

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»

Выполнил студент гр. ИВТб-1303-06-00	_____ /Гортоломей И.К./
Проверил преподаватель кафедры ЭВМ	_____ /Баташев П.А./

Киров
2025

Цель

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

Задание

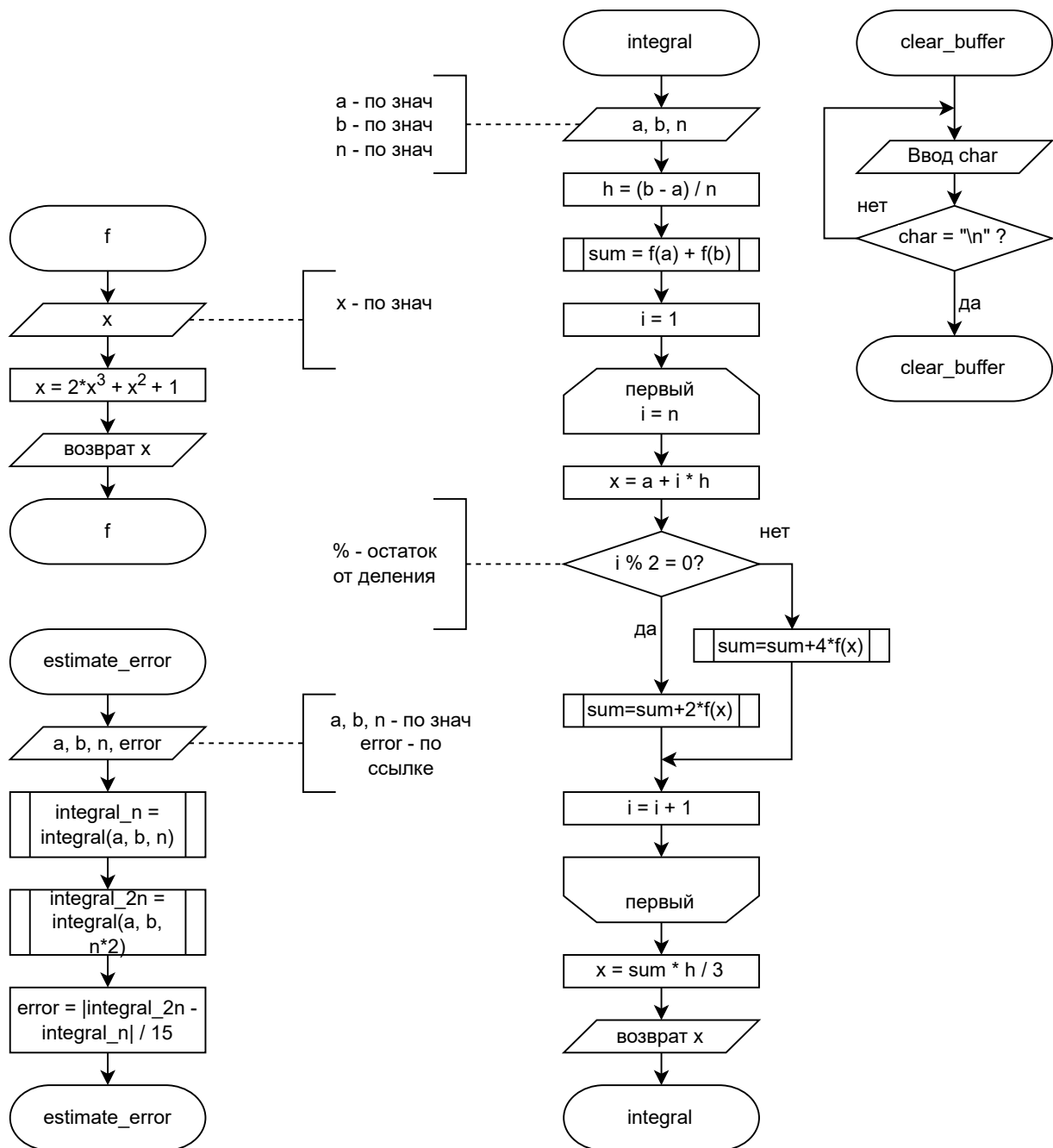
1. Реализовать программу вычисления площади фигуры, ограниченной кривой и осью OX (в положительной части по оси OY).
2. Вычисление определенного интеграла должно выполняться численно, с применением метода.
3. Пределы интегрирования вводятся пользователем.
4. Взаимодействие с пользователем должно осуществляться посредством case-меню.
5. Требуется реализовать возможность оценки погрешности полученного результата.
6. Необходимо использовать процедуры и функции там, где это целесообразно (программа должна содержать минимум одну процедуру, одну функцию, один пример передачи данных в подпрограммы по ссылке, один пример передачи данных в подпрограммы по значению).

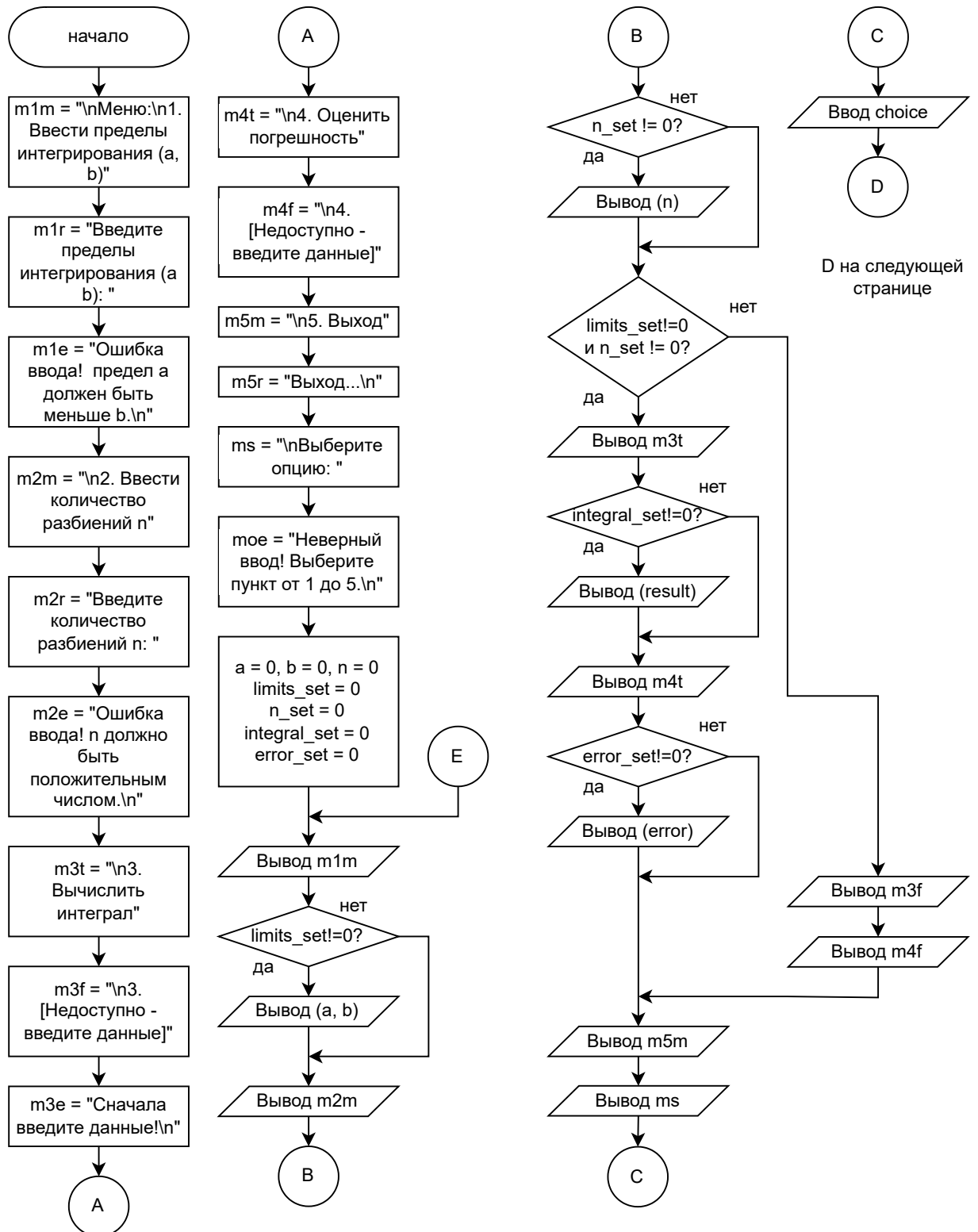
Дано:

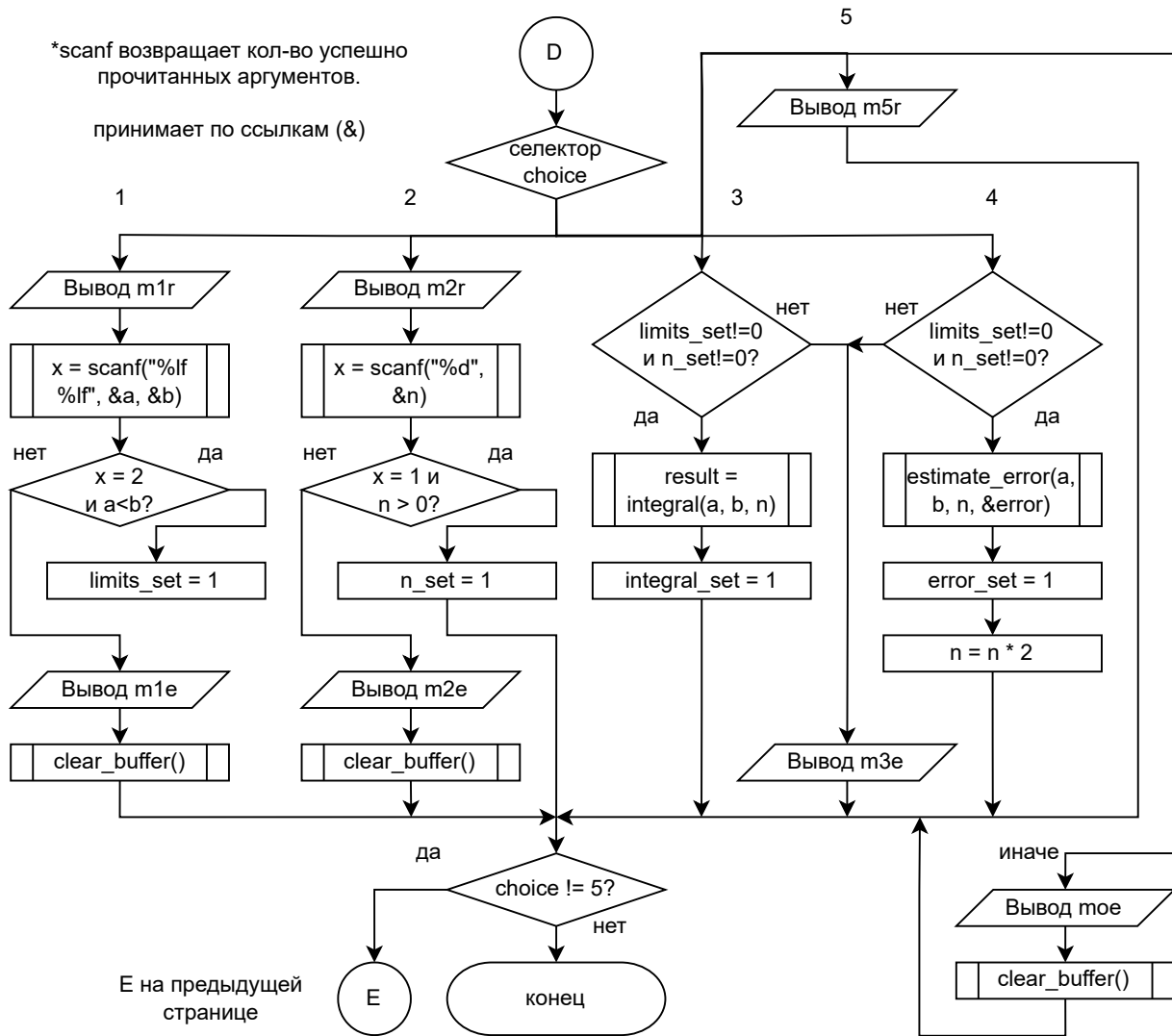
- Уравнение кривой: $2 * x^3 + 1 * x^2 + 0 * x + 1$
- Метод: Симпсона
- Язык: Си

Решение

Схема программы:







Код программы на C:

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

// Функция для вычисления значения кривой в точке x
double f(double x) {
    return 2 * pow(x, 3) + pow(x, 2) + 1;
}

// Функция вычисления интеграла методом Симпсона
double integral(double a, double b, int n) {
    double h = (b - a) / n;
    double sum = f(a) + f(b);

    for (int i = 1; i < n; i++) {
        double x = a + i * h;
        if (i % 2 == 0)
            sum += 2 * f(x);
        else
            sum += 4 * f(x);
    }
    return sum * h / 3;
}

// Процедура для оценки погрешности методом Рунге
void estimate_error(double a, double b, int n, double* error) {
    double integral_n = integral(a, b, n);
    double integral_2n = integral(a, b, n * 2);
    *error = fabs(integral_2n - integral_n) / 15;
}
```

```
}
```

```
void clear_buffer() {  
    while (getchar() != '\n');  
}
```

```
static char* m1m = "\nМеню:\n1. Ввести пределы интегрирования (a, b)";  
static char* m1r = "Введите пределы интегрирования (a b): ";  
static char* m1e = "Ошибка ввода! предел a должен быть меньше b.\n";  
static char* m2m = "\n2. Ввести количество разбиений n";  
static char* m2r = "Введите количество разбиений n: ";  
static char* m2e = "Ошибка ввода! n должно быть положительным числом.\n";  
static char* m3t = "\n3. Вычислить интеграл";  
static char* m3f = "\n3. [Недоступно - введите данные]";  
static char* m3e = "Сначала введите данные!\n";  
static char* m4t = "\n4. Оценить погрешность";  
static char* m4f = "\n4. [Недоступно - введите данные]";  
static char* m5m = "\n5. Выход";  
static char* m5r = "Выход...\n";  
static char* ms = "\nВыберите опцию: ";  
static char* moe = "Неверный ввод! Выберите пункт от 1 до 5.\n";
```

```
int main() {  
    double a = 0, b = 0, result, error; int n = 0, choice;  
    int limits_set = 0, n_set = 0, integral_set = 0, error_set = 0;  
    do {  
        printf(m1m);  
        if (limits_set) printf(" (%f, %f)", a, b);  
        printf(m2m);  
        if (n_set) printf(" (%d)", n);
```

```

if (limits_set && n_set) {
    printf(m3t);
    if (integral_set) printf(" (%.6f)", result);
    printf(m4t);
    if (error_set) printf(" (%.6f)", error);
} else {
    printf(m3f);
    printf(m4f);
}
printf(m5m);
printf(ms);
scanf("%d", &choice);

switch (choice) {
    case 1:
        printf(m1r);
        if (scanf("%lf %lf", &a, &b) == 2 && a < b) {
            limits_set = 1;
        } else {
            printf(m1e);
            clear_buffer();
        } break;

    case 2:
        printf(m2r);
        if (scanf("%d", &n) == 1 && n > 0) {
            n_set = 1;
        } else {
            printf(m2e);
            clear_buffer();
        }
}

```

```

    } break;

case 3:
    if (limits_set && n_set) {
        result = integral(a, b, n);
        integral_set = 1;
    } else printf(m3e); break;

case 4:
    if (limits_set && n_set) {
        estimate_error(a, b, n, &error);
        error_set = 1;
        n *= 2; // Увеличиваем n для следующего вычисления
    } else printf(m3e); break;

case 5:
    printf(m5r); break;

default:
    printf(moe);
    clear_buffer();
}
} while (choice != 5);
return 0;
}

```

Примеры работы программы:

```
Меню:  
1. Ввести пределы интегрирования (a, b)  
2. Ввести количество разбиений n  
3. [Недоступно - введите данные]  
4. [Недоступно - введите данные]  
5. Выход  
Выберите опцию: █
```

```
Меню:  
1. Ввести пределы интегрирования (a, b) (-5.000000, 5.000000)  
2. Ввести количество разбиений n (2)  
3. Вычислить интеграл  
4. Оценить погрешность  
5. Выход  
Выберите опцию: █
```

```
Меню:  
1. Ввести пределы интегрирования (a, b) (-5.000000, 5.000000)  
2. Ввести количество разбиений n (4)  
3. Вычислить интеграл (93.333333)  
4. Оценить погрешность (0.000000)  
5. Выход  
Выберите опцию: █
```

Выводы

Во время выполнения лабораторной работы научился писать функции, передавать в них аргументы по ссылке или значению и возвращать из них результат. В программе был реализован простой пользовательский интерфейс в котором пользователь вводит пределы и разбиение. Также имеется функция для вычисления интеграла и оценки ошибки. Взаимодействие с пользователем реализованно в форме case-меню.