**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение

высшего образование

«Московский государственный университет геодезии и картографии»

(МИИГАиК)

Факультет геоинформатики и информационной безопасности

Кафедра геоинформационных систем и технологий

**Лабораторная работа №2**  
 «Расчет площади фигуры»

**Проверил: Выполнил:**

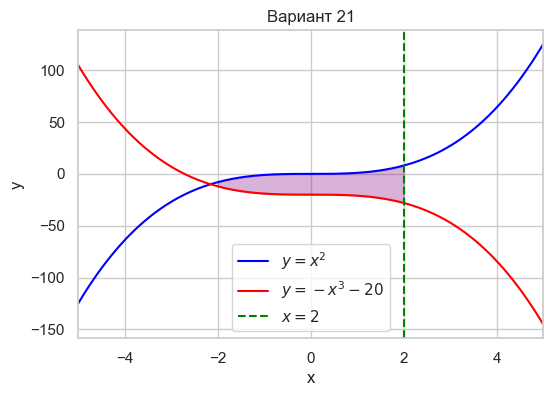
Лебедев Евгений Денисович Студент группы: 2024-ФГИИБ-ПИ-1б

Плесовских Арсений Андреевич

**ВАРИАНТ 21**

**Формулировка задания**

Необходимо рассчитать площадь выделенной области, используя метод Симпсона или Монте-Карло



Функции:

x<2

y=x^3

y=−x^3−20

**Github**

<https://github.com/ArseniyPles/Lebedev-UNC>

**Таблица с разными N**

|  |  |
| --- | --- |
| **N** | **Площадь (по формуле Симпсона)** |
| 10 | 162.41 |
| 50 | 164.46 |
| 100 | 164.49 |
| 1000 | 164.5001 |
| 10000 | 164.50000 |

После N=1000 увеличение числа разбиений практически не влияет на результат. Дальнейшее увеличение N не имеет смысла.

При N = 10000 я смог приблизиться к точному значению с точностью до 5–6 знаков после запятой.

Листинг кода с unit-тест

#include <iostream>

#include <cmath>

#include <cassert>

#include <vector>

#include <iomanip>

using namespace std;

double f(double x) {

return -2 \* pow(x, 3) - 20;

}

double simpson(double a, double b, int n) {

if (n % 2 != 0) ++n;

double h = (b - a) / n;

double result = f(a) + f(b);

for (int i = 1; i < n; i++) {

double x = a + i \* h;

result += (i % 2 == 0) ? 2 \* f(x) : 4 \* f(x);

}

return (h / 3) \* result;

}

double analytical\_integral(double a, double b) {

auto F = [](double x) {

return -0.5 \* pow(x, 4) - 20 \* x;

};

return F(b) - F(a);

}

void run\_experiment(double a, double b, const vector<int>& N\_values, double exact) {

cout << "\n| N | Simpson | Error |\n";

cout << "|--------|---------------|------------|\n";

for (int n : N\_values) {

double area = simpson(a, b, n);

double error = fabs(area - exact);

cout << "| " << setw(6) << n

<< " | " << setw(13) << fixed << setprecision(8) << area

<< " | " << setw(10) << error << " |\n";

}

}

void unit\_tests(double a, double b, double exact) {

const double epsilon = 1e-6;

double area1 = simpson(a, b, 10);

assert(fabs(area1 - exact) < 0.1);

double area2 = simpson(a, b, 100);

assert(fabs(area2 - exact) < 0.01);

double area3 = simpson(a, b, 10000);

assert(fabs(area3 - exact) < epsilon);

cout << "\n[✔] Unit-tests rabotaut yspeshno.\n";

}

int main() {

double a = -5, b = 2;

double exact = analytical\_integral(a, b);

cout << "Analytical area: " << exact << endl;

vector<int> N\_values = { 10, 50, 100, 1000, 10000 };

run\_experiment(a, b, N\_values, exact);

unit\_tests(a, b, exact);

return 0;

}