**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Основы информатики»

Отчет по лабораторной работе №6

# «Сортировка одномерного числового массива»

Вариант 7

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-13 |  | преподаватель каф. ИУ5 |
| Коновалов Максим |  | Папшев И.С |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2020 г.

## Содержание отчета

Отчет по лабораторной работе состоит из 4-х разделов, отражающих основные этапы разработки программы:

- Постановка задачи;

- Разработка алгоритма;

- Кодирование (соответствующий раздел отчета называется «Текст программы»);

- Тестирование (соответствующий раздел отчета называется «Анализ результатов»).

В разделе «Постановка задачи» приведен текст задания.

Раздел «Разработка алгоритма» содержит следующую информацию:

* краткое описание (обоснование) алгоритма,
* описание входных, выходных и вспомогательных данных с указанием их идентификаторов и типов,
* схему алгоритма, состоящую из двух частей: общей (укрупненной) схемы и уточненной схемы одной из функций:
  + double IntRect(TPF f,double a,double b,double eps,int& n);
  + double IntTrap(TPF f,double a,double b,double eps,int& n).

(Номер блока равен остатку от деления порядкового номера студента в списке группы на 2 плюс 1.)

В разделе «Текст программы» приведен листинг программы, включающий необходимые комментарии.

В разделе «Анализ результатов» приведены распечатки экранов для обоих методов численного интегрирования при eps = 0.00001 и eps = 0.000001. Для экономии краски изображения экранов должны иметь белый фон.

Отчет распечатан на принтере на листах бумаги формата А4, скрепленных в левом верхнем углу с помощью степлера, и подписан исполнителем с указанием даты сдачи отчета преподавателю.

## - Постановка задачи

1. Численное интегрирование функции с заданной точностью методом прямоугольников.

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

Вычисление интеграла оформить в виде функции IntRect.

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

Исследовать быстродействие алгоритма в зависимости от подынтегральной функции и требуемой точности (быстродействие алгоритма можно оценить числом элементарных прямоугольников ***n,*** при котором достигнута требуемая точность интегрирования).

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

Для печати таблицы результатов использовать приведенную в приложении 2 функцию

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

Здесь i\_prn[] – массив структур типа I\_print размерностью k.

Для ее использования нужно создать и заполнить массив из структур типа I\_print и передать его в функцию в качестве параметра.

Вид таблицы приведен в Приложении 1.

Для использования в функции PrintTabl( ) символов псевдографики заменитете в ней кодировочную таблицу Windous 1251 на кодировочную таблицу MS Dos 866 (так как это глобальная переменная, то перед выходом из функции нужно восстановить кодировочную таблицу Windous 1251).

2. Выполнить п.1, используя для интегрирования метод трапеций. Вычисление интеграла оформить в виде функции IntTrap.

Для печати таблиц результатов использовать ту же функцию, что и в методе прямоугольников.

## - Разработка алгоритма

* краткое описание (обоснование) алгоритма,

Для удобства и чтобы не потеряться в множестве функций будем пользоваться ресурсными файлами cpp, а также заголовочными. Их содержание представлено в блоке 3.

Создадим некоторую структуру, которая будет соответствовать 1 строчке в нашей таблице. Таблицу представим в виде массива, состоящего из 4 структур.

I\_print table[4] =

{

{"y=x",3.0,3.0,3},

{"y=sin(22x)",3.0,3.0,3},

{"y=x^4",3.0,3.0,3},

{"y=arctg(x)",3.0,3.0,3}

};

В main создадим некоторый цикл for по i на отрезке от 0 до 4 в соответствии с требуемой точностью . (Всего их 5 0.1,0.001 … и тд) Тогда EPS будет соответствовать pow(0.1, i+1) .

Определим массивы указателей на интегрируемую функцию и функцию интеграла от а до б , вычисленного вручную.

До этого спросим у пользователя какую ссылку на требуемый метод интегрирования использовать и выберем ее путем switch

И так arg – ссылка на функцию интегрирования(метод трапеций/метод прямоугольников)

Дальше будем проходиться по строкам каждой таблицы и присваивать им необходимое значение

В конце цикла i-ой таблицы(после заполнения i-ой таблицы) выводим ее функцией PrintTab.

for (int i=0;i<5;i++) //Проходимся по таблицам

{

cout << "Eps for this table is " <<fixed<< pow(0.1, i + 1)<<endl;

for (int j = 0; j < 4; j++) // Проходимся по строкам 1-ой таблицы

{

table[j].i\_sum = alg(funcs[j], a, b, pow(0.1,i+1), table[j].n);

table[j].i\_toch = realIntegr[j](a, b);

}

PrintTabl(table, 4);

}

Нахождение нужного количества делений\* функции реализуем путем цикла с условием

Do{}while(aбсолютное значение(интеграл с предыдущим n - интеграл со след. n) >eps)

* описание входных, выходных и вспомогательных данных с указанием их идентификаторов и типов

//Вводим тип алгоритма

int typeOfAlg = 0;

typedef double (\*Alg)(TPF f, double a, double b, double eps, int& n); // определяем ссылку на функцию интегрирования

Alg alg = IntRect; //Соответсвенно инициализируем и объявляем ссылку, чтобы не поймать ошибки

double (\*realIntegr[4])(double a, double b) = {realF1,realF2,realF3,realF4};

double (\*funcs[4])(double x) = { f1,f2,f3,f4 }; // Массивы указателей. Real – вычесленный вручную интеграл, f – ссылка на функцию

I\_print table[4]; // Массив структур описанный выше

struct I\_print {

public: //данные для печати результатов интегрирования

char name[12]; //название функции

double i\_sum; //значение интегральной суммы

double i\_toch; //точное значение интеграла

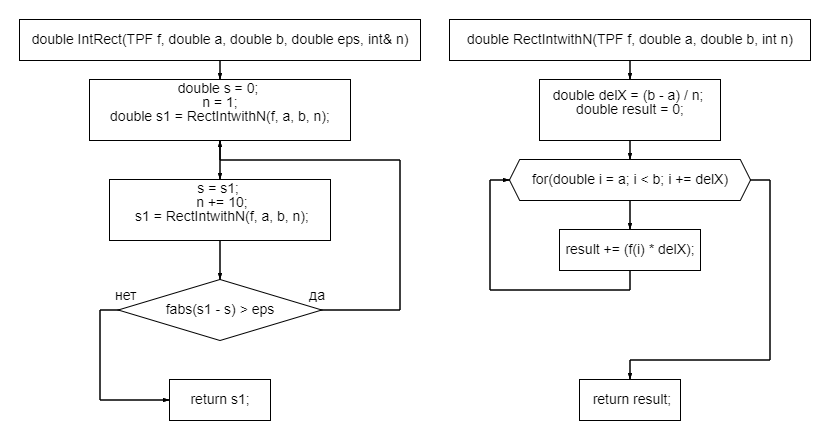
int n; //число разбиений области интегрирования при котором достигнута требуемая точность

};

переменные s и s1 созданы для текущего и следующего интеграла соответственно.

int delX; // Дельта X

* схема алгоритма, состоящая из уточненной схемы одной из функций:
  + double IntRect(TPF f,double a,double b,double eps,int& n);



* + ~~double IntTrap(TPF f,double a,double b,double eps,int& n).~~

(Номер блока равен остатку от деления порядкового номера студента в списке группы на 2 плюс 1.) 1 соответственно

## - Текст программы

***Исходные файлы***

Lab 6\_1.cpp

#include <iostream>

#include <iomanip>

#include <math.h>

#include "MainFunctions.h"

#include "IntegrateFunctions.h"

#include "TablManip.h"

using namespace std;

//Инициализация структуры вне main файла не проходит. Почему?

int main() {

//Вводим тип алгоритма

cout << "1 - RiemannSum \n 2- TrapezoidalRule\n";

cout << "Choose your algorithm >> ";

int typeOfAlg = 0;

cin >> typeOfAlg;

typedef double (\*Alg)(TPF f, double a, double b, double eps, int& n);

Alg alg = IntRect;

switch(typeOfAlg) {

case(1):

alg = IntRect;

break;

case(2):

alg = IntTrap;

break;

}

//Вводим начало и конец интегрирования

cout << "Put your a and b >> ";

double a = -1.0;

double b = 3.0;

cin >> a >> b;

I\_print table[4] =

{

{"y=x",3.0,3.0,3},

{"y=sin(22x)",3.0,3.0,3},

{"y=x^4",3.0,3.0,3},

{"y=arctg(x)",3.0,3.0,3}

};

double (\*realIntegr[4])(double a, double b) = {realF1,realF2,realF3,realF4};

double (\*funcs[4])(double x) = { f1,f2,f3,f4 };

for (int i=0;i<5;i++) //Проходимся по таблицам

{

cout << "Eps for this table is " << pow(0.1, i + 1)<<endl;

for (int j = 0; j < 4; j++) // Проходимся по строкам 1-ой таблицы

{

table[j].i\_sum = alg(funcs[j], a, b, pow(0.1,i+1), table[j].n);

table[j].i\_toch = realIntegr[j](a, b);

}

PrintTabl(table, 4);

}

}

IntegrateFunctions.cpp

#include <math.h>

double abs(double a) {

if (a > 0) return a;

return -a;

}

typedef double(\*TPF)(double);

double RectIntwithN(TPF f, double a, double b, int n);

double TrapIntwithN(TPF f, double a, double b, int n);

double IntTrap(TPF f, double a, double b, double eps, int& n) {

double s = 0;

n = 1;

double s1 = TrapIntwithN(f, a, b, n);

do {

s = s1;

n +=10;

s1 = TrapIntwithN(f, a, b, n);

} while (fabs(s1 - s) > eps);

return s1;

}

double TrapIntwithN(TPF f, double a, double b, int n)

{

double result = 0;

double left = f(a);

double delX = (b - a) / n;

for (double i = a + delX; i <= b; i += delX) //((b - a) / n) - delta x

{

result += 0.5 \* (left + f(i))\*delX;

left = f(i);

}

return result;

}

double IntRect(TPF f, double a, double b, double eps, int& n) {

double s = 0;

n = 1;

double s1 = RectIntwithN(f, a, b, n);

do {

s = s1;

n += 10;

s1 = RectIntwithN(f, a, b, n);

} while (fabs(s1 - s) > eps);

return s1;

}

double RectIntwithN(TPF f, double a, double b, int n) {

double delX = (b - a) / n;

double result = 0;

for (double i = a; i < b; i += delX) {

result += (f(i) \* delX);

}

return result;

}

double realF1(double a, double b)

{

return (b \* b - a \* a) / 2.0;

}

double realF2(double a, double b)

{

return (cos(a \* 22.0) - cos(b \* 22.0)) / 22.0;

}

double realF3(double a, double b)

{

return (b \* b \* b \* b \* b - a \* a \* a \* a \* a) / 5.0;;

}

double realF4(double a, double b)

{

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

}

MainFunctions.cpp

#include <math.h>

double f1(double x) {

return x;

}

double f2(double x) {

return sin(22 \* x);

}

double f3(double x) {

return pow(x, 4);

}

double f4(double x) {

return atan(x);

}

TablManip.cpp

#include <iostream>

#include <iomanip>

using namespace std;

struct I\_print {

public: //данные для печати результатов интегрирования

char name[12]; //название функции

double i\_sum; //значение интегральной суммы

double i\_toch; //точное значение интеграла

int n; //число разбиений области интегрирования при котором достигнута требуемая точность

};

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

{

system("chcp 866>nul");

const int m = 4;//число столбцов таблицы

int wn[m] = { 12,18,18,10 };//ширина столбцов таблицы

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

int size[m];

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

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

//шапка таблицы

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++)

{//заполнение таблицы

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) << 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 << 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) << endl;

//восстановление первоначальных значений

cout << setprecision(6) << setfill(' ');

system("chcp 1251>nul");

}

***Файлы заголовков***

IntegrateFunctions.h

#pragma once

typedef double (\*TPF)(double);

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

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

double realF1(double a, double b);

double realF2(double a, double b);

double realF3(double a, double b);

double realF4(double a, double b);

MainFunctions.h

#pragma once

double f1(double x);

double f2(double x);

double f3(double x);

double f4(double x);

TablManip.h

#pragma once

struct I\_print {

public: //данные для печати результатов интегрирования

char name[12]; //название функции

double i\_sum; //значение интегральной суммы

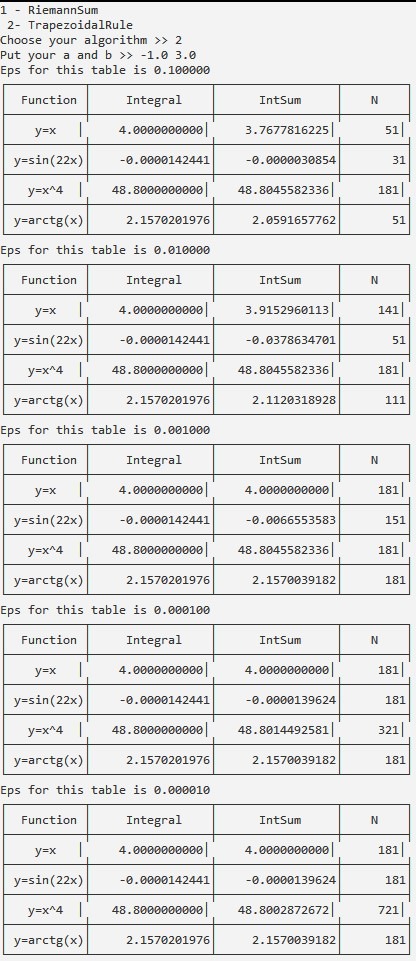
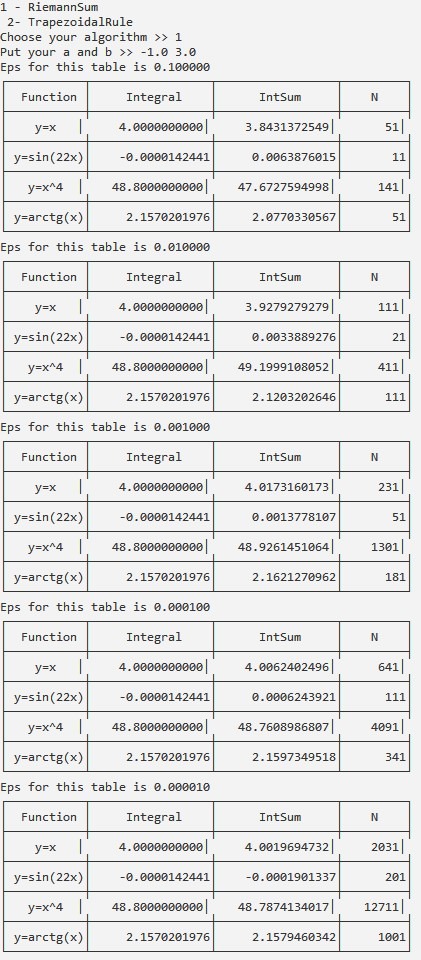
double i\_toch; //точное значение интеграла

int n; //число разбиений области интегрирования при котором достигнута требуемая точность

};

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

## - Анализ результатов



Метод прямоугольников(Римана) Метод трапеций