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: Ю. В. Павлова
Группа: М8О-207БВ-24
Преподаватель: Е. С. Миронов

Москва, 2025

Условие

Цель работы:

- Приобретение практических навыков в:
  - Создании динамических библиотек;
  - Создании программ, использующих функции динамических библиотек.

Задание:

Требуется создать динамические библиотеки, которые реализуют заданный вариантом функционал. Далее использовать данные библиотеки двумя 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, заданные вариантом;
- Тестовая программа (программа №1), которая использует одну из библиотек, используя информацию, полученную 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 обеих 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). Можно реализовать лабораторную работу без данной функции, но максимальная оценка в этом случае будет «хорошо»;
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: 24

№	Описание	Сигнатура	Реализация 1	Реализация 2
3	Вычисление наибольшего общего делителя (НОД) для двух натуральных чисел	`int GCF(int A, int B)`	Алгоритм Евклида	Наивный алгоритм: перебор всех возможных делителей от 1 до $\min(A, B)$
7	Вычисление площади фигуры по двум параметрам	`float Square(float A, float B)`	Площадь прямоугольника: $A \cdot B$	Площадь прямоугольного треугольника: $0.5 \cdot A \cdot B$

Метод решения

Алгоритм решения задачи:

1. Пользователь вводит команду в консоль одной из тестовых программ:
   - `0` — выход из программы (в обеих версиях);
   - `3` — переключение реализаций (только для программы №2, `prog_dynamic`);
   - `1 A B` — вызов функции `GCF(A, B)`;
   - `2 A B` — вызов функции `Square(A, B)`.
2. Для программы `prog_static`:
   - Связывание с библиотеками `libgcf_euclid.so` и `libsquare_rect.so` происходит на этапе линковки;
   - При запуске ОС автоматически загружает указанные `.so`-файлы;
   - Вызовы функций разрешаются до начала выполнения основной логики.
3. Для программы `prog_dynamic`:
   - На старте загружаются библиотеки `./libgcf_euclid.so` и `./libsquare_rect.so` с помощью `dlopen`;
   - Через `dlsym` получаются указатели на функции `GCF` и `Square`;
   - При вводе команды `3` текущие библиотеки выгружаются (`dlclose`), загружаются альтернативные (`libgcf_naive.so`, `libsquare_triangle.so`), обновляются указатели на функции;
   - Результат выводится в консоль.
4. Все функции экспортируются с `extern "C"`, чтобы избежать name mangling и обеспечить корректную работу `dlsym`.

5. Память в библиотеках не выделяется динамически, так как обе функции возвращают простые типы (`int`, `float`), поэтому освобождение памяти не требуется.

6. Обработка ошибок:

   - При ошибке `dlopen` или `dlsym` выводится сообщение и программа завершается с кодом `EXIT_FAILURE`;
   - Все ресурсы (дескрипторы библиотек) освобождаются вручную через `dlclose` при завершении.

Архитектура программы

```
lab4-var24/
 CMakeLists.txt
 include/
    functions.h
 src/
    gcf_euclid.cpp
    gcf_naive.cpp
    square_rect.cpp
    square_triangle.cpp
 main_static.cpp
 main_dynamic.cpp
```

Описание программы

`functions.h` — объявляет функции с `extern "C"` для экспорта без name mangling.

`gcf_euclid.cpp` — реализует НОД через рекурсивный алгоритм Евклида.

`gcf_naive.cpp` — реализует НОД перебором делителей от 1 до $\min(A, B)$.

`square_rect.cpp` — возвращает площадь прямоугольника: $A \cdot B$.

`square_triangle.cpp` — возвращает площадь прямоугольного треугольника: $0.5 \cdot A \cdot B$.

`main_static.cpp` — программа №1, использует статическую линковку, поддерживает команды `0`, `1`, `2`.

`main_dynamic.cpp` — программа №2, загружает библиотеки вручную, поддерживает команды `0` (выход), `1`, `2`, `3` (переключение реализации).

Все программы скомпилированы с флагом `C++17`, используют `dlopen/dlsym/dlclose` из `<dlfcn.h>`.

Результаты

Программа успешно создаёт четыре динамические библиотеки:

- `libgcf_euclid.so`
- `libgcf_naive.so`
- `libsquare_rect.so`
- `libsquare_triangle.so`

Тестовая программа `prog_static` корректно использует библиотеки, подключённые на этапе линковки.

Программа `prog_dynamic` загружает библиотеки во время выполнения, поддерживает переключение реализаций по команде `3` и корректно вызывает функции через указатели.

Пример сессии:

```
Lab 4: Dynamic Loading
0 - Exit
1 <A> <B> - GCF
2 <A> <B> - Area
3 - Switch implementation
1 48 18
GCF Result: 6
3
Switched to: Naive & Triangle
2 4 5
Square Result: 10
0
Exiting...
```

Все команды обрабатываются корректно. Ошибки загрузки библиотек обрабатываются с выводом диагностики.

Выводы

В ходе выполнения лабораторной работы были реализованы:

- Четыре динамические библиотеки с двумя парами реализаций заданных функций.
- Две тестовые программы, демонстрирующие разные способы подключения библиотек: статический (implicit linking) и динамический (explicit linking).
- Механизм переключения реализаций во время выполнения, что подчеркивает гибкость динамической загрузки.

Работа показала, что динамические библиотеки позволяют:

- Снижать размер исполняемых файлов за счёт разделения кода.

- Обновлять функционал без перекомпиляции основной программы.
- Реализовывать гибкие архитектуры, в которых поведение определяется конфигурацией на этапе запуска.

Исходный код

```h
#pragma once

extern "C" {
    int GCF(int A, int B);
    float Square(float A, float B);
}
```

Листинг 1: include/functions.h

```cpp
#include "functions.h"

extern "C" {
    int GCF(int A, int B) {
        if (B == 0)
            return A;
        return GCF(B, A % B);
    }
}
```

Листинг 2: src/gcf_euclid.cpp

```cpp
#include "functions.h"
#include <algorithm>

extern "C" {
    int GCF(int A, int B) {
        int gcd = 1;
        int limit = std::min(A, B);
        for (int i = 1; i <= limit; ++i) {
            if (A % i == 0 && B % i == 0)
                gcd = i;
        }
        return gcd;
    }
}
```

Листинг 3: src/gcf_naive.cpp

```cpp
#include "functions.h"
```

```
2
3 extern "C" {
4     float Square(float A, float B) {
5         return A * B;
6     }
7 }
```

Листинг 4: src/square$_r$*ect.cpp*

```
1 #include "functions.h"
2
3 extern "C" {
4     float Square(float A, float B) {
5         return 0.5f * A * B;
6     }
7 }
```

Листинг 5: src/square$_t$*riangle.cpp*

```
1  #include <iostream>
2  #include "functions.h"
3
4  int main() {
5      int cmd;
6      std::cout << "Lab 4: Static Linking (Euclid + Rectangle)\n";
7      std::cout << "0 - Exit\n";
8      std::cout << "1 <A> <B> - GCF\n";
9      std::cout << "2 <A> <B> - Area\n";
10
11     while (std::cin >> cmd) {
12         if (cmd == 0) {
13             std::cout << "Exiting...\n";
14             break;
15         } else if (cmd == 1) {
16             int a, b;
17             std::cin >> a >> b;
18             std::cout << "GCF(" << a << ", " << b << ") = " << GCF(a,
                   ↪  b) << "\n";
19         } else if (cmd == 2) {
20             float a, b;
21             std::cin >> a >> b;
22             std::cout << "Area: " << Square(a, b) << "\n";
23         } else {
24             std::cout << "Unknown command.\n";
25         }
```

```
    }
    return 0;
}
```

Листинг 6: main_static.cpp

```
#include <iostream>
#include <dlfcn.h>
#include "functions.h"

using GCF_Func = int (*)(int, int);
using Square_Func = float (*)(float, float);

const char* LIB_EUCLID = "./libgcf_euclid.so";
const char* LIB_NAIVE = "./libgcf_naive.so";
const char* LIB_RECT = "./libsquare_rect.so";
const char* LIB_TRIANGLE = "./libsquare_triangle.so";

int main() {
    void* handle_gcf = nullptr;
    void* handle_square = nullptr;

    GCF_Func CalcGCF = nullptr;
    Square_Func CalcSquare = nullptr;

    int mode = 0;

    auto load_libs = [&](int current_mode) {
        if (handle_gcf) dlclose(handle_gcf);
        if (handle_square) dlclose(handle_square);

        const char* path_gcf = (current_mode == 0) ? LIB_EUCLID :
            ↪ LIB_NAIVE;
        const char* path_sq = (current_mode == 0) ? LIB_RECT :
            ↪ LIB_TRIANGLE;

        handle_gcf = dlopen(path_gcf, RTLD_LAZY);
        handle_square = dlopen(path_sq, RTLD_LAZY);

        if (!handle_gcf || !handle_square) {
            std::cerr << "Error loading libs: " << dlerror() << "\n";
            exit(EXIT_FAILURE);
        }

```

```
        CalcGCF = (GCF_Func)dlsym(handle_gcf, "GCF");
        CalcSquare = (Square_Func)dlsym(handle_square, "Square");

        if (!CalcGCF || !CalcSquare) {
            std::cerr << "Error loading symbols: " << dlerror() << "\
                ↪ n";
            exit(EXIT_FAILURE);
        }

        std::cout << "Switched to: "
                  << ((current_mode == 0) ? "Euclid & Rectangle" : "
                     ↪ Naive & Triangle")
                  << "\n";
    };

    load_libs(mode);

    int cmd;
    std::cout << "Lab 4: Dynamic Loading\n";
    std::cout << "0 - Exit\n";
    std::cout << "1 <A> <B> - GCF\n";
    std::cout << "2 <A> <B> - Area\n";
    std::cout << "3 - Switch implementation\n";

    while (std::cin >> cmd) {
        if (cmd == 0) {
            std::cout << "Exiting...\n";
            break;
        } else if (cmd == 3) {
            mode = !mode;
            load_libs(mode);
        } else if (cmd == 1) {
            int a, b;
            std::cin >> a >> b;
            std::cout << "GCF Result: " << CalcGCF(a, b) << "\n";
        } else if (cmd == 2) {
            float a, b;
            std::cin >> a >> b;
            std::cout << "Square Result: " << CalcSquare(a, b) << "\n
                ↪ ";
        } else {
            std::cout << "Unknown command\n";
        }
```

```
    }

    if (handle_gcf) dlclose(handle_gcf);
    if (handle_square) dlclose(handle_square);

    return 0;
}
```

Листинг 7: main_dynamic.cpp

strace

Листинг 1: strace вывод программы

```
execve("./prog_dynamic", ["./prog_dynamic"], 0x7ffc7eceb5f0 /* 36
    ↪ vars */) = 0
brk(NULL)                                       = 0x557cdf291000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
    ↪  0) = 0x7f741857e000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or
    ↪ directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=21863, ...}) = 0
mmap(NULL, 21863, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f7418578000
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
fstat(3, {st_mode=S_IFREG|0644, st_size=2592224, ...}) = 0
mmap(NULL, 2609472, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0
    ↪ x7f74182fa000
mmap(0x7f7418397000, 1343488, PROT_READ|PROT_EXEC, MAP_PRIVATE|
    ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0x9d000) = 0x7f7418397000
mmap(0x7f74184df000, 552960, PROT_READ, MAP_PRIVATE|MAP_FIXED|
    ↪ MAP_DENYWRITE, 3, 0x1e5000) = 0x7f74184df000
mmap(0x7f7418566000, 57344, PROT_READ|PROT_WRITE, MAP_PRIVATE|
    ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0x26b000) = 0x7f7418566000
mmap(0x7f7418574000, 12608, PROT_READ|PROT_WRITE, MAP_PRIVATE|
    ↪ MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f7418574000
close(3)                                        = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|
    ↪ O_CLOEXEC) = 3
```

```
19 read(3, "\177ELF
   ↪ \2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\243\2\0\0\0\0\0"
   ↪ ..., 832) = 832
20 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
21 fstat(3, {st_mode=S_IFREG|0755, st_size=2125328, ...}) = 0
22 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
23 mmap(NULL, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0
   ↪ x7f74180e8000
24 mmap(0x7f7418110000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|
   ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7f7418110000
25 mmap(0x7f7418298000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|
   ↪ MAP_DENYWRITE, 3, 0x1b0000) = 0x7f7418298000
26 mmap(0x7f74182e7000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|
   ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000) = 0x7f74182e7000
27 mmap(0x7f74182ed000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|
   ↪ MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f74182ed000
28 close(3)                                      = 0
29 openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libm.so.6", O_RDONLY|
   ↪ O_CLOEXEC) = 3
30 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"...,
   ↪ 832) = 832
31 fstat(3, {st_mode=S_IFREG|0644, st_size=952616, ...}) = 0
32 mmap(NULL, 950296, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0
   ↪ x7f7417fff000
33 mmap(0x7f741800f000, 520192, PROT_READ|PROT_EXEC, MAP_PRIVATE|
   ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0x10000) = 0x7f741800f000
34 mmap(0x7f741808e000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|
   ↪ MAP_DENYWRITE, 3, 0x8f000) = 0x7f741808e000
35 mmap(0x7f74180e6000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
   ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0xe7000) = 0x7f74180e6000
36 close(3)                                      = 0
37 openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libgcc_s.so.1", O_RDONLY|
   ↪ O_CLOEXEC) = 3
38 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"...,
   ↪ 832) = 832
39 fstat(3, {st_mode=S_IFREG|0644, st_size=183024, ...}) = 0
40 mmap(NULL, 185256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0
   ↪ x7f7417fd1000
```

```
mmap(0x7f7417fd5000, 147456, PROT_READ|PROT_EXEC, MAP_PRIVATE|
    ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0x4000) = 0x7f7417fd5000
mmap(0x7f7417ff9000, 16384, PROT_READ, MAP_PRIVATE|MAP_FIXED|
    ↪ MAP_DENYWRITE, 3, 0x28000) = 0x7f7417ff9000
mmap(0x7f7417ffd000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
    ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0x2b000) = 0x7f7417ffd000
close(3)                                = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
    ↪  0) = 0x7f7417fcf000
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS,
    ↪ -1, 0) = 0x7f7417fcc000
arch_prctl(ARCH_SET_FS, 0x7f7417fcc740) = 0
set_tid_address(0x7f7417fcca10)         = 10005
set_robust_list(0x7f7417fcca20, 24)     = 0
rseq(0x7f7417fcd060, 0x20, 0, 0x53053053) = 0
mprotect(0x7f74182e7000, 16384, PROT_READ) = 0
mprotect(0x7f7417ffd000, 4096, PROT_READ) = 0
mprotect(0x7f74180e6000, 4096, PROT_READ) = 0
mprotect(0x7f7418566000, 45056, PROT_READ) = 0
mprotect(0x557ccc051000, 4096, PROT_READ) = 0
mprotect(0x7f74185b6000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=
    ↪ RLIM64_INFINITY}) = 0
munmap(0x7f7418578000, 21863)           = 0
futex(0x7f74185747bc, FUTEX_WAKE_PRIVATE, 2147483647) = 0
getrandom("\x2f\x33\x8e\x8b\xcd\xb8\x9d\x5c", 8, GRND_NONBLOCK) = 8
brk(NULL)                               = 0x557cdf291000
brk(0x557cdf2b2000)                     = 0x557cdf2b2000
openat(AT_FDCWD, "./libgcf_euclid.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
fstat(3, {st_mode=S_IFREG|0755, st_size=15672, ...}) = 0
getcwd("/home/pavlovau/lab4_os/build", 128) = 29
mmap(NULL, 16400, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0
    ↪ x7f7418579000
mmap(0x7f741857a000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED
    ↪ |MAP_DENYWRITE, 3, 0x1000) = 0x7f741857a000
mmap(0x7f741857b000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|
    ↪ MAP_DENYWRITE, 3, 0x2000) = 0x7f741857b000
mmap(0x7f741857c000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
    ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f741857c000
close(3)                                = 0
```

```
mprotect(0x7f741857c000, 4096, PROT_READ) = 0
openat(AT_FDCWD, "./libsquare_rect.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
fstat(3, {st_mode=S_IFREG|0755, st_size=15640, ...}) = 0
getcwd("/home/pavlovau/lab4_os/build", 128) = 29
mmap(NULL, 16400, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0
    ↪ x7f7417fc7000
mmap(0x7f7417fc8000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED
    ↪ |MAP_DENYWRITE, 3, 0x1000) = 0x7f7417fc8000
mmap(0x7f7417fc9000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|
    ↪ MAP_DENYWRITE, 3, 0x2000) = 0x7f7417fc9000
mmap(0x7f7417fca000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
    ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f7417fca000
close(3)                                = 0
mprotect(0x7f7417fca000, 4096, PROT_READ) = 0
fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x5), ...}) = 0
write(1, "Switched to: Euclid & Rectangle\n", 32) = 32
write(1, "Lab 4: Dynamic Loading\n", 23) = 23
write(1, "0 - Exit\n", 9)               = 9
write(1, "1 <A> <B> - Compute GCF\n", 24) = 24
write(1, "2 <A> <B> - Compute Area\n", 25) = 25
write(1, "3 - Switch implementation\n", 26) = 26
fstat(0, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x5), ...}) = 0
read(0, "1\n", 1024)                    = 2
read(0, "12 14\n", 1024)                = 6
write(1, "GCF Result: 2\n", 14)         = 14
read(0, "2\n", 1024)                    = 2
read(0, "12 13\n", 1024)                = 6
write(1, "Square Result: 156\n", 19)    = 19
read(0, "3\n", 1024)                    = 2
munmap(0x7f7418579000, 16400)           = 0
munmap(0x7f7417fc7000, 16400)           = 0
openat(AT_FDCWD, "./libgcf_naive.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
fstat(3, {st_mode=S_IFREG|0755, st_size=15832, ...}) = 0
getcwd("/home/pavlovau/lab4_os/build", 128) = 29
mmap(NULL, 16408, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0
    ↪ x7f7418579000
```

```
mmap(0x7f741857a000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED
    ↪ |MAP_DENYWRITE, 3, 0x1000) = 0x7f741857a000
mmap(0x7f741857b000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|
    ↪ MAP_DENYWRITE, 3, 0x2000) = 0x7f741857b000
mmap(0x7f741857c000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
    ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f741857c000
close(3)                                = 0
mprotect(0x7f741857c000, 4096, PROT_READ) = 0
openat(AT_FDCWD, "./libsquare_triangle.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
fstat(3, {st_mode=S_IFREG|0755, st_size=15640, ...}) = 0
getcwd("/home/pavlovau/lab4_os/build", 128) = 29
mmap(NULL, 16400, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0
    ↪ x7f7417fc7000
mmap(0x7f7417fc8000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED
    ↪ |MAP_DENYWRITE, 3, 0x1000) = 0x7f7417fc8000
mmap(0x7f7417fc9000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|
    ↪ MAP_DENYWRITE, 3, 0x2000) = 0x7f7417fc9000
mmap(0x7f7417fca000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
    ↪ MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f7417fca000
close(3)                                = 0
mprotect(0x7f7417fca000, 4096, PROT_READ) = 0
write(1, "Switched to: Naive & Triangle\n", 30) = 30
read(0, "1\n", 1024)                    = 2
read(0, "12 31\n", 1024)                = 6
write(1, "GCF Result: 1\n", 14)         = 14
read(0, "2\n", 1024)                    = 2
read(0, "13 12\n", 1024)                = 6
write(1, "Square Result: 78\n", 18)     = 18
read(0, "0\n", 1024)                    = 2
write(1, "Exiting...\n", 11)            = 11
munmap(0x7f7418579000, 16408)           = 0
munmap(0x7f7417fc7000, 16400)           = 0
lseek(0, -1, SEEK_CUR)                  = -1 ESPIPE (Illegal seek)
exit_group(0)                           = ?
+++ exited with 0 +++
```