МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Институт математики и информационных систем
Факультет автоматики и вычислительной техники
Кафедра электронных вычислительных машин

Отчёт по лабораторной работе №3

по дисциплине

«Программирование»

Разработал студент гр. ИВТб-1301-05-00 _____/Черкасов А. А./

(подпись)

Заместитель кафедры ЭВМ _____/Долженкова М. Л./

(подпись)

Киров

2024

Цель

Цель работы: Освоить синтаксис построения процедур и функций, изучить способы передачи данных в подпрограммы, получить навыки организации минимального пользовательского интерфейса.

Задание

- Реализовать программу вычисления площади фигуры, ограниченной кривой $2 \times x^3 + 2 \times x^2 + 3 \times x + 1$ и осью OX (в положительной части оси OY).
- Вычисление определенного интеграла должно выполняться численно, с применением метода средних прямоугольников.
- Пределы интегрирования вводятся пользователем.
- Взаимодействие с пользователем должно осуществляться посредством case-меню.
- Требуется реализовать возможность оценки погрешности полученного результата.
- Необходимо использовать процедуры и функции там, где это целесообразно.

Решение

Схема алгоритма Задания представлена на Рисунке 1.
Исходный код представлен в Приложении А1.

Рисунок 1 - Схема алгоритма Задания.

Приложеие А1. Исходный код Задания.

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>

#define MAX_ITEMS 5

double curve(double x) {
    return 2 * pow(x, 3) + 2 * pow(x, 2) + 3 * x + 1;
}

double antiderivative(double x) { // Первообразная
    return 0.5 * pow(x, 4) + (2.0 / 3.0) * pow(x, 3) + 1.5 * pow(x, 2) + x;
}

double integrate(double a, double b, int n) {
    double width = (b - a) / n;
    double area = 0.0;
    for (int i = 0; i < n; i++) {
        double x = a + (i + 0.5) * width;
        double y = curve(x);
        if (y > 0) {
            area += y * width;
        }
    }
    return area;
}
```

```
void estimate_error(double a, double b, int n, double *abs_error, double *rel_error) {
    double numerical = integrate(a, b, n);
    double exact = antiderivative(b) - antiderivative(a);
    *abs_error = fabs(exact - numerical);
    *rel_error = (exact != 0) ? (*abs_error / fabs(exact)) * 100.0 : 0.0;
}

// Общие вспомогательные функции
void wait_and_return() {
    printf("Нажмите любую клавишу, чтобы вернуться в меню...\n");
    _getch();
}

double find_x() {
    double left = -1, right = 0, centre;
    while(right - left > 1e-10) {
    centre = (left + right) / 2;
    if(curve(centre) * curve(right) < 0)
        left = centre;
    else
        right = centre;
    }
    double foundX = (left + right) / 2;
    return foundX;
}

void input_limits(double *a, double *b) {
    printf("Введите пределы интегрирования (a b): ");
    scanf("%lf %lf", a, b);
    if(*a < find_x()) *a = find_x();
}

void input_rectangles(int *n) {
```

```
    printf("Введите количество прямоугольников: ");
    scanf("%d", n);
    if (*n <= 0) {
        printf("Количество прямоугольников должно быть положительным.\n");
        *n = 0;
    }
}

// Меню
void print_menu(int highlight, double a, double b, int n) {
    const char *choices[MAX_ITEMS] = {
        "Ввести верхний и нижний пределы интегрирования",
        "Ввести количество прямоугольников",
        "Вычислить площадь",
        "Найти погрешность",
        "Завершить"
    };
    system("cls");
    for (int i = 0; i < MAX_ITEMS; ++i) {
        if (highlight == i) printf("> %s (a: %lf, b: %lf, n: %d)\n", choices[i],\
a, b, n);
        else printf("%s\n", choices[i]);
    }
}

// Главная функция
int main() {
    int highlight = 0;
    int choice = -1;
    double a = 0, b = 0;
    int n = 0;
    int limits = 0, rect = 0, exit = 1;
```

```
while (exit) {
    print_menu(highlight, a, b, n);
    int c = _getch();
    // Скролл меню стрелками и выбор Enter
    switch (c) {
        case 224:
            switch (_getch()) {
                case 72: // Стрелка вверх
                    highlight = (highlight == 0) ? MAX_ITEMS - 1 : highlight - 1;
                    break;
                case 80: // Стрелка вниз
                    highlight = (highlight == MAX_ITEMS - 1) ? 0 : highlight + 1;
                    break;
            }
            break;
        case 13: // Enter
            choice = highlight;
            break;
        default:
            continue;
    }

    if (choice >= 0) {
        switch (choice) {
            case 0: // Ввести пределы
                input_limits(&a, &b);
                limits = 1;
                wait_and_return();
                break;
            case 1: // Ввести количество прямоугольников
                input_rectangles(&n);
                rect = (n > 0);
                wait_and_return();
```

```
                    break;
                case 2: // Вычислить площадь
                    if (limits && rect) {
                        double area = integrate(a, b, n);
                        printf("Площадь: %.6lf\n", area);
                    } else {
                        printf("Необходимо задать пределы и\
количество прямоугольников.\n");
                    }
                    wait_and_return();
                    break;
                case 3: // Найти погрешность
                    if (limits && rect) {
                        double abs_error, rel_error;
                        estimate_error(a, b, n, &abs_error, &rel_error);
                        printf("Абсолютная погрешность: %.6lf\n", abs_error);
                        printf("Относительная погрешность: %.6lf%%\n", rel_error);
                    } else {
                        printf("Необходимо задать пределы и\
количество прямоугольников.\n");
                    }
                    wait_and_return();
                    break;
                case 4: // Завершить
                    printf("Выход...\n");
                    exit = 0;
            }
            choice = -1; // Сброс выбора
        }
    }
    return 0;
}
```

Вывод