МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №5

по дисциплине «Вычислительная математика»

Тема: Численное вычисление определенного интеграла

Студент гр. 0304 _____ Алексеев Р.В.

Преподаватель _____ Попова Е.В.

Санкт-Петербург

2021

Вариант 1

Цель работы.

Изучение и сравнение различных методов численного интегрирования на примере составных формул прямоугольников, трапеций, Симпсона.

Основные теоретические положения.

Пусть требуется найти определенный интеграл

$$I = \int_a^b f(x)\,dx,$$

где функция f(x) непрерывна на отрезке [a,b]. Для приближенного вычисления интегралов чаще всего подынтегральную функцию заменяют «близкой» ей вспомогательной функцией, интеграл от которой вычисляется аналитически. За приближенное значение интеграла принимают значение интеграла от вспомогательной функции.

Заменим функцию на отрезке [a,b] её значением в середине отрезка. Искомый интеграл, равный площади криволинейной фигуры, заменяется на площадь прямоугольника. Из геометрических соображений нетрудно записать формулу прямоугольников:

$$\int_a^b f(x)dx \approx f\left(\frac{a+b}{2}\right)(b-a).$$

Приблизив f(x) линейной функцией и вычислив площадь соответствующей трапеции, получим формулу трапеций:

$$\int_a^b f(x)dx \approx 1/2\,(f(a)+f(b))(b-a).$$

Если же приблизить подынтегральную функцию параболой, проходящей через точки $(a, f(a))$, $(a+b/2, f((a+b)/2))$, $(b, f(b))$, то получим формулу Симпсона (парабол):

$$\int_a^b f(x)dx \approx 1/6\left(f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right)(b-a).$$

Для повышения точности интегрирования применяют составные формулы. Для этого разбивают отрезок [*a*, *b*] на четное *n* = 2*m* число отрезков длины h = (*b*−*a*)/*n* и на каждом из отрезков длины 2h применяют соответствующую формулу. Таким образом получают составные формулы прямоугольников, трапеций и Симпсона.

На сетке *xi* = *a* + *ih*, *yi*= f(*xi*), *i* = $\overline{0,2m}$ составные формулы имеют следующий вид:

Формула прямоугольников:

$$\int_a^b f(x)dx = h\sum_{i=0}^{n-1} f\left(x_i+\frac{h}{2}\right)+R_1$$

$$R_1 \approx \frac{h^2}{24}\int_a^b f''(x)dx$$

Формула трапеций:

$$\int_a^b f(x)dx = h\sum_{i=0}^{n-1}\left(f(x_i)+f(x_{i+1})\right)+R_2$$

$$R_2 \approx -\frac{h^2}{12}\int_a^b f''(x)dx$$

Формула Симпсона:

$$\int_a^b f(x)dx = h/3\sum_{i=0}^{m-1}\left(f(x_{2i})+4f(x_{2i+2})\right)+R_3$$

$$R_3 \approx -\frac{h^4}{180}\int_a^b f^{IV}(x)dx$$

где *R*1, *R*2, *R*3 – остаточные члены. Оценки остаточных членов получены в предположении, что соответствующие производные f(x) непрерывны на [a,b]. При $n \to \infty$ приближенные значения интегралов для всех трех формул (в предположении отсутствия погрешностей округления) стремятся к точному значению интеграла.

Для практической оценки погрешности квадратурной можно использовать правило Рунге. Для этого проводят вычисления на сетках с шагом h и h/2, получают приближенные значения интеграла Ih и $Ih/2$ за окончательные значения интеграла принимают величины:

- $I_{h/2}+\frac{I_{¿}}{3}$ – для формулы прямоугольников;
- $I_{h/2}-\frac{I_{¿}}{3}$ – для формулы трапеций;
- $I_{h/2}-\frac{I_{¿}}{15}$ – для формулы Симпсона.

При этом за погрешность приближённого значения интеграла принимается величина $|\frac{I_{¿}}{2^k-1}|$, причём для формул прямоугольников и трапеций k = 2, для формулы Симпсона k = 4.

Такую оценку погрешностей применяют обычно для построения адаптивных алгоритмов, т.е. таких алгоритмов, которые автоматически так определяют величину шага h, что результат удовлетворяет требуемой точности.

Порядок выполнения работы.

1. Составить подпрограмму-функцию F для вычисления подынтегральной функции.

2. Составить подпрограммы-функции RECT, TRAP, SIMPS для вычисления интегралов по формулам прямоугольников, трапеций и Симпсона соответственно.

3. Составить головную программу, содержащую оценку по Рунге погрешности каждой из перечисленных ранее квадратурных формул, удваивающих n до тех пор, пока погрешность не станет меньше ε, и осуществляющих печать результатов значения интеграла и значения n для каждой формулы.

4. Провести вычисления по программе, добиваясь, чтобы результат удовлетворял требуемой точности.

Выполнение работы.

Интеграл: $\int_0^1 \cos(x+x^3)\,dx$

1. При помощи сайта www.kontrolnaya-rabota.ru было найдено значение интеграла, равное 0,633446120160759. График функции представлен на рис. 1.

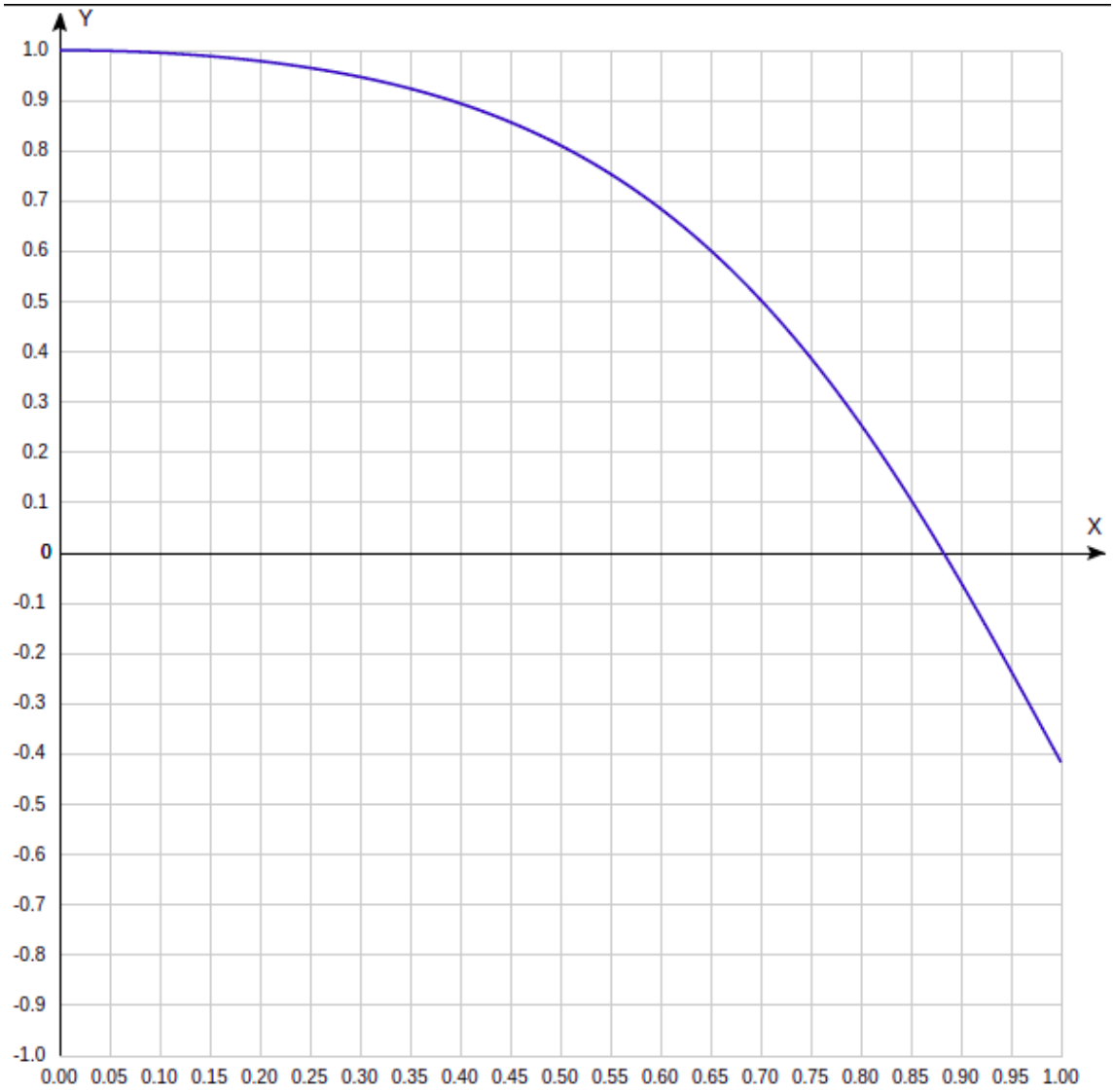

Рисунок 1 — график функции $f(x)=\cos(x+x^3)\,dx$

2. С помощью методов прямоугольников, трапеций и Симпсона было найдено количество разбиений отрезка, требуемое для получения результата с необходимой точностью eps = 0,000001. Результаты работы программы представлены на рис. 2.

Метод	Eps	Result	Diff result	Погреш	n
Прямоугольников	0.000001	0.633447	0.000001	0.000001	390
Трапеций	0.000001	0.633445	0.000001	0.000001	552
Симпсона	0.000001	0.633447	0.000001	0.000001	14

Рисунок 2 — Результаты вычисления тремя методами при eps = 0,000001.

По рисунку видно, что метод Симпсона требует меньше разбиений, больше всех требует метод трапеций, следовательно метод Симпсона является самым быстрым из трех. Погрешности полученных значений разными методами равны для eps = 0,000001. Вычисленные значения всеми тремя методами примерно равны.

3. С помощью трех методов было вычисленно количество разбиений отрезка, требуемое для получения результата с точностью eps = 0,00001. Результаты работы программы представлены на рис. 3.

Метод	Eps	Result	Diff result	Погреш	n
Прямоугольников	0.000010	0.633460	0.000010	0.000010	124
Трапеций	0.000010	0.633440	0.000010	0.000010	176
Симпсона	0.000010	0.633450	0.000007	0.000007	8

Рисунок 3 — Результаты вычисления тремя методами при eps = 0,00001.

На рисунке видно, что при eps = 0,00001 наибольшее количество разбиений совершает метод трапеций (176), а наименьшее — метод Симпсона (8). Оценка погрешности для метода Симпсона (0,000007) меньше чем для методов прямоугольников и трапеций (0,00001). Вычисленные значения всеми тремя методами примерно равны.

4. С помощью методов прямоугольников, трапеций и Симпсона найдено количество разбиений, требуемое для вычисления результата с

точностью eps = 0,0001. Результаты работы программы представлены на рис. 4.

Метод	Eps	Result	Diff result	Погреш	n
Прямоугольников	0.000100	0.633500	0.000095	0.000095	40
Трапеций	0.000100	0.633300	0.000097	0.000097	56
Симпсона	0.000100	0.633500	0.000022	0.000021	6

Рисунок 4 - Результаты вычисления тремя методами при eps = 0,0001.

По рисунку видно, что метод Симпсона требует меньшее количество разбиений, равное 6, методы трапеций и метод прямоугольников требуют 56 и 40 разбиений соответственно, следовательно метод Симпсона является самым быстрым из трех. Наименьшей погрешностью обладает значение, вычисленное методом Симпсона (0,000021), погрешности в методах трапеций и прямоугольников примерно равны и больше чем в методе Симпсона (0,000097 и 0,000095 соответсвенно). Вычисленные значения всеми тремя методами примерно равны.

5. С помощью методов прямоугольников, трапеций и Симпсона было найдено количество разбиений отрезка, требуемое для получения результата с точностью eps = 0,001. Результаты работы программы представлены на рис. 5.

Метод	Eps	Result	Diff result	Погреш	n
Прямоугольников	0.001000	0.634000	0.000776	0.000786	14
Трапеций	0.001000	0.633000	0.000937	0.000941	18
Симпсона	0.001000	0.634000	0.000110	0.000092	4

Рисунок 5 - Результаты вычисления тремя методами при eps = 0,001.

На рисунке видно, что при eps = 0,001 наибольшее количество разбиений совершает метод трапеций, а именно 18, наименьшее — метод Симпсона (8), метод прямоугольников совершает 14 разбиений. Оценка погрешности для метода Симпсона равна 0,000092, для метода трапеций — 0,000941, для метода прямоугольников — 0,000786, следовательно у метода Симпсона

наименьшая погрешность. Вычисленные значения всеми тремя методами примерно равны.

6. Был построен график зависимостей количества разбиений от eps (N(eps)) для всех трех методов, смотрите рис. 6.

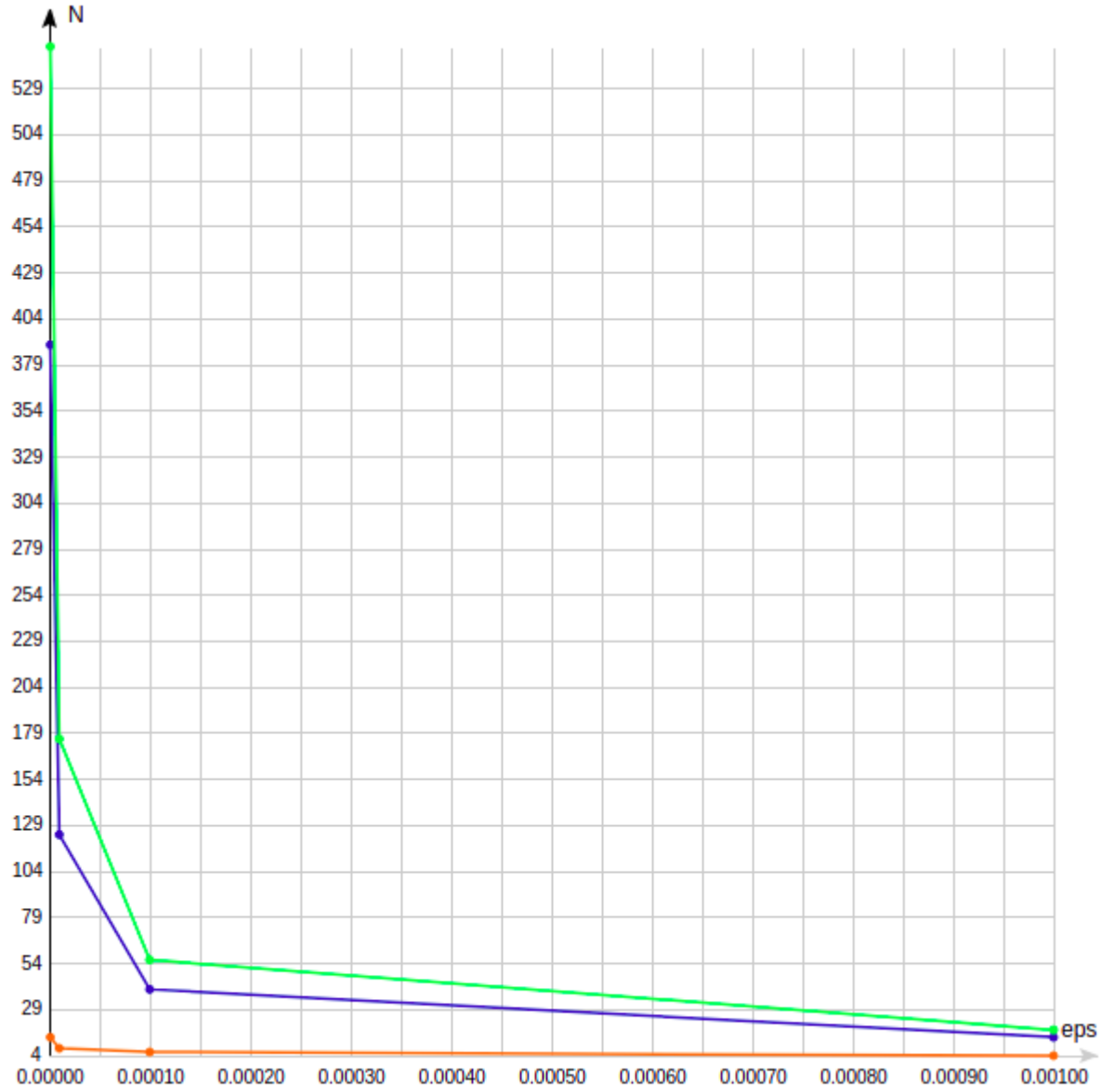

Рисунок 6 — График зависимости N(eps) для всех трех методов.

На графике синяя линяя — метод прямоугольников, зеленая линяя — метод трапеций, оранжевая линия — метод Симпсона. По графику видно, что график метода Симпсона находится ниже остальных, следовательно количество разбиений для метода Симпсона меньше чем для других методов. Это обусловлено тем, что в методах прямоугольников и трапеций

подынтегральная функция приближается с помощью прямоугольников и трапеций соответсвенно, а в методе Симпсона с помощью параболы.

Разработанный программный код см. в приложении А.

Выводы.

Были исследованны три метода численного вычисления определенного интегралы: прямоугольников, трапеций, Симпсона. Было доказано, что методу Симпсона требуется меньше разбиений чем двум другим методам, следовательно метод Симпсона быстрее. Также было выявлено, что погрешность для метода Симпсона меньше погрешностей для методов прямоугольников и трапеций.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

```
#include <iostream>
#include <cstdio>
#include <cmath>

const double internet_answer = 0.633446120160759;

double f(double x){
	return cos(x + x*x*x);
}

double ROUND(double x, double delta){
	if(delta <= 1e-9){
		puts("Неверно задана точность округления!");
		exit(1);
	}
	if(x > 0.0)
		return delta * (long((x / delta) + 0.5));
	else
		return delta * (long((x / delta) - 0.5));
}

	double RECT(double a, double b, double eps, int& n, double internet_answer, double& inaccuracy){
		n = 2;
		double h, integral, prev_integral;
		while(true){
			integral = 0;
			h = (b - a) / n;
			for (size_t i = 0; i < n; ++i)
				integral += f(a + (i + .5) * h);
			integral *= h;
			if(fabs(internet_answer - integral) < eps)
				break;
			n += 2;
		}
		prev_integral = 0;
		for(size_t i = 0; i < n / 2; ++i)
			prev_integral += f(a + (i + .5) * 2 * h);
		prev_integral *= 2*h;
		inaccuracy = fabs(integral - prev_integral) / 3.0;
		return integral;
	}

	double TRAP(double a, double b, double eps, int& n, double internet_answer, double& inaccuracy){
		n = 2;
		double h, integral, prev_integral;
		while(true){
			integral = 0;
			h = (b - a) / n;
```

```
                integral += (f(a) + f(b)) / 2.0;
                for(size_t i = 1; i < n; ++i)
                     integral += f(a + h*i);
                integral *= h;
                if(fabs(internet_answer - integral) < eps)
                     break;
                n += 2;
           }
           prev_integral = (f(a) + f(b)) / 2.0;
           for(size_t i = 1; i < n / 2; ++i)
                prev_integral += f(a + 2 * h * i);
           prev_integral *= 2*h;
           inaccuracy = fabs(integral - prev_integral) / 3.0;
           return integral;
     }

     double SIMPS(double a, double b, double eps, int& n, double
internet_answer, double& inaccuracy){
           n = 2;
           double h, integral, prev_integral;
           while (true){
                integral = 0;
                h = (b - a) / n;
                for(size_t i = 0; i < n; ++i)
                     integral += f(a + i*h) + 4*f(a + (i + .5)*h) +
f(a + (i + 1)*h);
                integral *= h / 6.0;
                if(fabs(internet_answer - integral) < eps)
                     break;
                n += 2;
           }
           prev_integral = 0;
           for(size_t i = 0; i < n / 2; ++i)
                prev_integral += f(a + 2*i*h) + 4*f(a + 2*(i + .5)*h)
+ f(a + 2*(i + 1)*h);
           prev_integral *= 2 * h / 6.0;
           inaccuracy = fabs(integral - prev_integral) / 15.0;
           return integral;
     }

     int main()
     {
           int n;
           double a = 0.0, b = 1.0;
           double result, eps, diff_result, inac;

           eps = 0.000001;


puts("-----------------------------------------------------------------
----------------------------");
           printf("      Метод\t%12s\t%10s\t%10s\t%10s\t%6s\n", "Eps",
"Result", "Diff result", "Погреш", "n");

puts("-----------------------------------------------------------------
----------------------------");
```

```
            printf("Прямоугольников\t       %-6f    ", eps);
            result = RECT(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f     %8f%8d\n", result, diff_result, inac,
n);
            printf("Трапеций\t       %-6f    ", eps);
            result = TRAP(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f     %8f%8d\n", result, diff_result, inac,
n);
            printf("Симпсона\t       %-6f    ", eps);
            result = SIMPS(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f     %8f%8d", result, diff_result, inac,
n);
            puts("\
n-----------------------------------------------------------------------
----------------------");


            eps = 0.00001;


puts("-----------------------------------------------------------------
----------------------------");
            printf("      Метод\t%12s\t%10s\t%10s\t%10s\t%6s\n", "Eps",
"Result", "Diff result", "Погреш", "n");

puts("-----------------------------------------------------------------
----------------------------");
            printf("Прямоугольников\t       %-6f    ", eps);
            result = RECT(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f     %8f%8d\n", result, diff_result, inac,
n);
            printf("Трапеций\t       %-6f    ", eps);
            result = TRAP(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f     %8f%8d\n", result, diff_result, inac,
n);
            printf("Симпсона\t       %-6f    ", eps);
            result = SIMPS(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f     %8f%8d", result, diff_result, inac,
n);
            puts("\
n-----------------------------------------------------------------------
----------------------");
```

```
            eps = 0.0001;


puts("-------------------------------------------------------------------
----------------------------");
            printf("       Метод\t%12s\t%10s\t%10s\t%10s\t%6s\n", "Eps",
"Result", "Diff result", "Погреш", "n");

puts("-------------------------------------------------------------------
----------------------------");
            printf("Прямоугольников\t       %-6f    ", eps);
            result = RECT(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f     %8f%8d\n", result, diff_result, inac,
n);
            printf("Трапеций\t       %-6f    ", eps);
            result = TRAP(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f     %8f%8d\n", result, diff_result, inac,
n);
            printf("Симпсона\t       %-6f    ", eps);
            result = SIMPS(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f      %8f%8d", result, diff_result, inac,
n);
            puts("\
n-------------------------------------------------------------------
-----------------------");


            eps = 0.001;


puts("-------------------------------------------------------------------
----------------------------");
            printf("       Метод\t%12s\t%10s\t%10s\t%10s\t%6s\n", "Eps",
"Result", "Diff result", "Погреш", "n");

puts("-------------------------------------------------------------------
----------------------------");
            printf("Прямоугольников\t       %-6f    ", eps);
            result = RECT(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f     %8f%8d\n", result, diff_result, inac,
n);
            printf("Трапеций\t       %-6f    ", eps);
            result = TRAP(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f     %8f%8d\n", result, diff_result, inac,
n);
            printf("Симпсона\t       %-6f    ", eps);
```

```
            result = SIMPS(a, b, eps, n, internet_answer, inac);
            diff_result = fabs(internet_answer - result);
            result = ROUND(result, eps);
            printf("%8f\t  %8f      %8f%8d", result, diff_result, inac,
n);
            puts("\
n------------------------------------------------------------------------
-----------------------");



            return 0;
      }
```