Московский государственный технический
университет им. Н. Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управление»

Курс «Основы программирования»

Отчет по лабораторной работе №5
«Численное интегрирование функций»

Выполнил:	Проверил:
Студент группы ИУ5-11Б	Преподаватель каф. ИУ5
Алехин Сергей	Правдина Анна Дмитриевна
Подпись и дата:	Подпись и дата:

Москва, 2018 г.

Задание

На примере разработки программы для численного интегрирования функции с заданной точностью методом прямоугольников и методом трапеций освоить 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;

Планируемое время выполнения работы - 8 часов.

Разработка алгоритма

Входные переменные:

1) int n – количество прямоугольников/трапеций;
2) double a – левая граница интегрирования;
3) double b – правая граница интегрирования;
4) TPF f[4] – массив из указателей на функции;
5) I_print i_prn[4] – массив структур на каждую из 4 функций;
6) int i – переменная цикла функций(4 функции);
7) int j – переменная цикла выбора методов(прямоугольники/трапеции);
8) double eps – переменная цикла отвечающая за точность.

Функции:

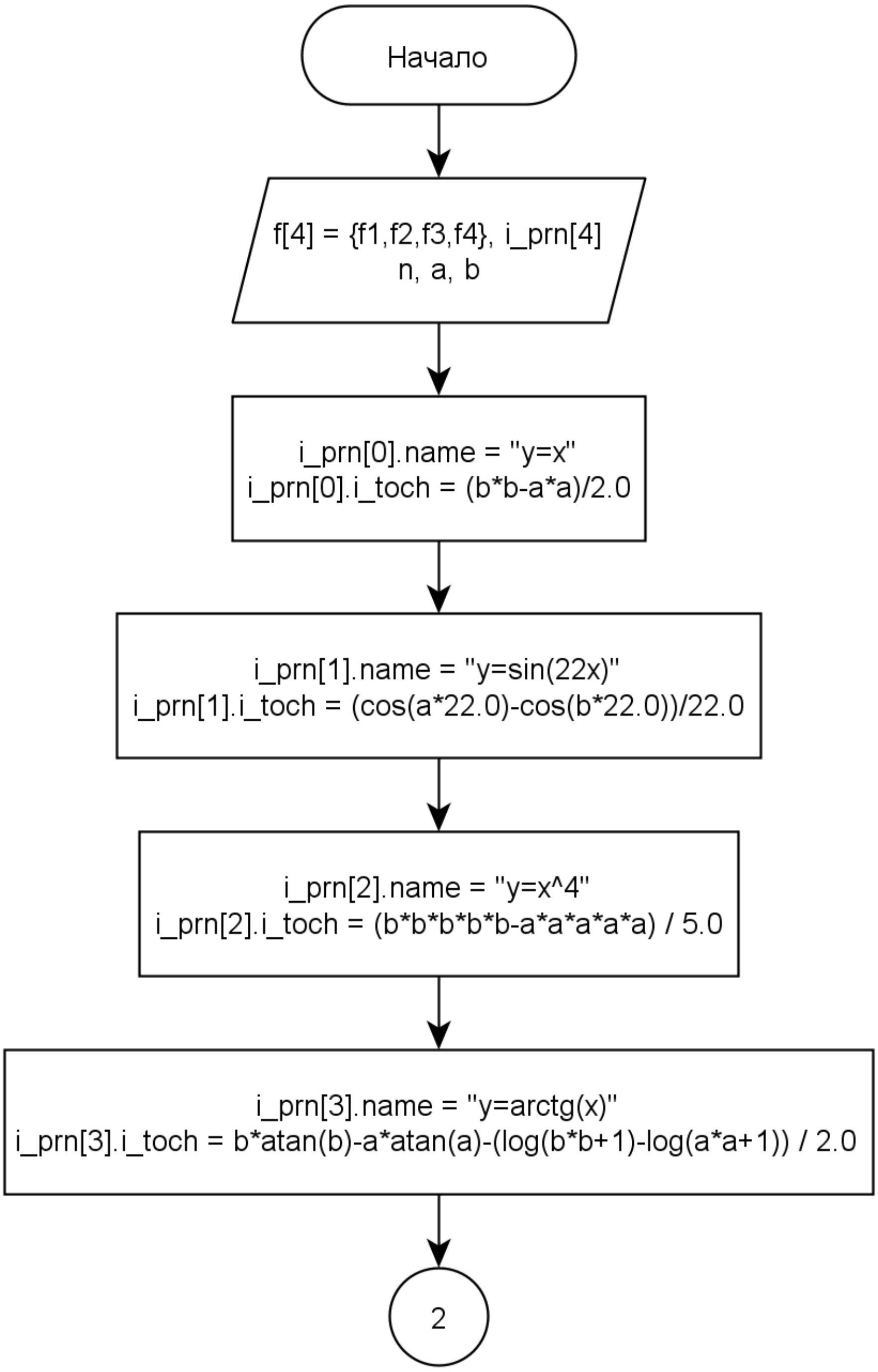
1) void PrintTabl – вывод структурного массива на экран;
   a. *Входные переменные:*
      i. I_print i_prn[] – массив структурных переменных;
      ii. int k – количество переменных в структурном массиве;
2) double f1 – функция y=x;
   a. *Входные переменные:*
      i. Возвращенное значение: x;
3) double f2 – функция y=sin(22x);
   a. *Входные переменные:*
      i. Возвращенное значение: sin(22 * x);
4) double f3 – функция y=x^4;
   a. *Входные переменные:*
      i. Возвращенное значение: x * x * x * x;
5) double f4 – функция y=arctg(x);
   a. *Входные переменные:*
      i. Возвращенное значение: atan(x);
6) double IntRect – функция подсчета методом прямоугольников;
   a. *Входные переменные:*
      i. TPF f – указатель на функцию;
      ii. double a – левая граница интегрирования;
      iii. double b – правая граница интегрирования;
      iv. double eps – точность измерений;
      v. int &n – количество прямоугольников;
   b. *Локальные переменные:*
      i. double dx – ширина прямоугольника;
      ii. double s0 – предыдущее значение интеграла;
      iii. double s1 – текущее значение интеграла;
   c. Возвращенное значение:
      i. Возращенное значение: s1;

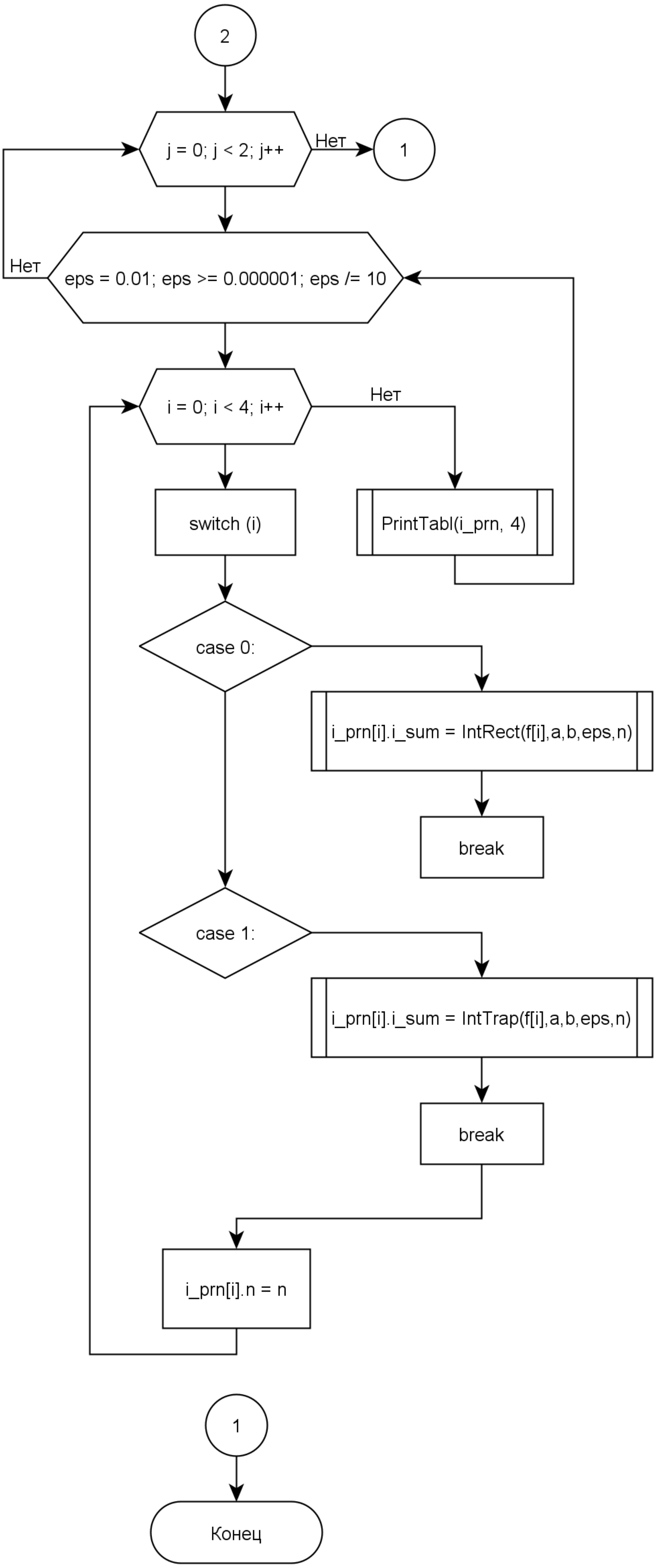
7) double IntTrap – функция подсчета методом трапеций;

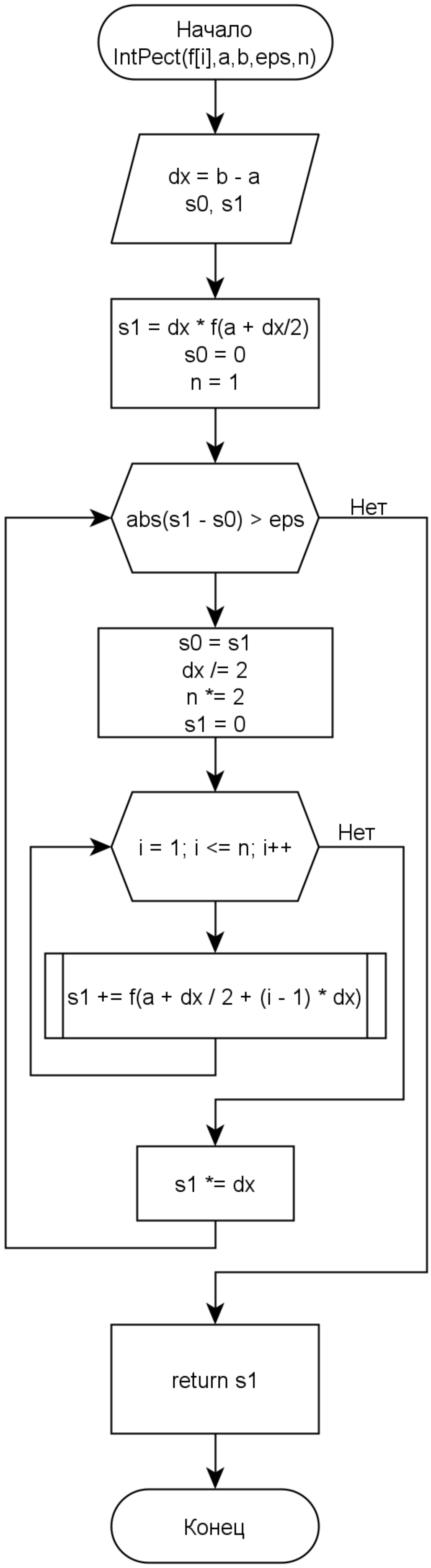
   a. *Входные переменные:*
      - i. TPF f – указатель на функцию;
      - ii. double a – левая граница интегрирования;
      - iii. double b – правая граница интегрирования;
      - iv. double eps – точность измерений;
      - v. int &n – количество трапеций;

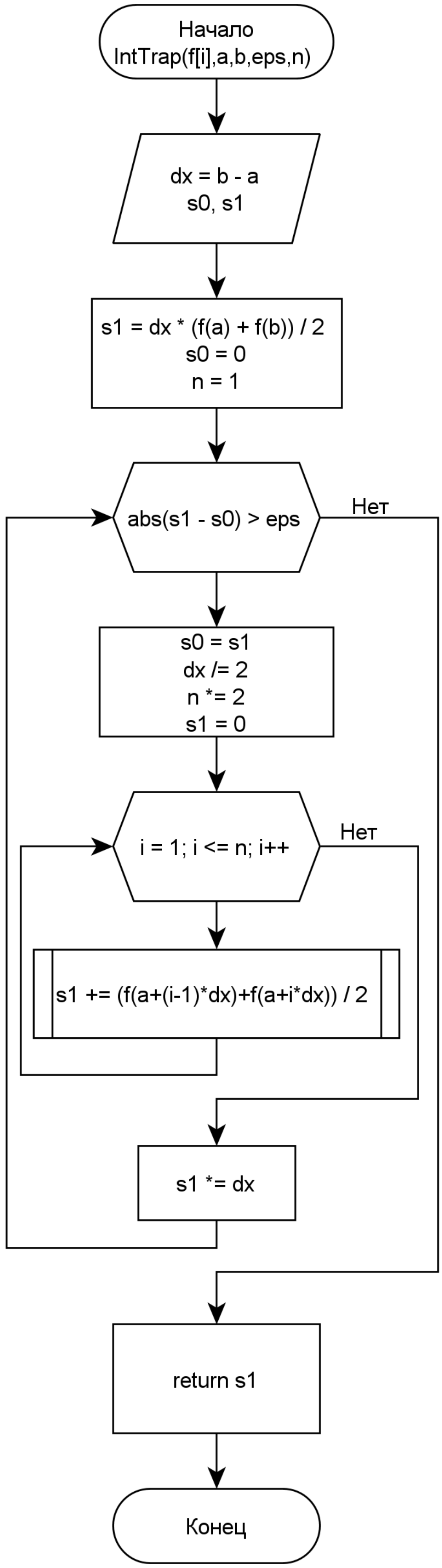
   b. *Локальные переменные:*
      - i. double dx – ширина трапеций;
      - ii. double s0 – предыдущее значение интеграла;
      - iii. double s1 – текущее значение интеграла;

   c. Возвращенное значение:
      - i. Возращенное значение: s1;




2
j = 0; j < 2; j++
Нет
1
Нет
eps = 0.01; eps >= 0.000001; eps /= 10
i = 0; i < 4; i++
Нет
switch (i)
PrintTabl(i_prn, 4)
case 0:
i_prn[i].i_sum = IntRect(f[i],a,b,eps,n)
break
case 1:
i_prn[i].i_sum = IntTrap(f[i],a,b,eps,n)
break
i_prn[i].n = n
1
Конец


Начало
IntPect(f[i],a,b,eps,n)
dx = b - a
s0, s1
s1 = dx * f(a + dx/2)
s0 = 0
n = 1
abs(s1 - s0) > eps
Нет
s0 = s1
dx /= 2
n *= 2
s1 = 0
i = 1; i <= n; i++
Нет
s1 += f(a + dx / 2 + (i - 1) * dx)
s1 *= dx
return s1
Конец


Начало
IntTrap(f[i],a,b,eps,n)
dx = b - a
s0, s1
s1 = dx * (f(a) + f(b)) / 2
s0 = 0
n = 1
abs(s1 - s0) > eps
Нет
s0 = s1
dx /= 2
n *= 2
s1 = 0
i = 1; i <= n; i++
Нет
s1 += (f(a+(i-1)*dx)+f(a+i*dx)) / 2
s1 *= dx
return s1
Конец

Текст программы

Файл Lab5.cpp

```
#pragma once
#include <iostream>
#include "str.h"
#include <iomanip>
#include <cmath>
using namespace std;
int main()
{
	TPF f[4] = { f1, f2, f3, f4 };
	I_print i_prn[4];
	int n = 0;
	double a = -1, b = 3;
	cout << "Function integration domain: " << a << " <= x <= " << b << endl;
	i_prn[0].name = "y=x"; i_prn[0].i_toch = (b * b - a * a) / 2.0;
	i_prn[1].name = "y=sin(22x)"; i_prn[1].i_toch = (cos(a * 22.0) - cos(b * 22.0)) / 
22.0;
	i_prn[2].name = "y=x^4"; i_prn[2].i_toch = (b * b * b * b * b - a * a * a * a * a) / 5.0;
	i_prn[3].name = "y=arctg(x)"; i_prn[3].i_toch = b * atan(b) - a * atan(a) - (log(b * b 
+ 1) - log(a * a + 1)) / 2.0;
	for (int j = 0; j < 2; j++)
	{
		switch (j)
		{
			case 0: cout << "Rectangle method" << endl; break;
			case 1: cout << "Trapezoid method" << endl; break;
		}
		for (double eps = 0.01; eps >= 0.000001; eps /= 10)
		{
			cout << "Calculation accuracy = " << fixed << setprecision(6) << 
eps << endl;
			for (int i = 0; i < 4; i++)
			{
				switch (j)
				{
					case 0: i_prn[i].i_sum = IntRect(f[i], a, b, eps, n); 
break;
					case 1: i_prn[i].i_sum = IntTrap(f[i], a, b, eps, n); 
break;
				}
				i_prn[i].n = n;
			}
			PrintTabl(i_prn, 4);
		}
	}
	return 0;
}
```

Файл str.h

```
#pragma once
struct I_print {
```

```
	const char *name;
	double i_sum;
	double i_toch;
	int n;
};

typedef double(*TPF)(double);
void PrintTabl(I_print[], int);
double IntRect(TPF, double, double, double, int&);
double IntTrap(TPF, double, double, double, int&);
double f1(double);
double f2(double);
double f3(double);
double f4(double);
```

Файл func.cpp

```
#include <iostream>
#include "str.h"
#include <iomanip>
#include <cmath>
using namespace std;
void PrintTabl(I_print i_prn[], int k)
{
	const int m = 4;
	int wn[m] = { 12,18,18,10 };
	char title[m][9] = { "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) + 1) / 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;
        }
        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;
        cout << endl;
}

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);
}

double IntRect(TPF f, double a, double b, double eps, int& n)
{
        double dx = b - a, s0, s1;
        s1 = dx * f(a + dx/2);
        s0 = 0;
        n = 1;
        while (abs(s1 - s0) > eps)
        {
                s0 = s1;
                dx /= 2;
                n *= 2;
                s1 = 0;
                for (int i = 1; i <= n; i++)
                        s1 += f(a + dx / 2 + (i - 1) * dx);
                s1 *= dx;
        }
        return s1;
}

double IntTrap(TPF f, double a, double b, double eps, int& n)
{
        double dx = b - a, s0, s1;
        s1 = dx * (f(a) + f(b)) / 2;
```

```
        s0 = 0;
        n = 1;
        while (abs(s1 - s0) > eps)
        {
                s0 = s1;
                dx /= 2;
                n *= 2;
                s1 = 0;
                for (int i = 1; i <= n; i++)
                        s1 += (f(a + (i - 1) * dx) + f(a + i * dx)) / 2;
                s1 *= dx;
        }
        return s1;
}
```

Анализ результатов

№	Входные данные	Полученный результат			
1	Метод прямоугольников eps = 0.01	Function	Integral	IntSum	N
		y=x	4.0000000000	4.0000000000	2
		y=sin(22x)	-0.0000142441	0.0001110392	8
		y=x^4	48.8000000000	48.7988606840	256
		y=arctg(x)	2.1570201976	2.1580597140	16
2	Метод прямоугольников eps = 0.001	Function	Integral	IntSum	N
		y=x	4.0000000000	4.0000000000	2
		y=sin(22x)	-0.0000142441	0.0001110392	8
		y=x^4	48.8000000000	48.7997151697	512
		y=arctg(x)	2.1570201976	2.1572804810	32
3	Метод прямоугольников eps = 0.0001	Function	Integral	IntSum	N
		y=x	4.0000000000	4.0000000000	2
		y=sin(22x)	-0.0000142441	0.0001110392	8
		y=x^4	48.8000000000	48.7999821981	2048
		y=arctg(x)	2.1570201976	2.1570364731	128
4	Метод прямоугольников eps = 0.00001	Function	Integral	IntSum	N
		y=x	4.0000000000	4.0000000000	2
		y=sin(22x)	-0.0000142441	-0.0000154313	64
		y=x^4	48.8000000000	48.7999988874	8192
		y=arctg(x)	2.1570201976	2.1570212148	512
5	Метод прямоугольников eps = 0.000001	Function	Integral	IntSum	N
		y=x	4.0000000000	4.0000000000	2
		y=sin(22x)	-0.0000142441	-0.0000145286	128
		y=x^4	48.8000000000	48.7999997218	16384
		y=arctg(x)	2.1570201976	2.1570204519	1024

		Function	Integral	IntSum	N
6	Метод трапеций eps = 0.01	y=x	4.0000000000	4.0000000000	2
		y=sin(22x)	-0.0000142441	-0.0354024633	2
		y=x^4	48.8000000000	48.8022786379	256
		y=arctg(x)	2.1570201976	2.1549393190	16
7	Метод трапеций eps = 0.001	y=x	4.0000000000	4.0000000000	2
		y=sin(22x)	-0.0000142441	-0.0354024633	2
		y=x^4	48.8000000000	48.8001424153	1024
		y=arctg(x)	2.1570201976	2.1568899987	64
8	Метод трапеций eps = 0.0001	y=x	4.0000000000	4.0000000000	2
		y=sin(22x)	-0.0000142441	-0.0354024633	2
		y=x^4	48.8000000000	48.8000089010	4096
		y=arctg(x)	2.1570201976	2.1569876461	128
9	Метод трапеций eps = 0.00001	y=x	4.0000000000	4.0000000000	2
		y=sin(22x)	-0.0000142441	-0.0354024633	2
		y=x^4	48.8000000000	48.8000022252	8192
		y=arctg(x)	2.1570201976	2.1570181631	512
10	Метод трапеций eps = 0.000001	y=x	4.0000000000	4.0000000000	2
		y=sin(22x)	-0.0000142441	-0.0000141036	256
		y=x^4	48.8000000000	48.8000001391	32768
		y=arctg(x)	2.1570201976	2.1570200704	2048

Вывод: При маленьких точностях некоторые значения получаются неверные, это происходит из-за изменения знака графиков на интегрируемой области, но при увеличении точности уменьшаются размеры прямоугольников, из-за этого изменение знака меньше влияет на получаемое значение.