**Министерство образования и науки Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего профессионального образования  
«Московский государственный технический университет имени Н.Э. Баумана»  
(МГТУ им. Н.Э. Баумана)**

|  |  |
| --- | --- |
| **ФАКУЛЬТЕТ** | **«Информатики и систем управления»** |
| **КАФЕДРА** | **5-13** |

**ОТЧЕТ**

***ПО ЛАБОРАТОРНОЙ РАБОТЕ № 8***

ПО ДИСЦИПЛИНЕ:«Основы программирования»

Студент Гаврилов Л.Я.

(Группа) (Подпись, дата) (И.О. Фамилия)

Проверил Козлов А.Д.

(Подпись, дата) (И.О. Фамилия)

*2018  г.*

# Задания на лабораторную работу

## Задание 1

Численное интегрирование функций

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 уже разработанную функцию

void PrintTabl(I\_print i\_prn[],int k), приведенную в приложении 2.

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).

### Метод прямоугольников.

При численном интегрировании площадь под кривой заменяется суммой площадей «элементарных» прямоугольников с основанием ∆x и высотой, проведенной из середины основания.

Формула приближенного значения определенного интеграла представляется в виде

,

где: *xi = a + Δx/2 + (i-1)\*Δx; N -* число элементарных прямоугольников*.*

Для уменьшения объема вычислений множитель Δx следует вынести за знак суммы. Тогда в цикле нужно выполнять только суммирование, а затем полученную сумму один раз умножить на *Δx*.

Для оценки погрешности вычисления интеграла на практике используют правило Рунге. Суть правила состоит в том, что выполняют вычисление интеграла с двумя разными шагами изменения переменной *х*, а затем сравнивают результаты и получают оценку точности. Наиболее часто используемое правило связано с уменьшением *Δx* в два раза на каждом шаге вычисления интегральной суммы.

Для методов прямоугольников и трапеций погрешность R*Δx/2* вычисления интеграла с шагом *Δx/2* оценивается следующей формулой:

|R*Δx/2*| =, (1)

где I*Δx/2* – значение интеграла, вычисленное с шагом *Δx/2*; I*Δx* – значение интеграла, вычисленное с шагом *Δx.*

В программе вычисления интеграла с точностью ***eps***во внутреннем цикле находят значение определенного интеграла с шагом *Δx/2*.

Во внешнем цикле производится сравнение значений интегральных сумм, вычисленных с шагами *Δx* (вычислена на предыдущем шаге) и *Δx/2* соответственно. Если требуемая точность не достигнута, то число разбиений удваивается, а в качестве предыдущего значения интеграла берут текущее и вычисление интеграла выполняется при новом числе разбиений.

Для организации итерационного цикла перед входом в цикл нужно определить начальные значения *Δx* и I*Δx*.

Вычисление интеграла оформить в виде функции IntRect, формальными параметрами которой являются:

*f* - имя интегрируемой функции,

*a, b* – границы интервала интегрирования,

*eps* – требуемая точность,

*n* – число прямоугольников, при котором достигнута требуемая точность (выходной).

Функция возвращает значение интеграла.

Прототип функции:

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

Здесь:

TPF – имя типа указателя на подынтегральную функцию, определяемый пользователем:

typedef double (\*TPF)(double);

После выполнения этого оператора в программе вместо вместо типа

double (\*)(double) - указатель на функцию, которая возвращает double и имеет один параметр типа double - можно писать просто TPF, то есть TPF становится синонимом double (\*)(double).

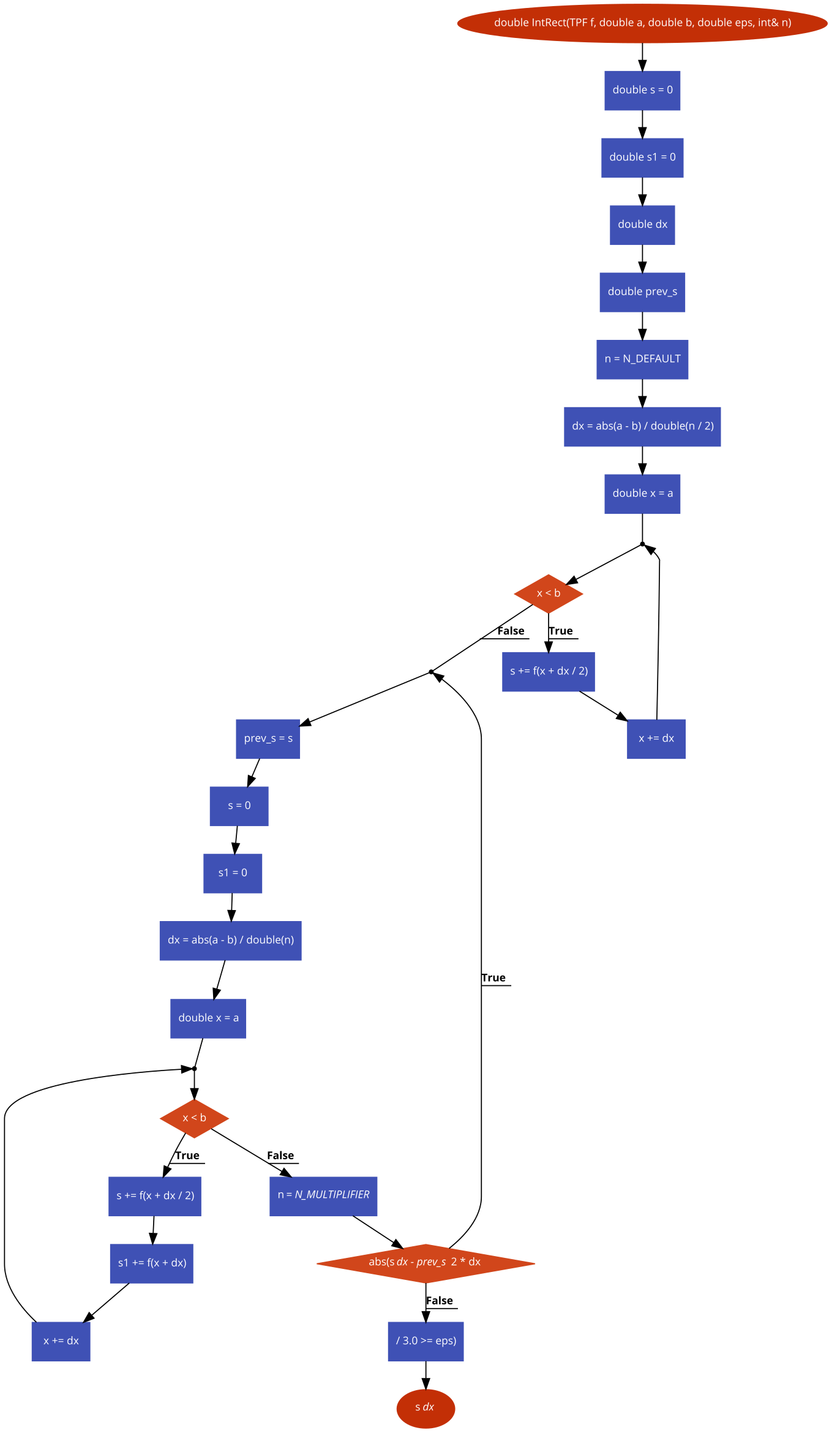
Для хранения и печати результатов вычислений используйте структуру, элементами которой являются наименование функции, значения интеграла (точное и вычисленное в виде суммы) и число «элементарных» прямоугольников ***n***, при котором достигнута требуемая точность. Точные значения, полученные аналитически, нужны для оценки правильности результатов численного интегрирования.

# Описание разработанных программ

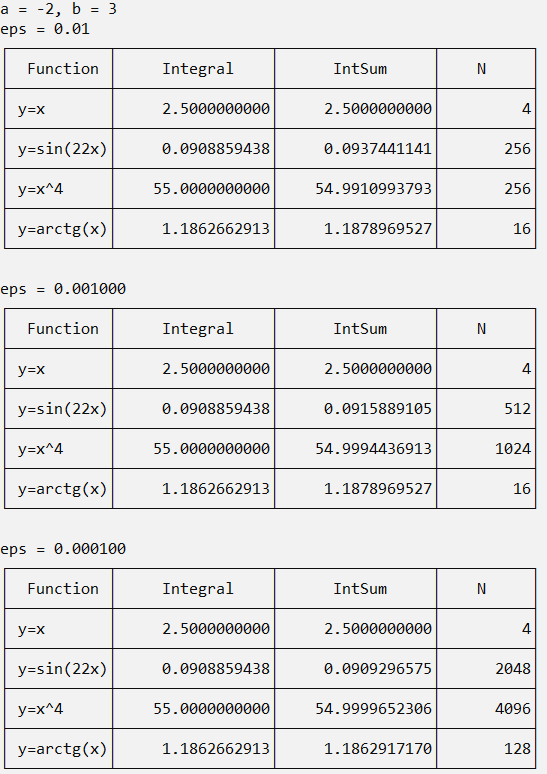
## Задание 1

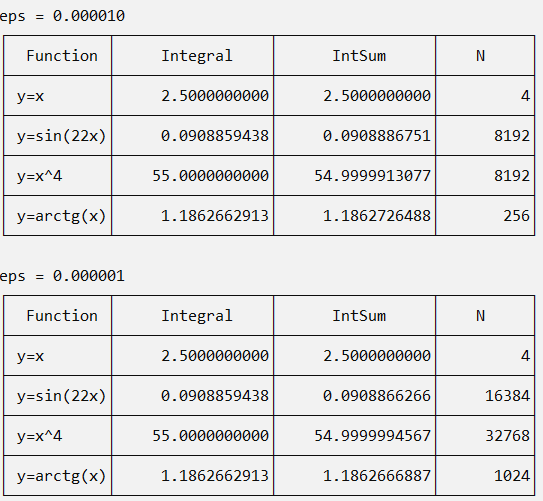
### Описание используемых алгоритмов

В программе описан основной метод – «Прямоугольников». Слишком большая программа для схемы алгоритма.



### Пример выполнения программы





### Текст программы

#include <iostream>

#include <cmath>

#include <iomanip>

#define MAX\_EPS 5 //Приближение

#define N\_DEFAULT 2

#define N\_MULTIPLIFIER 2

#define A -2

#define B 3

//Режим

#define MODE 0

typedef double(\*TPF)(double);

typedef double(\*EXF)(double, double);

using namespace std;

struct I\_print { //данные для печати результатов интегрирования

const char\* name;//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

};

double f1(double); //x

double f2(double); //sin(22x)

double f3(double); //x^4

double f4(double); //arctg(x)

double ex1(double a, double b); //Точный интеграл x

double ex2(double a, double b); //Точный интеграл sin(22x)

double ex3(double a, double b); //Точный интеграл x^4

double ex4(double a, double b); //Точный интеграл arctg(x)

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

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

int main()

{

system("color f0");

const TPF f[] = { f1, f2, f3, f4 };

const EXF ex[] = { ex1, ex2, ex3, ex4 };

int size = 4;

I\_print print\_data[4];

print\_data[0].name = "y=x ";

print\_data[1].name = "y=sin(22x)";

print\_data[2].name = "y=x^4 ";

print\_data[3].name = "y=arctg(x)";

const double eps[] = { 0.01, 0.001, 0.0001, 0.00001, 0.000001 };

double a = A;

double b = B;

int n;

double dx;

cout << setprecision(10);

cout << "a = " << a << ", b = " << b << endl;

for (int j = 0; j < MAX\_EPS; j++)

{

cout << "eps = " << eps[j] << endl;

for (int i = 0; i < 4; i++)

{

if (MODE == 0) print\_data[i].i\_sum = IntRect(f[i], a, b, eps[j], n);

print\_data[i].i\_toch = ex[i](a, b);

print\_data[i].n = n;

}

PrintTabl(print\_data, size);

cout << "\n";

}

return 0;

}

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

{

double s = 0;

double s1 = 0;

double dx;

double prev\_s;

n = N\_DEFAULT;

dx = abs(a - b) / double(n / 2);

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

{

s += f(x + dx / 2);

}

do

{

prev\_s = s;

s = 0;

s1 = 0;

dx = abs(a - b) / double(n);

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

{

s += f(x + dx / 2);

s1 += f(x + dx);

}

n \*= N\_MULTIPLIFIER;

} while (abs(s \* dx - prev\_s \* 2 \* dx) / 3.0 >= eps);

return s \* dx;

}

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

{

const int m = 4;//число столбцов таблицы

int wn[m] = { 12,18,18,11 };//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 << 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;

}

double ex1(double a, double b)

{

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

}

double ex2(double a, double b)

{

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

}

double ex3(double a, double b)

{

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

}

double ex4(double a, double b)

{

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

}

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);

}

# Вводы

* Определение и использование функций.
* Обмен данными с функциями через возвращаемые значения, параметры и глобальные переменные. Передача параметров по ссылке и по значению.
* Перегрузка функций.
* Указатели на функции и их использование (объявление, передача указателей на функцию в качестве параметров и вызов функций с помощью указателя в теле функции) для динамического выбора функций во время выполнения программы.
* Передача одномерных массивов в функцию;
* Объединение разнородных данных в структуру;
* Использование массивов из элементов типа структура.