**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Отчет по лабораторной работе №6

«**Численное интегрирование функции**»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Выполнил:** | |  | **Принял:** | |
| ФИО: | \_Цыпышев Т. А.\_\_\_\_\_ |  | ФИО: | \_**Самохвалов А. Э.**\_\_\_ |
| Группа: | \_ИУ5-11Б\_\_\_\_\_\_\_\_\_\_ |  | Должность: | \_Преподаватель\_\_\_\_ |
| Дата: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | Дата: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Подпись: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | Подпись: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Москва, 2022 г.

**Постановка задачи**

Численное интегрирование функции с заданной точностью методом прямоугольников и трапеций.

Вычислить определённый интеграл в пределах от a до b для четырех функций **f1 = x, f2 = sin( 22x ), f3 = x4 и f4 = arctg(x)**.

Вычисление интеграла оформить в виде функции **IntegrationRectangle** и **IntegrationTrapezoidal**.

Вычисления выполнить для пяти значений точности: **0.01**, **0.001**, **0.0001**, **0.00001** и **0.000001**.

Результаты представить в виде 5 таблиц для каждого метода (по одной таблице для каждого значения точности). В каждой таблице выводить данные для всех четырех функций.

Исследовать быстродействие алгоритмов в зависимости от подынтегральной функции и требуемой точности.

**Разработка алгоритма**

**Входные данные**

* double a – левая граница интегрирования
* double b – правая граница интегрирования
* double n – количество разбиений
* double EPS[5] – пять значений точности вычислений
* char\* name\_function[] – четыре функции (названия)
* double exact\_values[] – формулы для вычисления точного значения интеграла
* TPF functions[4] – четыре функции для вычисления приближённого значения интеграла
* I\_print arr[4] – массив типа структуры I\_print, в который будут заноситься данные

**Функция IntegrationRectangle**

Вычисляет заданный интеграл методом прямоугольников

Входные данные:

* TPF f – нужная функция
* double a – левая граница интегрирования
* double b – правая граница интегрирования
* double eps – погрешность вычисления
* int& n – ссылка на внешнюю переменную с количеством разбиений

Выходные данные:

* double S1 – приближённое значение интеграла

**Функция IntegrationTrapezoidal**

Вычисляет определённый интеграл методом трапеций

Входные данные:

* TPF f – функция
* double a – левая граница интегрирования
* double b – правая граница интегрирования
* double eps – погрешность вычисления
* int& n – ссылка на внешнюю переменную с количеством разбиений

Выходные данные:

* double S1 – приближённое значение интеграла

**Текст программы**

**Main.cpp**

/// Copyright 2022 ttsypyshev <ttsypyshev01@gmail.com>

#include "../include/All\_functios.h"

#include <iostream>

#include <iomanip>

#include <math.h>

using namespace std;

int main() {

double a = -1, b = 3, n = 1;

double EPS[]{0.01, 0.001, 0.0001, 0.00001, 0.000001};

TPF functions[4] = {f1, f2, f3, f4};

double exact\_values[] = {

(b \* b - a \* a) / 2.0,

(cos(a \* 22.0) - cos(b \* 22.0)) / 22.0,

(b \* b \* b \* b \* b - a \* a \* a \* a \* a) / 5.0,

b \* atan(b) - a \* atan(a) - (log(b \* b + 1) - log(a \* a + 1)) / 2.0};

char \*name\_function[] = {

"y = x",

"y = sin(22x)",

"y = x^4",

"y = acrtg(x)"};

cout << "===Rectangle method===\n";

for (int epsilon\_case = 0; epsilon\_case < 5; epsilon\_case++) {

cout << setw(6) << "\nCalculation ccuracy = " << EPS[epsilon\_case] << endl;

struct I\_print arr[4];

for (int line = 0; line < 4; line++) {

n = 1.;

double S = IntegrationRectangle(functions[line], a, b, EPS[epsilon\_case], n);

arr[line].name = name\_function[line];

arr[line].i\_sum = S;

arr[line].i\_toch = exact\_values[line];

arr[line].n = n;

}

PrintTabl(arr, 4);

}

cout << "\n\n\n===Trapezoid method===\n";

for (int epsilon\_case = 0; epsilon\_case < 5; epsilon\_case++) {

cout << setw(6) << "\nCalculation ccuracy = " << EPS[epsilon\_case] << endl;

struct I\_print arr[4];

for (int line = 0; line < 4; line++) {

n = 1.;

double S = IntegrationTrapezoidal(functions[line], a, b, EPS[epsilon\_case], n);

arr[line].name = name\_function[line];

arr[line].i\_sum = S;

arr[line].i\_toch = exact\_values[line];

arr[line].n = n;

}

PrintTabl(arr, 4);

}

}

**Method\_Integration\_Rectangle.cpp**

/// Copyright 2022 ttsypyshev <ttsypyshev01@gmail.com>

#include "../include/All\_functios.h"

#include <math.h>

double IntegrationRectangle(TPF function, double a, double b, double eps, double &n) {

static double S0 = 0, S1 = 0; //global s0 s1 for function IntegrationRectangle

double dx = (b - a) / n;

for (int i = 0; i < n; ++i) {

S1 += function(a + i \* dx);

}

S1 \*= dx;

if ((abs(S1 - S0)) < eps) {

return S1;

} else {

n \*= 2;

S0 = S1;

S1 = 0;

return IntegrationRectangle(function, a, b, eps, n);

}

}

**Method\_Integration\_Trapezoid.cpp**

/// Copyright 2022 ttsypyshev <ttsypyshev01@gmail.com>

#include "../include/All\_functios.h"

#include <math.h>

double IntegrationRectangle(TPF function, double a, double b, double eps, double &n) {

static double S0 = 0, S1 = 0; //global s0 s1 for function IntegrationRectangle

double dx = (b - a) / n;

for (int i = 0; i < n; ++i) {

S1 += function(a + i \* dx);

}

S1 \*= dx;

if ((abs(S1 - S0)) < eps) {

return S1;

} else {

n \*= 2;

S0 = S1;

S1 = 0;

return IntegrationRectangle(function, a, b, eps, n);

}

}

**Other\_Functions.cpp**

/// Copyright 2022 ttsypyshev <ttsypyshev01@gmail.com>

#include "../include/All\_functios.h"

#include <iostream>

#include <iomanip>

#include <cstring>

#include <math.h>

using namespace std;

double f1(double x) {

return x;

}

double f2(double x) {

return sin(22 \* x);

}

double f3(double x) {

return x \* x \* x \* x;

}

double f4(double x) {

return atan(x);

}

void PrintTabl(I\_print i\_prn[], int k) {

const int m = 4;

int wn[m] = {16, 18, 18, 18};

const char \*title[m] = {"Function", "Integral", "IntSum", "N "};

int size[m];

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

size[i] = strlen(title[i]);

cout << char(218) << setfill(char(196));

for (int j = 0; j < m - 1; j++)

cout << setw(wn[j]) << char(194);

cout << setw(wn[m - 1]) << char(191) << endl;

cout << char(179);

for (int j = 0; j < m; j++)

cout << setw((wn[j] - size[j]) / 2) << setfill(' ') << ' ' << title[j]

<< setw((wn[j] - size[j]) / 2) << char(179);

cout << endl;

for (int i = 0; i < k; i++) {

cout << char(195) << fixed;

for (int j = 0; j < m - 1; j++)

cout << setfill(char(196)) << setw(wn[j]) << char(197);

cout << setw(wn[m - 1]) << char(180) << setfill(' ') << endl;

cout << char(179) << setw((wn[0] - strlen(i\_prn[i].name)) / 2) << ' ' << i\_prn[i].name

<< setw(((wn[0] - strlen(i\_prn[i].name)) / 2) + (strlen(i\_prn[i].name) % 2)) << char(179);

cout << setw(wn[1] - 1) << setprecision(10) << i\_prn[i].i\_toch << char(179)

<< setw(wn[2] - 1) << i\_prn[i].i\_sum << setprecision(6) << char(179)

<< setw(wn[3] - 1) << i\_prn[i].n << char(179) << endl;

}

//низ таблицы

cout << char(192) << setfill(char(196));

for (int j = 0; j < m - 1; j++)

cout << setw(wn[j]) << char(193);

cout << setw(wn[m - 1]) << char(217) << setfill(' ') << endl;

}

**All\_functios.h**

/// Copyright 2022 ttsypyshev <ttsypyshev01@gmail.com>

#ifndef INF\_LAB\_06\_ALL\_FUNCTIOS\_H

#define INF\_LAB\_06\_ALL\_FUNCTIOS\_H

typedef double (\*TPF)(double);

struct I\_print {

char \*name;

double i\_sum;

double i\_toch;

double n;

};

//method rectangle integration

double IntegrationRectangle(TPF f, double a, double b, double eps, double &n);

//method trapezoidal integration

double IntegrationTrapezoidal(TPF f, double a, double b, double eps, double &n);

//other functions (print table)

void PrintTabl(struct I\_print i\_prn[], int k);

double f1(double x);

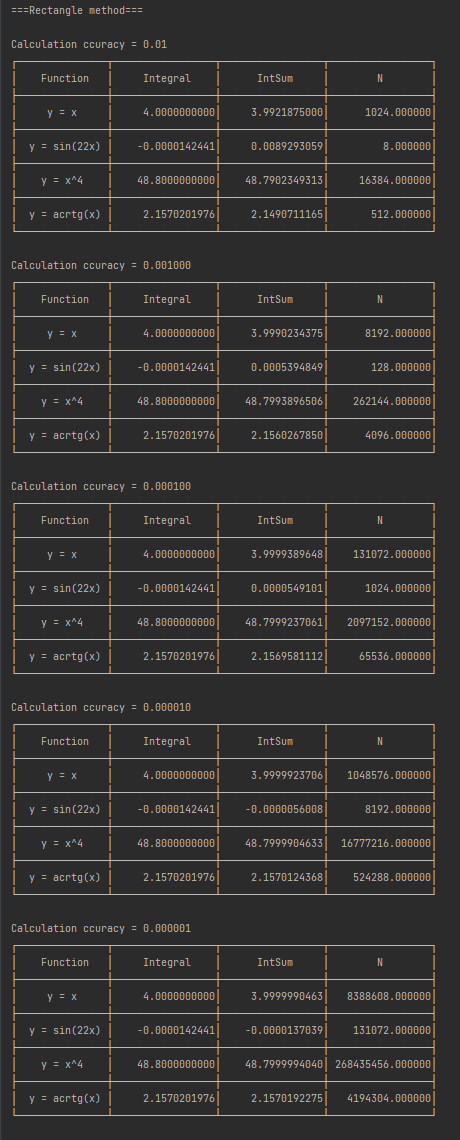
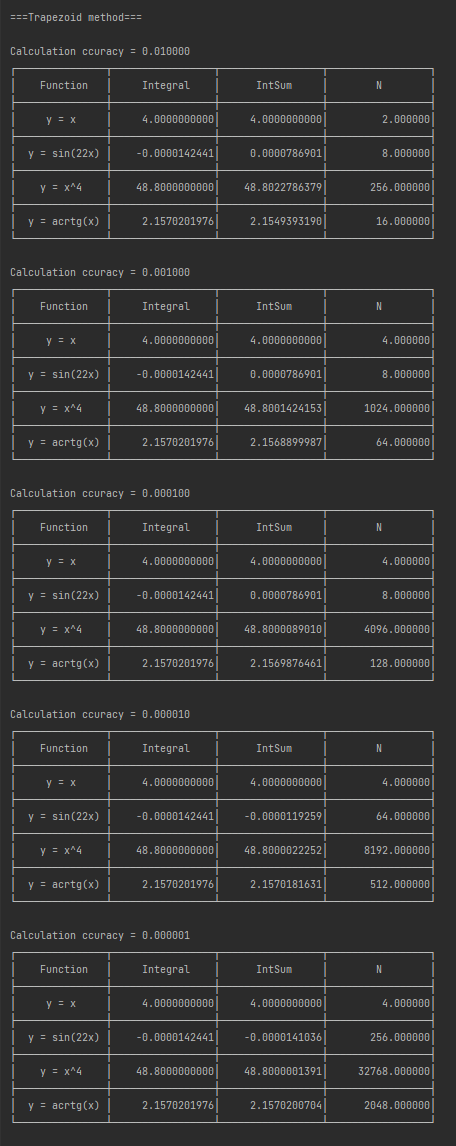
double f2(double x);

double f3(double x);

double f4(double x);

#endif //INF\_LAB\_06\_ALL\_FUNCTIOS\_H

**Анализ результатов**

****

**Вывод**

Метод трапеций быстрее и точнее метода прямоугольников.

Я научился использовать typedef и struct