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»

Выполнила: В. А. Гузова
Группа: М8О-207БВ-24
Преподаватель: Е. С. Миронов

Москва, 2025

Условие

Цель работы:
Приобретение практических навыков в:

- Создание динамических библиотек
- Создание программ, которые используют функции динамических библиотек

Задание:
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

В конечном итоге, в лабораторной работе необходимо получить следующие части:

- Динамические библиотеки, реализующие контракты, которые заданы вариантом
- Тестовая программа (программа №1), которая используют одну из библиотек, используя информацию полученные на этапе компиляции
- Тестовая программа (программа №2), которая загружает библиотеки, используя только их относительные пути и контракты.

Провести анализ двух типов использования библиотек.

Пользовательский ввод для обоих программ должен быть организован следующим образом:

1. Если пользователь вводит команду «0», то программа переключает одну реализацию контрактов на другую (необходимо только для программы №2)
2. «1 arg1 arg2 . . . argN», где после «1» идут аргументы для первой функции, предусмотренной контрактами. После ввода команды происходит вызов первой функции, и на экране появляется результат её выполнения
3. «2 arg1 arg2 . . . argM», где после «2» идут аргументы для второй функции, предусмотренной контрактами. После ввода команды происходит вызов второй функции, и на экране появляется результат её выполнения

Вариант: 11
Функции:

1. § Рассчет производной функции cos(x) в точке A с приращением deltaX
    - Float Derative(float A, float deltaX)
    - Реализация 1: f’(x) = (f(A + deltaX) – f(A))/deltaX
    - Реализация 2: f’(x) = (f(A + deltaX) – f(A-deltaX))/(2*deltaX
2. Рассчет значения числа Пи при заданной длине ряда (K)
    - float Pi(int K)
    - Реализация 1: Ряд Лейбница
    - Реализация 2: Формула Валлиса

Метод решения

Алгоритм решения задачи:

1. Создаем 2 контракта в виде заголовочных файлов:
   - `proizv_contract.h` — объявление функции `float Derivative(float A, float deltaX);` для рассчета производной функции cos(x) в точке A с приращением deltaX.
   - `pi_contract.h` — объявление функции `float Pi(int K);` для рассчета значения числа Пи при заданной длине ряда (K)
2. Реализуются две пары динамических библиотек:
   - `liblibrary_first.so и liblibrary_leibniz.so` — первая формула для производной и ряд Лейбница.
   - `liblibrary_second.so liblibrary_wallis.so` — вторая формула для производной и формула Валлиса.
3. Создаем абстрактный слой для работы с динамическими библиотеками в виде заголовочного файла `dynamic_library.h` для поддержки кроссплатформенности, реализуем под Linux:
   - `LibHandle lib_open(const char* path);` — загрузка библиотеки.
   - `void* lib_sym(LibHandle h, const char* name);` — получение адреса функции по имени.
   - `int lib_close(LibHandle h);` — выгрузка библиотеки.
4. Создаем тестовую программу №1 (`program1`), которая использует функции `Derivative` и `Pi`, линкуясь на этапе компиляции с библиотеками `liblibrary_first.so и liblibrary_leibniz.so`.
5. Создаем тестовую программу №2 (`program2`), которая загружает библиотеки динамически во время выполнения, используя указатели на функции.

Описание программы

`proizv_contract.h`, `pi_contract.h` — заголовочные файлы-контракты, объявляющие функции `Derivative` и `Pi`.

`program1.c` — тестовая программа №1. Использует статическую (на этапе линковки) привязку к библиотекам `library_first.so и library_leibniz.so`. Реализует интерфейс команд:

- `1 A deltaX` — вызывает `Derivative` и выводит значение производной;
- `2 N` — вызывает `Sort` и выводит значение числа pi.

`program2.c` — тестовая программа №2. Загружает библиотеки динамически во время выполнения. Позволяет переключать реализацию по команде:

- `0` — переключение между двумя реализациями;
- `1 A deltaX`, `2 N` — работа с текущей реализацией контрактов.

`lib/first.cpp` — реализация контракта для производной первой формулой.

`lib/second.cpp` — реализация контракта для производной второй формулой.

`lib/leibniz.cpp` — реализация контракта для числа пи рядом Лейбница.

`lib/wallis.cpp` — реализация контракта для числа пи формуллой Валлиса.

`dynamic_library.h` — интерфейс работы с динамическими библиотеками.

`src/dynamic_library.cpp` — реализация `dynamic_library.h` для Linux с использованием функций `dlopen`, `dlsym`, `dlclose`.

`CMakeLists.txt` — файл сборки проекта, описывающий цели для библиотек и программ.

Выводы

В ходе выполнения лабораторной работы были получены практические навыки работы с динамическими библиотеками на языке C++. Были разработаны два контракта (`proizv_contract.h`, `pi_contract.h`) и четыре реализации в виде динамических библиотек, использующие разные алгоритмы при едином интерфейсе.

Программа №1 продемонстрировала использование динамических библиотек с привязкой на этапе линковки, что упрощает запуск, но требует пересборки при смене реализации. Программа №2 реализует явную загрузку библиотек во время выполнения, что позволяет переключать реализацию алгоритмов без перекомпиляции.

Был реализован абстрактный слой `dynamic_library.h`, облегчающий возможную адаптацию программы для других ОС.

Программа демонстрирует:

- использование динамических библиотек при линковке (program1);
- динамическую загрузку и смену реализаций во время выполнения (program2);
- абстракцию системных вызовов для обеспечения кроссплатформенности;
- корректное завершение работы при возникновении ошибок.

В лабораторной работе были реализованы два подхода к использованию динамических библиотек.

Подход 1: линковка на этапе компиляции (program1)

- Привязка к библиотекам выполняется линковщиком во время сборки.
- Меньше накладных расходов при запуске.
- Недостаток: для смены реализации (например, на другие математические формулы) требуется менять настройки сборки и пересобирать программу.

Подход 2: явная загрузка во время выполнения (program2)

- Библиотеки загружаются в момент работы программы.

- Можно переключать реализацию контрактов по команде пользователя без перекомпиляции.
- Небольшие дополнительные накладные расходы на вызовы функций работы с динамическими библиотеками, но подход более гибкий и маштабируемый.
- Необходимо вручную обрабатывать ошибки.

Выводы

В ходе выполнения лабораторной работы были получены практические навыки работы с динамическими библиотеками на языке C++. Были разработаны два контракта (`proizv_contract.h`, `pi_contract.h`) и четыре реализации в виде динамических библиотек , которые используют разные алгоритмы при едином интерфейсе.

Программа №1 продемонстрировала использование динамических библиотек с привязкой на этапе линковки, что упрощает запуск, но требует пересборки при смене реализации. Программа №2 реализует явную загрузку библиотек во время выполнения, что позволяет переключать реализацию алгоритмов без перекомпиляции.

Был реализован абстрактный слой `dynamic_library.cpp`, облегчающий возможную адаптацию программы для других ОС.

Программа демонстрирует:

- использование динамических библиотек при линковке (program1);
- динамическую загрузку и смену реализаций во время выполнения (program2);
- абстракцию системных вызовов для обеспечения кроссплатформенности;
- обработку ошибок;
- корректное завершение работы при возникновении ошибок.

В лабораторной работе были реализованы два подхода к использованию динамических библиотек.

Подход 1: линковка на этапе компиляции (programram1)

- Привязка к библиотекам выполняется линковщиком во время сборки.
- Меньше накладных расходов при запуске.
- Недостаток: для смены реализации требуется менять настройки сборки и пересобирать программу.

Подход 2: явная загрузка во время выполнения (program2)

- Библиотеки загружаются в момент работы программы.
- Можно переключать реализацию контрактов по команде пользователя без перекомпиляции.
- Небольшие дополнительные накладные расходы на вызовы функций работы с динамическими библиотеками, но подход более гибкий и маштабируемый.
- Необходимо вручную обрабатывать ошибки.

Исходная программа

```
#pragma once

#include <string>
#include <memory>
#include <stdexcept>

class DynamicLibrary {
public:
    explicit DynamicLibrary(const std::string& path);
    ~DynamicLibrary();

    DynamicLibrary(const DynamicLibrary&) = delete;
    DynamicLibrary& operator=(const DynamicLibrary&) = delete;

    DynamicLibrary(DynamicLibrary&& other) noexcept;
    DynamicLibrary& operator=(DynamicLibrary&& other) noexcept;

    void* getSymbol(const char* symbolName) const;

    template<typename T>
    T getSymbolAs(const char* symbolName) const {
        return reinterpret_cast<T>(getSymbol(symbolName));
    }

private:
    void* handle_ = nullptr;
};
```

Листинг 1: include/dynamic_library.h

```
#pragma once

extern "C" {
    float Pi(int K);
}
```

Листинг 2: include/pi_contract.h

```
#pragma once

extern "C" {
    float Derivative(float A, float deltaX);
}
```

Листинг 3: include/proizv_contract.h

```
#include "proizv_contract.h"
#include <cmath>

float Derivative(float A, float deltaX) {
    return (std::cos(A + deltaX) - std::cos(A)) / deltaX;
```

```
}
```

Листинг 4: lib/first.cpp

```
#include "proizv_contract.h"
#include <cmath>

float Derivative(float A, float deltaX) {
    return (std::cos(A + deltaX) - std::cos(A - deltaX)) / (2.0f * deltaX);
}
```

Листинг 5: lib/second.cpp

```
#include "pi_contract.h"
#include <cmath>

float Pi(int K) {
    float pi = 0.0f;
    for (int i = 0; i < K; ++i) {
        float term = 1.0f / (2.0f * i + 1.0f);
        if (i % 2 == 1) {
            pi -= term;
        } else {
            pi += term;
        }
    }
    return pi * 4.0f;
}
```

Листинг 6: lib/leibniz.cpp

```
#include "pi_contract.h"

float Pi(int K) {
    double pi = 1.0f;
    for (int i = 1; i <= K; ++i) {
        double num1 = 2.0f * i;
        double den1 = 2.0f * i - 1.0f;
        double den2 = 2.0f * i + 1.0f;
        pi *= (num1 / den1) * (num1 / den2);
    }

    return (float)(pi * 2.0);
}
```

Листинг 7: lib/wallis.cpp

```
#pragma once

#include <string>
#include <memory>
#include <stdexcept>
```

```
6
7  class DynamicLibrary {
8  public:
9      explicit DynamicLibrary(const std::string& path);
10     ~DynamicLibrary();
11
12     DynamicLibrary(const DynamicLibrary&) = delete;
13     DynamicLibrary& operator=(const DynamicLibrary&) = delete;
14
15     DynamicLibrary(DynamicLibrary&& other) noexcept;
16     DynamicLibrary& operator=(DynamicLibrary&& other) noexcept;
17
18     void* getSymbol(const char* symbolName) const;
19
20     template<typename T>
21     T getSymbolAs(const char* symbolName) const {
22         return reinterpret_cast<T>(getSymbol(symbolName));
23     }
24
25 private:
26     void* handle_ = nullptr;
27 };
```

Листинг 8: include/dynamic_library.h

```
1  #include <dlfcn.h>
2  #include <stdexcept>
3  #include <string>
4
5  #include "dynamic_library.h"
6
7  DynamicLibrary::DynamicLibrary(const std::string& path) {
8      handle_ = dlopen(path.c_str(), RTLD_LAZY);
9      if (!handle_) {
10         throw std::runtime_error("Cannot load library: " + std::string(dlerror()));
11     }
12 }
13
14 DynamicLibrary::~DynamicLibrary() {
15     if (handle_) {
16         dlclose(handle_);
17     }
18 }
19
20 DynamicLibrary::DynamicLibrary(DynamicLibrary&& other) noexcept
21     : handle_(other.handle_) {
22     other.handle_ = nullptr;
23 }
24
25 DynamicLibrary& DynamicLibrary::operator=(DynamicLibrary&& other) noexcept {
26     if (this != &other) {
27         if (handle_) {
28             dlclose(handle_);
29         }
30         handle_ = other.handle_;
31         other.handle_ = nullptr;
32     }
```

```
    return *this;
}

void* DynamicLibrary::getSymbol(const char* symbolName) const {
    if (!handle_) {
        throw std::runtime_error("Library is not loaded");
    }
    void* sym = dlsym(handle_, symbolName);
    if (!sym) {
        throw std::runtime_error("Symbol '" + std::string(symbolName) + "' not found: "
            + dlerror());
    }
    return sym;
}
```

Листинг 9: src/dynamic_library.cpp

```
#pragma once
#include <iostream>
#include <string>

#include "proizv_contract.h"
#include "pi_contract.h"

int main() {
    std::cout << "Program 1: First derivative + Leibniz (static linking)" << std::endl
        ;


    std::string cmd;
    while (std::cin >> cmd) {
        if (cmd == "1") {
            float A, deltaX;
            std::cin >> A >> deltaX;
            std::cout << "Derivative: " << Derivative(A, deltaX) << std::endl;
        } else if (cmd == "2") {
            int N;
            std::cin >> N;
            std::cout << "Pi: " << Pi(N) << std::endl;
        }
    }
    return 0;
}
#include <iostream>
#include <vector>
#include <string>
#include <cstdint>
#include <mutex>

namespace BigInt {

constexpr int NUM = 8;

class BigInt {
private:
    uint64_t data_[NUM] = {0};

```

```
public:
    BigInt() = default;
    explicit BigInt(const std::string& hex_str);
    BigInt& operator+=(const BigInt& other);
    BigInt operator+(const BigInt& other) const;
    void add_word(uint64_t word);
    void divide_by_block(uint64_t divisor);
    uint64_t divide_by_10();
    std::string to_dec() const;
    const uint64_t* getData() const {
      return data_;
    }
    static std::mutex sum_mutex;
    friend std::ostream& operator<<(std::ostream& os, const BigInt& big_int);
};
}
```

Листинг 10: program1.cpp

```
#include <iostream>
#include <string>
#include <sstream>
#include <cmath>
#include <iomanip>
#include <memory>
#include <stdexcept>

#include "dynamic_library.h"
#include "proizv_contract.h"
#include "pi_contract.h"

static constexpr auto proizv_symbol = "Derivative";
static constexpr auto pi_symbol = "Pi";

static constexpr auto LIB1_PROIZV = "lib/liblibrary_first.so";
static constexpr auto LIB1_PI = "lib/liblibrary_leibniz.so";
static constexpr auto LIB2_PROIZV = "lib/liblibrary_second.so";
static constexpr auto LIB2_PI = "lib/liblibrary_wallis.so";


using DerivativeFunc = float (*)(float, float);
using PiFunc = float (*)(int);

std::unique_ptr<DynamicLibrary> proizv_lib;
std::unique_ptr<DynamicLibrary> pi_lib;
DerivativeFunc proizv_func = nullptr;
PiFunc pi_func = nullptr;

void load_version(const std::string& proizv_path, const std::string& pi_path) {
    proizv_lib = std::make_unique<DynamicLibrary>(proizv_path);
    pi_lib = std::make_unique<DynamicLibrary>(pi_path);
    proizv_func = proizv_lib->getSymbolAs<DerivativeFunc>(proizv_symbol);
    pi_func = pi_lib->getSymbolAs<PiFunc>(pi_symbol);
}

int main() {
    std::string current_proizv_path = LIB1_PROIZV;
```

```
39      std::string current_pi_path = LIB1_PI;
40      load_version(current_proizv_path, current_pi_path);
41      std::cout << "Initial version first or Leibniz" << std::endl;
42      std::string line;
43      while (std::getline(std::cin, line)) {
44          std::stringstream ss(line);
45          std::string cmd;
46          ss >> cmd;
47          if (cmd == "0") {
48              try {
49                  if (current_proizv_path == LIB1_PROIZV) {
50                      current_proizv_path = LIB2_PROIZV;
51                      current_pi_path = LIB2_PI;
52                      load_version(current_proizv_path, current_pi_path);
53                      std::cout << "--> Switched to second or Wallis" << std::endl;
54                  } else {
55                      current_proizv_path = LIB1_PROIZV;
56                      current_pi_path = LIB1_PI;
57                      load_version(current_proizv_path, current_pi_path);
58                      std::cout << "--> Switched to first or Leibniz" << std::endl;
59                  }
60              } catch (const std::exception& e) {
61                  std::cerr << "Switch failed: " << e.what() << std::endl;
62              }
63          }
64          else if (cmd == "1") {
65              float A, deltaX;
66              if (ss >> A >> deltaX) {
67                  float result = proizv_func(A, deltaX);
68                  std::cout << "Derivative: " << result << std::endl;
69              } else {
70                   std::cout << "Invalid arguments";
71              }
72          }
73          else if (cmd == "2") {
74              int N;
75              if (ss >> N) {
76                  float pi_result = pi_func(N);
77                  std::cout << "Pi: " << pi_result << std::endl;
78              } else {
79                   std::cout << "Invalid arguments" << std::endl;
80              }
81          }
82          else {
83              std::cout << "Unknown command: " << cmd << std::endl;
84          }
85      }
86      return 0;
87  }
```

Листинг 11: program2.cpp

Strace

(Завершаю работу программы Ctrl+c)

```
execve("./program1", ["./program1"], 0x7ffdd516afd0 /* 26 vars */) = 0
```

```
brk(NULL)                                 = 0x650c19e7e000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffcafabade0) = -1 EINVAL (Invalid argument)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
↪  0x7226b58a8000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/glibc-hwcaps/x86-64-v┘
↪  3/liblibrary_first.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or
↪  directory)
newfstatat(AT_FDCWD,
↪  "/home/guuuzova_v/oslab/LabsOS/lab4/lib/glibc-hwcaps/x86-64-v3",
↪  0x7ffcafaba000, 0) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/glibc-hwcaps/x86-64-v┘
↪  2/liblibrary_first.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or
↪  directory)
newfstatat(AT_FDCWD,
↪  "/home/guuuzova_v/oslab/LabsOS/lab4/lib/glibc-hwcaps/x86-64-v2",
↪  0x7ffcafaba000, 0) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/tls/x86_64/x86_64/lib┘
↪  library_first.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or
↪  directory)
newfstatat(AT_FDCWD,
↪  "/home/guuuzova_v/oslab/LabsOS/lab4/lib/tls/x86_64/x86_64", 0x7ffcafaba000,
↪  0) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD,
↪  "/home/guuuzova_v/oslab/LabsOS/lab4/lib/tls/x86_64/liblibrary_first.so",
↪  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
newfstatat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/tls/x86_64",
↪  0x7ffcafaba000, 0) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD,
↪  "/home/guuuzova_v/oslab/LabsOS/lab4/lib/tls/x86_64/liblibrary_first.so",
↪  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
newfstatat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/tls/x86_64",
↪  0x7ffcafaba000, 0) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD,
↪  "/home/guuuzova_v/oslab/LabsOS/lab4/lib/tls/liblibrary_first.so",
↪  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
newfstatat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/tls",
↪  0x7ffcafaba000, 0) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD,
↪  "/home/guuuzova_v/oslab/LabsOS/lab4/lib/x86_64/x86_64/liblibrary_first.so",
↪  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
newfstatat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/x86_64/x86_64",
↪  0x7ffcafaba000, 0) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD,
↪  "/home/guuuzova_v/oslab/LabsOS/lab4/lib/x86_64/liblibrary_first.so",
↪  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
newfstatat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/x86_64",
↪  0x7ffcafaba000, 0) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD,
↪  "/home/guuuzova_v/oslab/LabsOS/lab4/lib/x86_64/liblibrary_first.so",
↪  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
```

```
newfstatat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/x86_64",
↪  0x7ffcafaba000, 0) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/liblibrary_first.so",
↪  O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"...,
↪  832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=15536, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 16440, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7226b58a3000
mmap(0x7226b58a4000, 4096, PROT_READ|PROT_EXEC,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7226b58a4000
mmap(0x7226b58a5000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
↪  0x2000) = 0x7226b58a5000
mmap(0x7226b58a6000, 8192, PROT_READ|PROT_WRITE,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7226b58a6000
close(3)                                = 0
openat(AT_FDCWD,
↪  "/home/guuuzova_v/oslab/LabsOS/lab4/lib/liblibrary_leibniz.so",
↪  O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"...,
↪  832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=15208, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 16424, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7226b589e000
mmap(0x7226b589f000, 4096, PROT_READ|PROT_EXEC,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7226b589f000
mmap(0x7226b58a0000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
↪  0x2000) = 0x7226b58a0000
mmap(0x7226b58a1000, 8192, PROT_READ|PROT_WRITE,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7226b58a1000
close(3)                                = 0
openat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/libstdc++.so.6",
↪  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=29000, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 29000, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7226b5896000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libstdc++.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"...,
↪  832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2260296, ...}, AT_EMPTY_PATH) =
↪  0
mmap(NULL, 2275520, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7226b5600000
mprotect(0x7226b569a000, 1576960, PROT_NONE) = 0
mmap(0x7226b569a000, 1118208, PROT_READ|PROT_EXEC,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x9a000) = 0x7226b569a000
mmap(0x7226b57ab000, 454656, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
↪  0x1ab000) = 0x7226b57ab000
mmap(0x7226b581b000, 57344, PROT_READ|PROT_WRITE,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x21a000) = 0x7226b581b000
mmap(0x7226b5829000, 10432, PROT_READ|PROT_WRITE,
↪  MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7226b5829000
close(3)                                = 0
```

```
openat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/libgcc_s.so.1",
↪  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libgcc_s.so.1", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"...,
↪  832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=125488, ...}, AT_EMPTY_PATH) =
↪  0
mmap(NULL, 127720, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7226b5876000
mmap(0x7226b5879000, 94208, PROT_READ|PROT_EXEC,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x3000) = 0x7226b5879000
mmap(0x7226b5890000, 16384, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
↪  0x1a000) = 0x7226b5890000
mmap(0x7226b5894000, 8192, PROT_READ|PROT_WRITE,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1d000) = 0x7226b5894000
close(3)                                = 0
openat(AT_FDCWD, "/home/guuuzova_v/oslab/LabsOS/lab4/lib/libc.so.6",
↪  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0\0"...,
↪  832) = 832
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"...,
↪  784, 64) = 784
pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"...,
↪  48, 848) = 48
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0O{\f\225\\=\201\327\312\301P\32\230\
↪  266\235"..., 68, 896) = 68
newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2220400, ...}, AT_EMPTY_PATH) =
↪  0
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"...,
↪  784, 64) = 784
mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7226b5200000
mprotect(0x7226b5228000, 2023424, PROT_NONE) = 0
mmap(0x7226b5228000, 1658880, PROT_READ|PROT_EXEC,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7226b5228000
mmap(0x7226b53bd000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
↪  0x1bd000) = 0x7226b53bd000
mmap(0x7226b5416000, 24576, PROT_READ|PROT_WRITE,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x215000) = 0x7226b5416000
mmap(0x7226b541c000, 52816, PROT_READ|PROT_WRITE,
↪  MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7226b541c000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libm.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"...,
↪  832) = 832
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=940560, ...}, AT_EMPTY_PATH) =
↪  0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
↪  0x7226b5874000
mmap(NULL, 942344, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7226b5519000
mmap(0x7226b5527000, 507904, PROT_READ|PROT_EXEC,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe000) = 0x7226b5527000
```

```
mmap(0x7226b55a3000, 372736, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
↪  0x8a000) = 0x7226b55a3000
mmap(0x7226b55fe000, 8192, PROT_READ|PROT_WRITE,
↪  MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe4000) = 0x7226b55fe000
close(3)                                = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
↪  0x7226b5871000
arch_prctl(ARCH_SET_FS, 0x7226b5871740) = 0
set_tid_address(0x7226b5871a10)         = 3949
set_robust_list(0x7226b5871a20, 24)     = 0
rseq(0x7226b58720e0, 0x20, 0, 0x53053053) = 0
mprotect(0x7226b5416000, 16384, PROT_READ) = 0
mprotect(0x7226b55fe000, 4096, PROT_READ) = 0
mprotect(0x7226b5894000, 4096, PROT_READ) = 0
mprotect(0x7226b581b000, 45056, PROT_READ) = 0
mprotect(0x7226b58a1000, 4096, PROT_READ) = 0
mprotect(0x7226b58a6000, 4096, PROT_READ) = 0
mprotect(0x650be98e3000, 4096, PROT_READ) = 0
mprotect(0x7226b58e2000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024,
↪  rlim_max=RLIM64_INFINITY}) = 0
munmap(0x7226b5896000, 29000)           = 0
getrandom("\x04\x85\x9f\x30\x1a\x37\x89\x12", 8, GRND_NONBLOCK) = 8
brk(NULL)                               = 0x650c19e7e000
brk(0x650c19e9f000)                     = 0x650c19e9f000
futex(0x7226b582977c, FUTEX_WAKE_PRIVATE, 2147483647) = 0
newfstatat(1, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x2), ...},
↪  AT_EMPTY_PATH) = 0
write(1, "Program 1: First derivative + Le"..., 55) = 55
newfstatat(0, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x2), ...},
↪  AT_EMPTY_PATH) = 0
read(0, "2 1000000000\n", 1024)         = 13
write(1, "Pi: 3.1416\n", 11)            = 11
read(0, 0x650c19e902c0, 1024)           = ? ERESTARTSYS (To be restarted if
↪  SA_RESTART is set)
--- SIGINT {si_signo=SIGINT, si_code=SI_KERNEL} ---
+++ killed by SIGINT +++
```