**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

«Численное интегрирование функций»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-15Б |  | преподаватель каф. ИУ5 |
| Коновалов Илья |  | Папшев И.С. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2021 г.

**Задача:**

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

**Алгоритм:**

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 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.

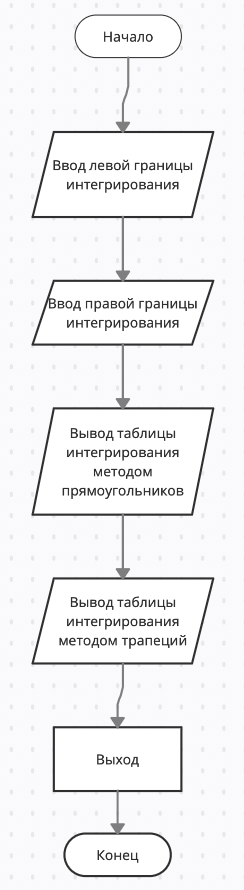
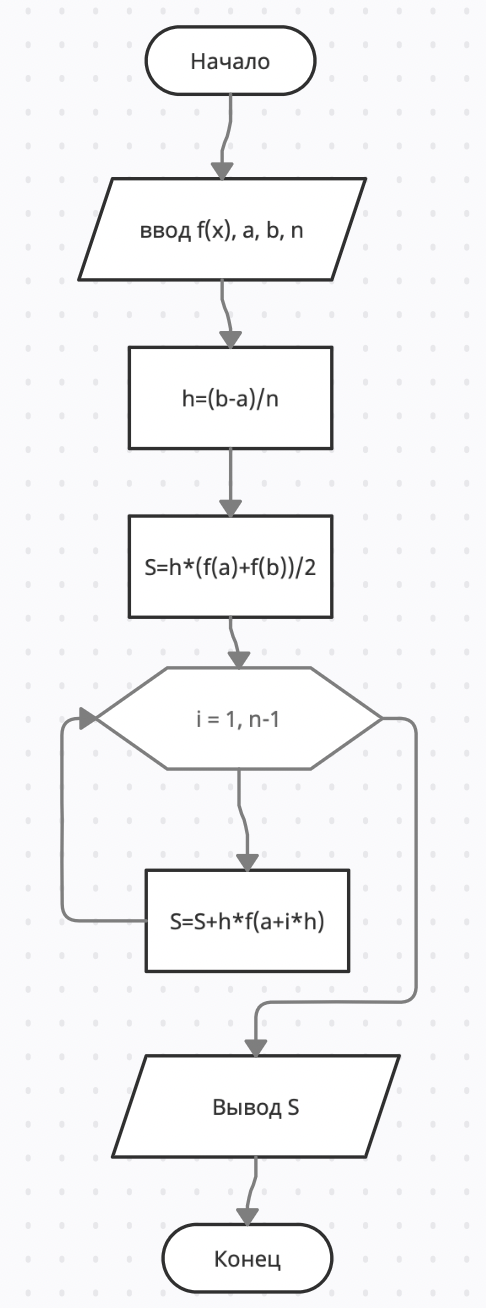
Для методов прямоугольников и трапеций погрешность R*Δx/2* вычисления интеграла с шагом *Δx/2* оценивается следующей формулой:

|R*Δx/2*| =, (1)

где I*Δx/2* – значение интеграла, вычисленное с шагом *Δx/2*; I*Δx* – значение интеграла, вычисленное с шагом *Δx.*Деление *Δx/2* продолжается до тех пор, пока погрешность |R*Δx/2*| не станет меньше ***eps.***

Блок-схема общей Блок-схема метода трапеций

программы

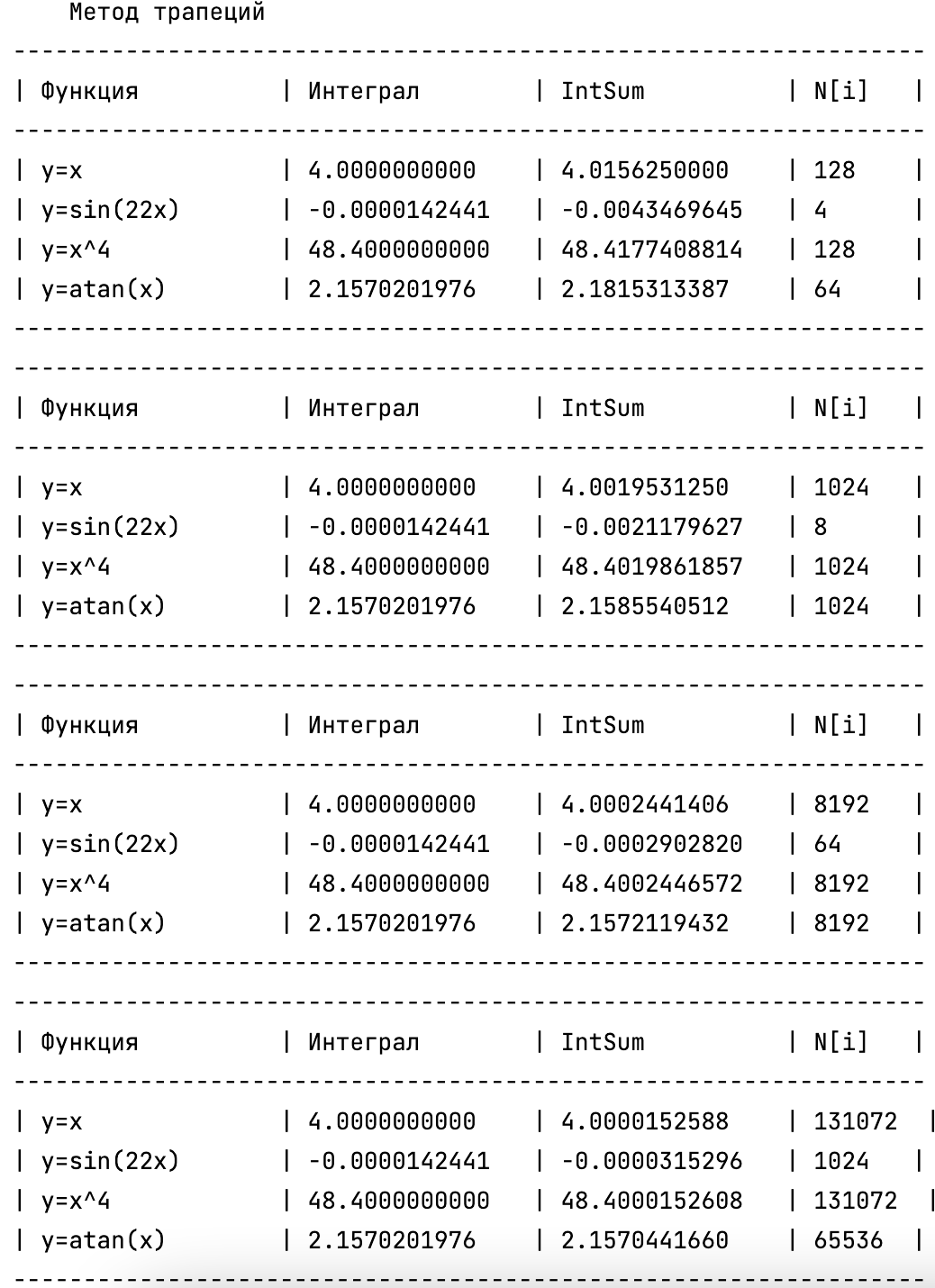
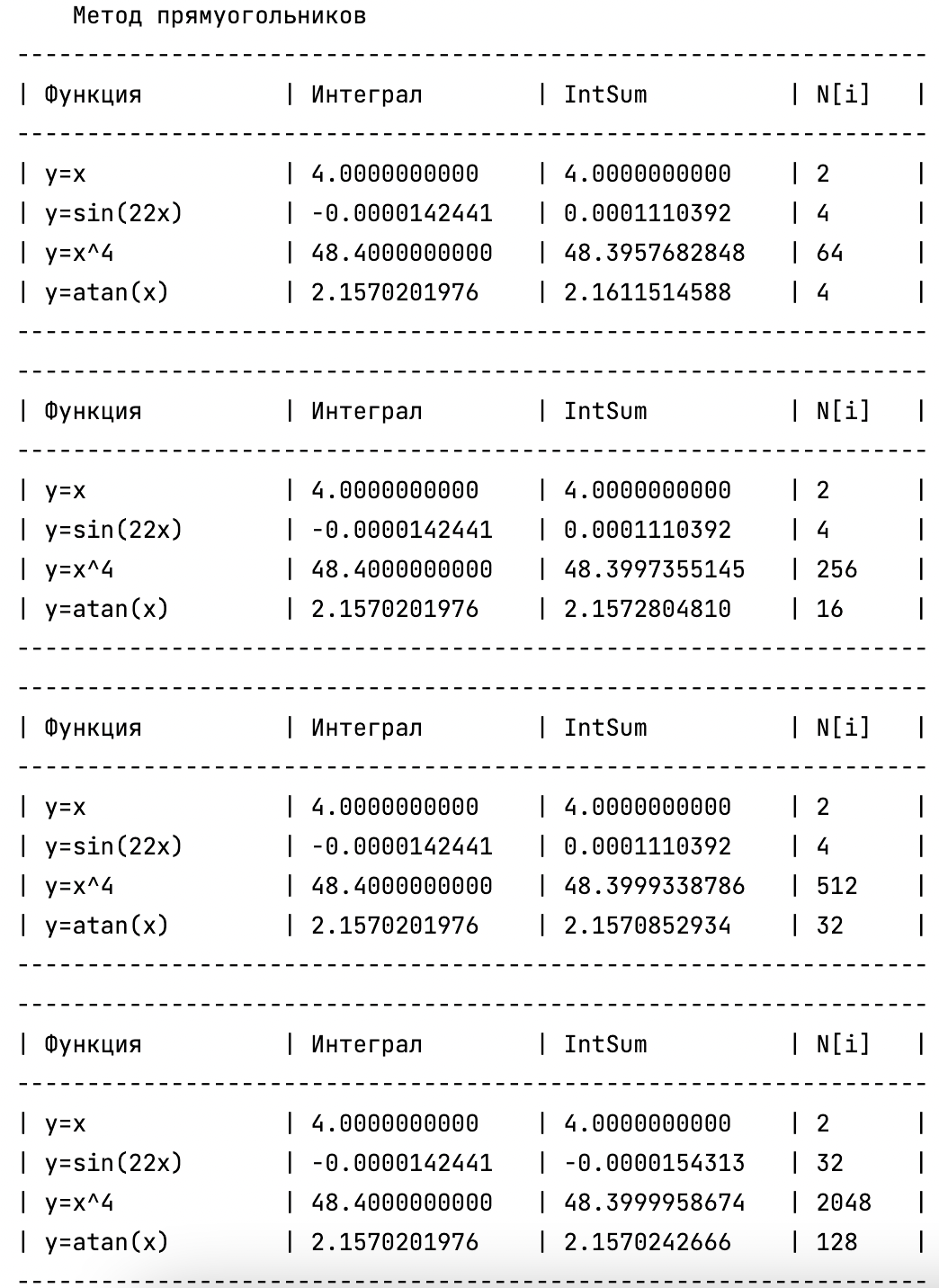
**Код программы:**

#include **<iostream>**#include **<cmath>**#include**<iomanip>  
  
using namespace** std;  
**typedef double**(\*TPF)(**double**);  
**double** f1(**double** x) {  
 **return** x;  
}  
**double** f2(**double** x) {  
 **return** sin(22 \* x);  
}  
**double** f3(**double** x) {  
 **double** d = x \* x;  
 **return** d \* d;  
}  
**double** f4(**double** x) {  
 **return** atan(x);  
}  
  
**double** IntTrap(TPF f, **double** a, **double** b, **int** n) {  
 **double** x, h;  
 **double** sum = 0;  
 **double** S;  
 h = (b - a) / n;  
  
 **for** (**int** i = 0; i < n; i++) {  
 x = a + i \* h;  
 S = f(x);  
 sum += S;  
 }  
 sum += (f(a)+f(b))/2;  
 **return** (sum \* h);  
}  
  
  
**double** IntRect(TPF f, **double** a, **double** b, **int** n) {  
 **double** sum = 0.0, step;  
  
 **if** (0 == n) **return** sum;  
  
 step = (b - a) / (n);  
 **for** (**int** i = 0 ; i < n ; i++ ) {  
 sum += f (a + step\*(i+0.5));  
 }  
 sum \*= step;  
 **return** sum;  
}  
  
**struct** print {  
 **const char**\* name;  
 **double** i\_sum, i\_toch;  
 **int** n;  
};  
  
 *// вывод результатов на экран* **void** PrintRes(print i\_prn[], **int** k) {  
 cout << **"-----------------------------------------------------------------"** << endl;  
 cout << **"| Функция | Интеграл | IntSum | N[i] |"** << endl;  
 cout << **"-----------------------------------------------------------------"** << endl;  
 **for** (**int** i = 0; i < k; i++) {  
 cout << setiosflags(ios::left);  
 cout << **"| "** << setw(16) << i\_prn[i].name << **" | "** << setw(15) << fixed << setprecision(10) << i\_prn[i].i\_toch << **" | "** << setw(15) << i\_prn[i].i\_sum << **" | "** << setw(5) << i\_prn[i].n << **" |"** << endl;  
 }  
 cout << **"-----------------------------------------------------------------"** << endl;  
 }  
  
  
**int** main() {  
 **double** a, b;  
 **double** S1, S;  
 print arr[4];  
 **int** n = 1, c = 0;  
 **double** toch[4] = {0.01, 0.001, 0.0001, 0.00001};  
 **double** (\*fun[4])(**double**) = {f1, f2, f3, f4};  
 **const char** \*name[] = {**"y=x "**, **"y=sin(22x)"**, **"y=x^4 "**, **"y=atan(x) "**};  
 cout << **"Левая граница интегрирования a = "**;  
 cin >> a;  
 cout << **"Правая граница интегрирования b = "**;  
 cin >> b;  
 **double** d = b \* b, g = d \* d, e = a \* a, u = e \* e;  
 **double** O[4] = {(d - e) / 2, (cos(22 \* a) - cos(22 \* b)) / 22, (g \* b - u \* a) / 5,  
 b \* atan(b) - a \* atan(a) - (log(d + 1) - log(e + 1)) / 2};  
 cout << **"\tМетод прямуогольников\n"**;  
 **for** (**int** i = 0; i < 4; i++) {  
 **for** (**int** j = 0; j < 4; j++) {  
 **int** n = 1;  
 arr[j].name = name[j];  
 arr[j].n = toch[i];  
 S1 = IntRect(\*fun[j], a, b, n);  
 **int** k = 0;  
 **do** {  
 S = S1;  
 n = 2 \* n;  
 S1 = IntRect(\*fun[j], a, b, n);  
 k++;  
 } **while** (fabs((S1 - S)/3) > toch[i]);  
 arr[j].i\_sum = S1;  
 arr[j].i\_toch = O[j];  
 arr[j].n = n;  
 }  
 PrintRes(arr, 4);  
 }  
 cout << **"\tМетод трапеций\n"**;  
 **for** (**int** i = 0; i < 4; i++) {  
 **for** (**int** j = 0; j < 4; j++) {  
 **int** n = 1;  
 arr[j].name = name[j];  
 arr[j].n = toch[i];  
 S1 = IntTrap(\*fun[j], a, b, n);  
 **int** k = 0;  
 **do** {  
 S = S1;  
 n = 2 \* n;  
 S1 = IntTrap(\*fun[j], a, b, n);  
 *// k++;* } **while** (fabs((S1 - S)/3) > toch[i]);  
 arr[j].i\_sum = S1;  
 arr[j].i\_toch = O[j];  
 arr[j].n = n;  
 }  
 PrintRes(arr, 4);  
 }  
}

**Результаты:**

Левая граница интегрирования a = 1

Правая граница интегрирования b = 3



**Вывод:**

В ходе выполнения лабораторной работы я приобрел навыки разработки программы для численного интегрирования функции с заданной точностью методом прямоугольников и методом трапеций и освоил следующие приемы программирования:

- передача в функцию параметров «по значению» и «по адресу»;

- передача в функцию другой функции в виде параметра;

- передача одномерных массивов в функцию;

- объединение разнородных данных в структуру;

- использование массивов из элементов типа структура;

- повышение эффективности использования вложенных циклов.