

Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

**Лабораторная работа №4 по курсу**

**«Операционные системы»**

Группа: М8О-211БВ-24

Студент: Бахшалиев М.А.

Преподаватель: Бахарев В.Д.

Оценка: \_\_\_\_\_

Дата: 26.11.25

Москва, 2025

# Постановка задачи

## Вариант 5.

Требуется создать динамические библиотеки, которые реализуют заданный вариантом функционал. Далее использовать данные библиотеки 2-мя способами:

1. Во время компиляции (на этапе линковки/linking)
2. Во время исполнения программы. Библиотеки загружаются в память с помощью интерфейса ОС для работы с динамическими библиотеками.

Функции:

1. Расчет интеграла функции  $\sin(x)$  на отрезке  $[A, B]$  с шагом  $e$ :

Сигнатура функции: `float sin_integral(float a, float b, float e);`

- Реализация №1: Подсчет интеграла методом прямоугольников;
- Реализация №2: Подсчет интеграла методом трапеций.

6. Расчет значения числа  $e$  (основание натурального логарифма):

Сигнатура функции: `float e(int x);`

- Реализация №1:  $(1 + 1/x)^x$
- Реализация №2: Сумма ряда по  $n$  от 0 до  $x$ , где элементы ряда равны:  $(1 / (n!))$

## Общий метод и алгоритм решения

В рамках данной работы была реализована программа для вычисления интеграла функции  $\sin(x)$  и значения числа  $e$  двумя различными математическими методами для каждой задачи.

Общий подход к решению заключался в создании модульной архитектуры, позволяющей использовать различные алгоритмы как в статическом, так и в динамическом режиме. Основная идея заключалась в разделении реализации алгоритмов в отдельные библиотечные модули с единым интерфейсом, что обеспечивает гибкость использования и легкость расширения.

Для вычисления интеграла  $\sin(x)$  были реализованы два численных метода:

- Метод прямоугольников: алгоритм разбивает отрезок  $[A, B]$  на элементарные отрезки шириной  $e$  и вычисляет сумму площадей прямоугольников с высотой, равной значению  $\sin(x)$  в начале каждого отрезка.
- Метод трапеций: алгоритм также разбивает отрезок на элементарные части, но вычисляет сумму площадей трапеций, образованных значениями  $\sin(x)$  на концах каждого отрезка, что обеспечивает более высокую точность приближения.

Для вычисления числа  $e$  были применены два различных подхода:

- Предельное определение: алгоритм вычисляет выражение  $(1 + 1/x)^x$  для заданного  $x$ , где с ростом  $x$  результат стремится к истинному значению числа  $e$ .

- Разложение в ряд Тейлора: алгоритм вычисляет сумму ряда  $1/n!$  для  $n$  от 0 до  $x$ , что обеспечивает быструю сходимость к точному значению числа  $e$  даже при относительно небольших  $x$ .

Архитектурное решение предусматривает создание двух отдельных библиотек ([libimpl1.so](#) и [libimpl2.so](#)), экспортирующих единые функции `sin_integral(float a, float b, float e)` и `e(int x)` через общий заголовочный файл `math_lib.h`. Каждая библиотека содержит различные реализации этих функций: [libimpl1.so](#) использует метод прямоугольников и предельное определение, а [libimpl2.so](#) — метод трапеций и разложение в ряд.

Для демонстрации работы реализованы две тестовые программы: `test_static`, статически линкующаяся с одной из библиотек на этапе компиляции, и `test_dynamic`, динамически загружающая библиотеки во время выполнения с возможностью переключения между ними по команде "0" без перезапуска программы. Динамическая версия использует механизм `dlopen/dlsym` для загрузки символов из разделяемых библиотек, что обеспечивает инкапсуляцию алгоритмов и возможность их горячей замены. Такой подход демонстрирует принципы модульного программирования и позволяет наглядно сравнить эффективность различных математических методов при одинаковых параметрах точности.

## Код программы

### lib.h:

```
#pragma once

#include <stdlib.h>

float sin_integral(float a, float b, float e);

float e(int x);
```

### lib1.c:

```
#include "../include/lib.h"
#include <math.h>

// Реализация №1: Метод прямоугольников
float sin_integral(float a, float b, float e) {
    float integral = 0.0;
    for (float x = a; x < b; x += e) {
        integral += sin(x) * e;
    }
    return integral;
}

// Реализация №1: Формула  $(1 + 1/x)^x$ 
float e(int x) {
    return pow(1.0 + 1.0 / x, x);
}
```

```
}
```

### **lib2.c:**

```
#include "../include/lib.h"
#include <math.h>

// Реализация №2: Метод трапеций
float sin_integral(float a, float b, float e) {
    float integral = 0.0;
    float n = (b - a) / e;
    for (int i = 0; i < n; i++) {
        float x1 = a + i * e;
        float x2 = a + (i + 1) * e;
        integral += (sin(x1) + sin(x2)) * e / 2;
    }
    return integral;
}

// Реализация №2: Сумма ряда
float e(int x) {
    float sum = 1.0;
    float factorial = 1.0;
    for (int i = 1; i <= x; i++) {
        factorial *= i;
        sum += 1.0 / factorial;
    }
    return sum;
}
```

### **example\_dynamic.c:**

```
#include <stdio.h>
#include <stdlib.h>
#include <dlfcn.h>

typedef float (*sin_integral_t)(float, float, float);
typedef float (*e_t)(int);

char *paths[] = {"./src/libown1.so", "./src/libown2.so"};
char *lib_names[] = {"Library 1 (Rectangles method, Formula)",
                    "Library 2 (Trapezoids method, Series sum)"};

sin_integral_t sin_integral_func;
e_t e_func;
void *lib = NULL;
```

```

int current_lib = 0;

void print_menu() {
    printf("Current library: %s\n", lib_names[current_lib]);
    printf("1 - Calculate integral of sin(x) from A to B\n");
    printf("2 - Calculate Euler's number e\n");
    printf("0 - Switch library implementation\n");
    printf("-1 - Exit\n");
    printf("Choose function: ");
}

int open_and_init(int ind) {
    if (lib) dlclose(lib);
    lib = dlopen(paths[ind], RTLD_LAZY);
    if (!lib) {
        printf("Error: Cannot load library %s\n", paths[ind]);
        return -1;
    }

    sin_integral_func = (sin_integral_t)dlsym(lib, "sin_integral");
    e_func = (e_t)dlsym(lib, "e");

    if (!sin_integral_func || !e_func) {
        printf("Error: Cannot load functions from library\n");
        return -1;
    }
    return 0;
}

int main() {
    int choice;

    printf("Available libraries:\n");
    printf("• Library 1: Rectangle method for integral, (1+1/x)^x for e\n");
    printf("• Library 2: Trapezoid method for integral, series sum for e\n\n");

    if (open_and_init(current_lib) != 0) {
        printf("Fatal error: Cannot load initial library\n");
        return 1;
    }

    print_menu();
    scanf("%d", &choice);
}

```

```

while (choice != -1) {
    if (choice == 0) {
        // Switch library
        current_lib = 1 - current_lib;
        if (open_and_init(current_lib) != 0) {
            printf("Error switching library\n");
            return 1;
        }
        printf("✓ Switched to: %s\n", lib_names[current_lib]);

    } else if (choice == 1) {
        float a, b, e_val;
        printf("\n--- Integral of sin(x) ---\n");
        printf("Enter A (start): ");
        scanf("%f", &a);
        printf("Enter B (end): ");
        scanf("%f", &b);
        printf("Enter E (step): ");
        scanf("%f", &e_val);

        float result = sin_integral_func(a, b, e_val);
        printf("Result:  $\int \sin(x)dx$  from %.2f to %.2f = %f\n", a, b, result);

    } else if (choice == 2) {
        int x;
        printf("\n--- Euler's number e ---\n");
        printf("Enter precision (integer > 0): ");
        scanf("%d", &x);

        if (x <= 0) {
            printf("Error: Precision must be positive integer\n");
        } else {
            float result = e_func(x);
            printf("Result:  $e \approx$  %f\n", result);
        }

    } else {
        printf("Invalid choice! Please try again.\n");
    }

    print_menu();
    scanf("%d", &choice);
}

```

```

    }

    if (lib) dlclose(lib);
    return 0;
}

```

### **example static.c:**

```

#include <stdio.h>
#include "include/lib.h"

void print_menu() {
    printf("1 - Calculate integral of sin(x) from A to B\n");
    printf("2 - Calculate Euler's number e\n");
    printf("0 - Exit\n");
    printf("Choose function: ");
}

int main() {
    int choice;

    printf("Available functions:\n");
    printf("1. Integral of sin(x) from A to B with step E\n");
    printf("2. Euler's number e approximation\n\n");

    print_menu();
    scanf("%d", &choice);

    while (choice != 0) {
        if (choice == 1) {
            float a, b, e_val;
            printf("\n--- Integral of sin(x) ---\n");
            printf("Enter A (start): ");
            scanf("%f", &a);
            printf("Enter B (end): ");
            scanf("%f", &b);
            printf("Enter E (step): ");
            scanf("%f", &e_val);

            float result = sin_integral(a, b, e_val);
            printf("Result:  $\int \sin(x)dx$  from %.2f to %.2f = %f\n", a, b, result);

        } else if (choice == 2) {
            int x;

```

```

        printf("\n--- Euler's number e ---\n");
        printf("Enter precision (integer > 0): ");
        scanf("%d", &x);

        if (x <= 0) {
            printf("Error: Precision must be positive integer\n");
        } else {
            float result = e(x);
            printf("Result: e ≈ %f\n", result);
        }

    } else {
        printf("Invalid choice! Please try again.\n");
    }

    print_menu();
    scanf("%d", &choice);
}

return 0;
}

```

## **Протокол работы программы**

Тестирование:

```

./example_static

1 - Calculate integral of sin(x) from A to B
2 - Calculate Euler's number e
0 - Exit

Choose function: 1

--- Integral of sin(x) ---

Enter A (start): 1

Enter B (end): 123

Enter E (step): 0.01

Result: ∫sin(x)dx from 1.00 to 123.00 = 1.430995

```

```

./example_dynamic

```

Available libraries:



- Library 1: Rectangle method for integral,  $(1+1/x)^x$  for e
- Library 2: Trapezoid method for integral, series sum for e

Current library: Library 1 (Rectangles method, Formula)

1 - Calculate integral of  $\sin(x)$  from A to B

2 - Calculate Euler's number e

0 - Switch library implementation

-1 - Exit

Choose function: 0

✓ Switched to: Library 2 (Trapezoids method, Series sum)

Current library: Library 2 (Trapezoids method, Series sum)

1 - Calculate integral of  $\sin(x)$  from A to B

2 - Calculate Euler's number e

0 - Switch library implementation

-1 - Exit

Choose function: 1

--- Integral of  $\sin(x)$  ---

Enter A (start): 1

Enter B (end): 50

Enter E (step): 0.001

Result:  $\int \sin(x) dx$  from 1.00 to 50.00 = -0.424662

Current library: Library 2 (Trapezoids method, Series sum)

1 - Calculate integral of  $\sin(x)$  from A to B

2 - Calculate Euler's number e

0 - Switch library implementation

-1 - Exit

Choose function: 2

--- Euler's number e ---

Enter precision (integer > 0): -1

Error: Precision must be positive integer

Current library: Library 2 (Trapezoids method, Series sum)

1 - Calculate integral of  $\sin(x)$  from A to B

2 - Calculate Euler's number e

0 - Switch library implementation

-1 - Exit

Choose function: -1

## Вывод

В результате выполнения лабораторной работы была успешно разработана и реализована программа, состоящая из динамических библиотек и двух тестовых приложений. Программа реализует функционал для вычисления интеграла функции  $\sin(x)$  и значения числа  $e$  (основания натурального логарифма).

Для каждой математической задачи были созданы две альтернативные реализации:

- \* Интеграл  $\sin(x)$  вычислялся методами прямоугольников и трапеций.
- \* Число  $e$  рассчитывалось с использованием предельного определения  $(1 + 1/x)^x$  и суммы ряда  $1/n!$ .

Программа наглядно продемонстрировала эффективность модульного подхода в программировании. Была создана гибкая архитектура, позволяющая использовать библиотеки двумя способами:

1. На этапе компиляции (статическая линковка), что обеспечивает простоту развертывания.
2. Во время исполнения (динамическая загрузка), что предоставляет максимальную гибкость.

Оба алгоритма для каждой задачи показали свою работоспособность, сходясь к ожидаемым математическим результатам при корректных входных данных. Динамическая версия программы особенно наглядно продемонстрировала ключевое преимущество разделяемых библиотек — возможность «горячей» замены алгоритмов (например, по команде «0») без остановки, перекompиляции или пересборки основной программы.