з урахуванням правил записи математичних виразів і пріоритету операцій.

Формат вихідних даних

|  |  |  |
| --- | --- | --- |
| № | Вхідні дані | Тестове повідомлення |
| 1 | Вивід відповіді | Значение интеграла (метод) равно: 2.59066 |

**Функціональні вимоги до програми**

Програма повинна реалізувати наступні дії:

-Обчислення значення певного інтеграла із заданою точністю

-Забезпечувати введення користувачем вихідних даних для обчислень, з контро-лем вхідної інформації

-Вибір методу чисельного інтегрування

-Побудова графіка підінтегральної функції

-Збереження результатів обчислень відповідно до формату

-Надавати користувачеві довідкову інформацію про використані методах і порядку роботи з програмою.

**Вибір методу рішення завдання**

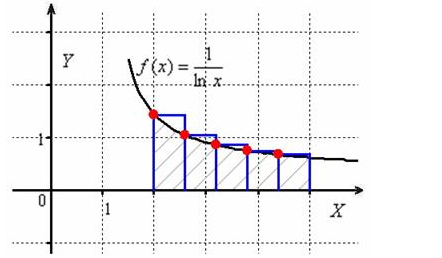
1)Метод прямокутників

Алгоритм знаходження інтеграла f(x) за лівими прямокутниками

Для початку знаходимо крок розбиття за формулою:

, де a і b – нижня та верхня межа інтегрування, n – кількість прямокутників за якими будемо знаходити площу (чим більше n тим точніше можна знайти інтеграл)

Метод лівих прямокутників отримав своє назва через те,

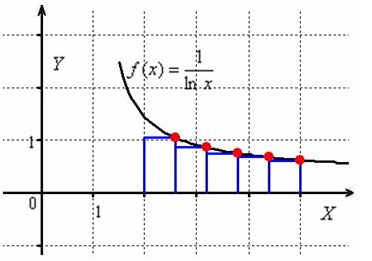


що висота прямокутників на проміжних відрізках дорівнюють значенням функції в лівих кінцях даних відрізків. Підставляємо точки, які отримали при розбитті, у нашу функцію

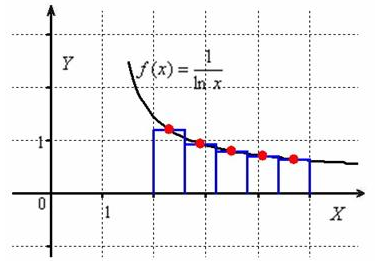
Після того як знайшли значення у всіх точках ми можемо знайти інтеграл даної функції за формулою

Для методі правих та середніх прямокутників все тож саме але кінцева формула трохи відрізняється:

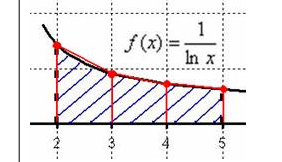
– правих прямокутників



– середніх прямокутників



2)Метод трапеції

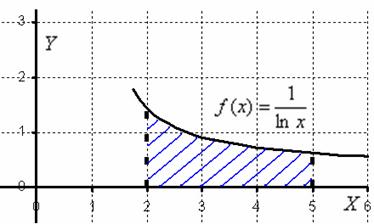


Для початку знаходимо крок розбиття за формулою:

Тоді певний інтеграл можна обчислити наближено за формулою трапецій:

, де - значення підінтегральної функції в точках

3)Метод Симпмона



Розглянемо певний інтеграл , де - функція, безперервна на відрізку [a; b]. Проведемо розбиття відрізка [a; b] на парне кількість рівних відрізків. Парна кількість відрізків позначають через 2n.

На практиці відрізків може бути:

**два:** 2n

**чотири:** 2n = 4

**вісім:** 2n = 8

**десять:** 2n = 10

**дванадцять:** 2n = 20

**Увага!** Число 2n розуміється як ЄДИНЕ ЧИСЛО. Тобто, **НЕ МОЖНА** скорочувати, наприклад, 2n = 8 на два, отримуючи n = 4. Запис 2n **лише позначає**, що кількість відрізків **парно**. І ні про які скорочення не йдеться.

Формула Сімпсона для наближеного обчислення певного інтеграла має наступний вигляд:

, де:

- довжина кожного з маленьких відрізків або крок;

- значення підінтегральної функції в точках

Деталізуючи це нагромадження, розберу формулу докладніше:

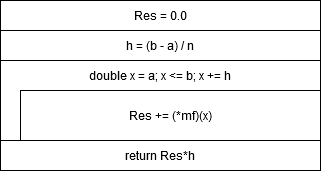
- сума першого і останнього значення підінтегральної функції;

- сума членів з парними індексами множиться на 2;

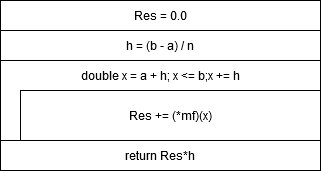
- сума членів з непарними індексами множиться на 4.

**Алгоритм програми**

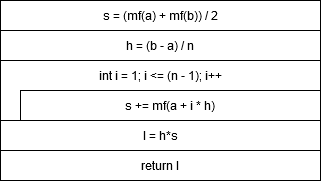
1)Метод лівих прямокутників(в \*mf передається функція)



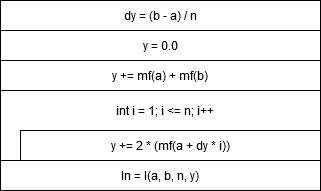
2) Метод правих прямокутників(в \*mf передається функція)



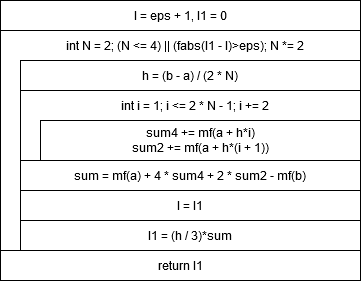
3) Метод середніх прямокутників(в \*mf передається функція)



4)Метод трапеції



5)Метод Симпсона



**Текст програми**

func.h

#ifndef func\_H

#define func\_H

typedef double(\*pointFunction)(double);

typedef double(\*pointMetod)(pointFunction, double, double, unsigned);

double left\_rectangle(pointFunction mf, double a, double b, unsigned n);

double right\_rectangle(pointFunction mf, double a, double b, unsigned n);

double middle\_rectangle(pointFunction mf, double a, double b, unsigned n);

void metod\_trapeze(pointFunction mf, double a, double b, unsigned n);

double metod\_parabol(pointFunction mf, double a, double b, unsigned n, double eps);

double Integral(pointFunction mf, pointMetod Metod, double a, double b, double eps, unsigned &n);

double function1(double x);

double function2(double x);

double function3(double x);

double function4(double x);

double function5(double x);

double input\_a();

double input\_b();

double input\_epsilon();

double input\_quantity\_n();

#endif

func.cpp

#include <iostream>

#include "func.h"

#include <math.h>

double function1(double x)

{

return 1 / log(x);

}

double function2(double x)

{

return log(1 + x);

}

double function3(double x)

{

return -3 \* x\*x + 2 \* x + 9;

}

double function4(double x)

{

return 2 \* pow(x, 3) - 7 \* x + 4;

}

double function5(double x)

{

return pow(x, 3) \* 12 - 5;

}

double I(double a, double b, unsigned n, double y)

{

return ((b - a) / (2 \* n)\*y);

}

double left\_rectangle(pointFunction mf, double a, double b, unsigned n)

{

double Res = 0.0;

double h = (b - a) / n;

for (double x = a; x <= b; x += h)

Res += (\*mf)(x);

return Res\*h;

}

double right\_rectangle(pointFunction mf, double a, double b, unsigned n)

{

double Res = 0.0;

double h = (b - a) / n;

for (double x = a + h; x <= b;x += h)

Res += (\*mf)(x);

return Res\*h;

}

double middle\_rectangle(pointFunction mf, double a, double b, unsigned n)

{

double s = (mf(a) + mf(b)) / 2;

double h = (b - a) / n;

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

{

s += mf(a + i \* h);

}

double I = h\*s;

return I;

}

void metod\_trapeze(pointFunction mf, double a, double b, unsigned n)

{

double y, dy, In;

dy = (b - a) / n;

y = 0.0;

y += mf(a) + mf(b);

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

{

y += 2 \* (mf(a + dy \* i));

}

In = I(a, b, n, y);

std::cout << "Значення інтеграла: " << In << std::endl;

}

double metod\_parabol(pointFunction mf, double a, double b, unsigned n, double eps)

{

double I = eps + 1, I1 = 0;//I-предыдущее вычисленное значение интеграла, I1-новое, с большим N.

for (int N = 2; (N <= 4) || (fabs(I1 - I)>eps); N \*= 2)

{

double h, sum2 = 0, sum4 = 0, sum = 0;

h = (b - a) / (2 \* N);//Шаг интегрирования.

for (int i = 1; i <= 2 \* N - 1; i += 2)

{

sum4 += mf(a + h\*i);//Значения с нечётными индексами, которые нужно умножить на 4.

sum2 += mf(a + h\*(i + 1));//Значения с чётными индексами, которые нужно умножить на 2.

}

sum = mf(a) + 4 \* sum4 + 2 \* sum2 - mf(b);//Отнимаем значение f(b) так как ранее прибавили его дважды.

I = I1;

I1 = (h / 3)\*sum;

}

return I1;

}

double Integral(pointFunction mf, pointMetod Metod, double a, double b, double eps, unsigned &n)

{

double Integral\_old, Integral\_new = 0.0;

Integral\_old = (\*Metod)(mf, a, b, n);

while (fabs(Integral\_old - Integral\_new)>eps)

{

Integral\_new = Integral\_old;

n \*= 2;

Integral\_old = (\*Metod)(mf, a, b, n);

}

return Integral\_old;

}

double input\_a()

{

double a;

std::cout << "Введіть ліву межу інтегрування a = ";

while (!(std::cin >> a) || (std::cin.peek() != '\n')) // цикл для перевірки введеної зміної на символи

{

//std::system("cls");

std::cin.clear();

while (std::cin.get() != '\n');

std::cerr << "Помилка вводу! a повино бути дійсним числом" << std::endl;

std::cout << std::endl;

std::cout << "Введіть ліву межу інтегрування a = ";

}

return a;

}

double input\_b()

{

double b;

std::cout << "Введіть праву межу інтегрування b = ";

while (!(std::cin >> b) || (std::cin.peek() != '\n')) // цикл для перевірки введеної зміної на символи

{

//std::system("cls");

std::cin.clear();

while (std::cin.get() != '\n');

std::cerr << "Помилка вводу! b повино бути дійсним числом" << std::endl;

std::cout << std::endl;

std::cout << "Введіть праву межу інтегрування b = ";

}

return b;

}

double input\_epsilon()

{

double epsilon;

std::cout << "Введіть необхідну точність epsilon = ";

while (!(std::cin >> epsilon) || (std::cin.peek() != '\n')) // цикл для перевірки введеної зміної на символи

{

//std::system("cls");

std::cin.clear();

while (std::cin.get() != '\n');

std::cerr << "Помилка вводу! epsilon повино бути дійсним числом" << std::endl;

std::cout << std::endl;

std::cout << "Введіть необхідну точність epsilon = ";

}

return epsilon;

}

double input\_quantity\_n()

{

double n;

std::cout << "Введіть кількість розбиття = ";

while (!(std::cin >> n) || (std::cin.peek() != '\n') || !(n > 1)) // цикл для перевірки введеної зміної на символи

{

//std::system("cls");

std::cin.clear();

while (std::cin.get() != '\n');

std::cerr << "Помилка вводу! n повино бути цілим числом та більшим за 1." << std::endl;

std::cout << std::endl;

std::cout << "Введіть кількість розбиття = ";

}

return n;

}

main.cpp

#include <iostream>

#include <Windows.h>

#include "func.h"

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

double a, b, epsilon, Integ;

double n; //начальное число шагов

int number\_fun, number\_metod;

pointFunction mf;

mf = 0;

do

{

system("cls");

std::cout << "Виберіть функцію: \n";

std::cout << "1. 1 / log(x);\n";

std::cout << "2. log(1 + x);\n";

std::cout << "3. -3\*x\*x + 2\*x + 9;\n";

std::cout << "4. 2\*pow(x, 3) - 7\*x +4;\n";

std::cout << "5. pow(x, 3) \* 12 - 5;\n";

std::cout << "6. Вихід.\n";

std::cin >> number\_fun;

switch (number\_fun)

{

case 1:

mf = function1;

break;

case 2:

mf = function2;

break;

case 3:

mf = function3;

break;

case 4:

mf = function4;

break;

case 5:

mf = function5;

break;

default:

std::cout << "You entered an invalid menu item!" << std::endl;

break;

}

if (number\_fun != 6)

{

do

{

system("cls");

std::cout << "Виберіть метод: \n";

std::cout << "1. Метод лівих прямокутників\n";

std::cout << "2. Метод правих прямокутників.\n";

std::cout << "3. Метод середніх прямокутників.\n";

std::cout << "4. Метод трапецій.\n";

std::cout << "5. Метод парабол (метод Сімпсона).\n";

std::cout << "6. Вихід в меню функцій\n";

std::cout << "7. Вихід з програми\n";

std::cin >> number\_metod;

switch (number\_metod)

{

case 1:

{

system("cls");

a = input\_a();

b = input\_b();

n = input\_quantity\_n();

Integ = left\_rectangle(mf, a, b, n);

std::cout << "Значение интеграла (метод л.прям.) равно: " << Integ << std::endl;

system("pause");

}

break;

case 2:

{

system("cls");

a = input\_a();

b = input\_b();

n = input\_quantity\_n();

Integ = right\_rectangle(mf, a, b, n);

std::cout << "Значение интеграла (метод п.прям.) равно: " << Integ << std::endl;

system("pause");

}

break;

case 3:

{

system("cls");

a = input\_a();

b = input\_b();

n = input\_quantity\_n();

Integ = middle\_rectangle(mf, a, b, n);

std::cout << "Значение интеграла (метод ср.прям.) равно: " << Integ << std::endl;

system("pause");

}

break;

case 4:

{

system("cls");

a = input\_a();

b = input\_b();

n = input\_quantity\_n();

metod\_trapeze(mf, a, b, n);

system("pause");

}

break;

case 5:

{

system("cls");

a = input\_a();

b = input\_b();

n = input\_quantity\_n();

epsilon = input\_epsilon();

Integ = metod\_parabol(mf, a, b, n, epsilon);

std::cout << "Значение интеграла равно:" << Integ << std::endl;

system("pause");

}

break;

case 7:

return 0;

default:

break;

}

} while (number\_metod != 6);

}

} while (number\_fun != 6);

}

**Розробка тестів**

Обчислимо крок розбиття(для всіх методів):

**Контрольний приклад**

Метод лівих прямокутників отримав своє назва через те, що висоти прямокутників на проміжних відрізках дорівнюють значенням функції в лівих кінцях даних відрізків:

;

Обчислимо площу ступінчастою фігури, яка дорівнює сумі площ прямокутників:

Таким чином, площа криволінійної трапеції:

Площа криволінійної трапеції(праві прямокутники):

Площа криволінійної трапеції(середніх прямокутники):

Площа криволінійної трапеції(метод трапецій):

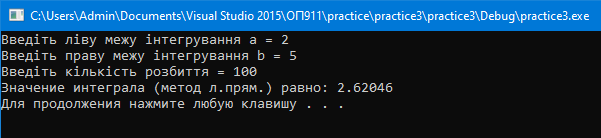
Площа криволінійної трапеції(метод Симпсона):

Вхідні дані для тестів:

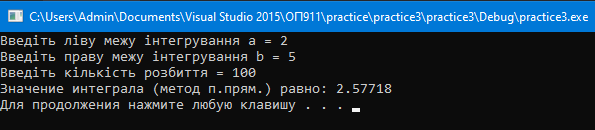
a = 2, b = 5, кількість розбиття = 100, точність = 0,001

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Функція | Метод прямокутників | | | Метод трапецій | Метод Симпсона |
| Лівих | Правих | Середніх |
| 1 |  | 2.59066 | 2.57718 | 2.5895 | 2.60814 | 2.58946 |

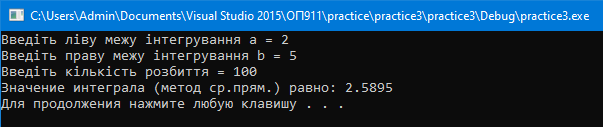
1) лівих прямокутники



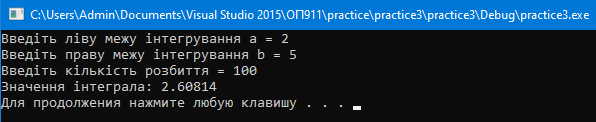
2) праві прямокутники



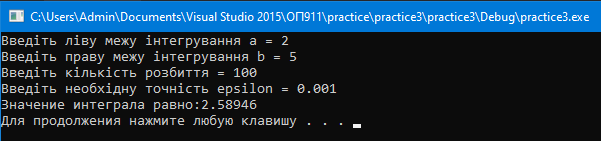
3) середніх прямокутники



4) метод трапецій



5) метод Сімпсона



Висновок: в даному завдані ми розглядали методи інтегрування:

-Метод прямокутників — найпростіший метод [чисельного інтегрування](https://uk.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%B5%D0%BB%D1%8C%D0%BD%D0%B5_%D1%96%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), що полягає у заміні значень [функції](https://uk.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D1%96%D1%8F_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) на проміжку значенням функції в деякій точці проміжку. Цей метод метод має 3 види: метод лівих, правих та середніх прямокутників.

-Метод трапецій - метод чисельного інтегрування функції однієї змінної, що полягає в заміні на кожному елементарному відрізку підінтегральної функції на многочлен першого ступеня, тобто лінійну функцію.

-Метод Сімпсона - суть методу полягає в наближенні підінтегральної функції на відрізку [a, b] інтерполяційним многочленом другого ступеня p2(x), тобто наближення графіка функції на відрізку параболою.

Після того як були зроблені всі методи я з’ясував, що якщо при великому приросту функції метод трапецій рахує набагато точніше.