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-209Б-24

Студент: Артонкин В.Н.

Преподаватель: Миронов Е.С.

Оценка: _____

Дата: 14.12.25

Москва, 2025

Постановка задачи

Вариант 35.

Необходимо разработать программу, демонстрирующую:

- Статическую загрузку библиотеки
- Динамическую загрузку библиотеки во время выполнения программы с использованием dlopen,dlsym, dlclose;
- корректную работу двух альтернативных реализаций функций float Square(float A, float B) и int* Sort(int* x, int n), размещённых в библиотеках libimpl1.so и libimpl2.so;
- возможность выбора реализаций библиотек (для программы с динамической загрузкой библиотек)

Общий метод и алгоритм решения

Использованные системные вызовы:

- **void *dlopen(const char *filename, int flags) -** открывает динамическую библиотеку
- **void *dlsym(void *handle, const char *symbol) -** получает адрес символа из библиотеки
- **int dlclose(void *handle) -** закрывает загруженную библиотеку
- **char *dlerror(void) -** возвращает сообщение об ошибке динамической загрузки
- **void *malloc(size_t size)** - выделяет блок памяти заданного размера
- **void free(void *ptr)** - освобождает ранее выделенную память

Сначала были описаны 2 реализации библиотек с общим заголовочным файлом. После написан способ взаимодействия с программой через консоль, общий как для статической, так и для динамической версии. После был добавлен блок, отвечающий за подключение динамических библиотек. Затем компиляция библиотек и исполняемых файлов

Код программы

calc.h

```
#ifndef CALC_H

#define CALC_H



int* Sort(int *array, int size);



float Square(float A, float B);



#endif
```

impl1.c

```
#include <stdio.h>

#include "calc.h"



// Прямоугольник

float Square(float A, float B){

    return A * B;

}



// Пузырьковая сортировка

int* Sort(int* array, int size){

    for (int i = 0; i < size - 1; i++) {

        for (int j = 0; j < size - i - 1; j++) {

            if (array[j] > array[j + 1]) {

                int temp = array[j];

                array[j] = array[j + 1];

                array[j + 1] = temp;

            }

        }

    }

    return array;

}
```

impl2.c

```c
#include <stdio.h>
#include "calc.h"


// Прямоугольный треугольник
float Square(float A, float B){
    return (A * B) / 2;
}


// Сортировка Хоара
void QuickSort(int* array, int low, int high) {
    if (low >= high) return;

    int pivot = array[(low + high) / 2];
    int i = low, j = high;

    while (i <= j) {
        while (array[i] < pivot) i++;
        while (array[j] > pivot) j--;
        if (i <= j) {
            int temp = array[i];
            array[i] = array[j];
            array[j] = temp;
            i++;
            j--;
        }
    }

    QuickSort(array, low, j);
    QuickSort(array, i, high);
```

```
}


int* Sort(int* array, int size){
    QuickSort(array, 0, size - 1);
    return array;
}
```

prog_static.c

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


int main(void) {
    char cmd;


    while (1) {
        if (scanf(" %c", &cmd) != 1) {
            break;
        }


        if (cmd == 'q' || cmd == 'Q') {
            break;
        } else if (cmd == '1') {
            float A, B;
            if (scanf("%f %f", &A, &B) != 2) {
                return 1;
            }
            float g = Square(A, B);
            printf("%f\n", g);
```

```
    } else if (cmd == '2') {
        int size;
        if (scanf("%d", &size) != 1 || size <= 0) {
            return 1;
        }

        int* array = (int*)malloc(size * sizeof(int));
        if (array == NULL) {
            return 1;
        }

        for (int i = 0; i < size; i++) {
            if (scanf("%d", &array[i]) != 1) {
                free(array);
                return 1;
            }
        }

        int* sorted = Sort(array, size);

        for (int i = 0; i < size; i++) {
            printf("%d", sorted[i]);
            if (i < size - 1) {
                printf(" ");
            }
        }
        printf("\n");

        free(array);
```

```
        } else {
        }
    }
    return 0;
}
```

prog_dynamic.c

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

typedef float (*Square_fn_t)(float, float);
typedef int* (*Sort_fn_t)(int*, int);

struct Impl {
    void    *handle;
    Square_fn_t Square;
    Sort_fn_t   Sort;
    const char *path;
};

int main(void) {
    struct Impl impls[2] = {
        {NULL, NULL, NULL, "./libimpl1.so"},
        {NULL, NULL, NULL, "./libimpl2.so"}
    };

    for (int i = 0; i < 2; ++i) {
```

```
        impls[i].handle = dlopen(impls[i].path, RTLD_LAZY);
        if (!impls[i].handle) {
            fprintf(stderr, "dlopen(%s) failed: %s\n",
                    impls[i].path, dlerror());
            for (int j = 0; j < i; ++j) {
                dlclose(impls[j].handle);
            }
            return 1;
        }

        dlerror();

        impls[i].Square = (Square_fn_t)dlsym(impls[i].handle, "Square");
        const char *err = dlerror();
        if (err) {
            fprintf(stderr, "dlsym(Square) in %s failed: %s\n",
                    impls[i].path, err);
            for (int j = 0; j <= i; ++j) {
                dlclose(impls[j].handle);
            }
            return 1;
        }

        impls[i].Sort = (Sort_fn_t)dlsym(impls[i].handle, "Sort");
        err = dlerror();
        if (err) {
            fprintf(stderr, "dlsym(Sort) in %s failed: %s\n",
                    impls[i].path, err);
            for (int j = 0; j <= i; ++j) {
```

```
            dlclose(impls[j].handle);
        }
        return 1;
    }
}

int current = 0;
char cmd;

while (1) {
if (scanf(" %c", &cmd) != 1) {
    break;
}

if (cmd == 'q' || cmd == 'Q') {
    break;
} else if (cmd == '0') {
    current = 1 - current;
} else if (cmd == '1') {
    float A, B;
    if (scanf("%f %f", &A, &B) != 2) {
        break;
    }
    float g = impls[current].Square(A, B);
    printf("%f\n", g);
} else if (cmd == '2') {
    int size;
    if (scanf("%d", &size) != 1 || size <= 0) {
        break;
```

```
        }

        int* array = (int*)malloc(size * sizeof(int));
        if (array == NULL) {
            break;
        }

        for (int i = 0; i < size; i++) {
            if (scanf("%d", &array[i]) != 1) {
                free(array);
                break;
            }
        }

        int* sorted = impls[current].Sort(array, size);

        for (int i = 0; i < size; i++) {
            printf("%d", sorted[i]);
            if (i < size - 1) {
                printf(" ");
            }
        }
        printf("\n");

        free(array);
    }
}

    for (int i = 0; i < 2; ++i) {
```

```
        if (impls[i].handle) {

            dlclose(impls[i].handle);

        }

    }



    return 0;

}
```

Протокол работы программы

```
vscode ➜ /workspaces/OS/lab4/src (main) $ strace -tt -T -v -s 100 -o trace_static.txt ./prog_static
1 2 3
6.000000
2 3 5 6 3
3 5 6
q
vscode ➜ /workspaces/OS/lab4/src (main) $ LD_LIBRARY_PATH=. strace -ff -tt -T -s 200 -o trace_dynamic ./prog_dynamic
1 2 3
6.000000
2 3 5 6 3
3 5 6
0
1 2 3
3.000000
2 3 5 6 3
3 5 6
q
vscode ➜ /workspaces/OS/lab4/src (main) $
```

prog_static

17:26:57.632698 execve("./prog_static", ["./prog_static"], ["SHELL=/bin/bash",
"COLORTERM=truecolor", "TERM_PROGRAM_VERSION=1.107.0",
"HOSTNAME=1044750a6712",
"REMOTE_CONTAINERS_IPC=/tmp/vscode-remote-containers-ipc-50516a1e-05b0-4b1f-86fe-cf69a6c
69dcf.sock", "PWD=/workspaces/OS/lab4/src",
"VSCODE_GIT_ASKPASS_NODE=/vscode/vscode-server/bin/linux-x64/618725e67565b290ba4da6fe2
d29f8fa1d4e3622"..., "HOME=/home/vscode", "LANG=en_US.UTF-8",
"LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01
:or=40;31;0"..., "REMOTE_CONTAINERS=true",
"WAYLAND_DISPLAY=vscode-wayland-dcec0b54-2c91-4eff-9348-f39cd119df95.sock",
"GIT_ASKPASS=/vscode/vscode-server/bin/linux-x64/618725e67565b290ba4da6fe2d29f8fa1d4e3622/e
xtensions/"..., "VSCODE_GIT_ASKPASS_EXTRA_ARGS=",
"VSCODE_PYTHON_AUTOACTIVATE_GUARD=1", "LESSCLOSE=/usr/bin/lesspipe %s %s",
"TERM=xterm-256color",
"REMOTE_CONTAINERS_SOCKETS=[\"/tmp/.X11-unix/X1\",\"/home/vscode/.gnupg/S.gpg-agent\"]"
, "LESSOPEN=| /usr/bin/lesspipe %s", "USER=vscode",
"VSCODE_GIT_IPC_HANDLE=/tmp/user/1000/vscode-git-86ac5bc9b1.sock", "DISPLAY=:1",
"SHLVL=2", "GIT_EDITOR=code --wait", "PROMPT_DIRTRIM=4",
"XDG_RUNTIME_DIR=/tmp/user/1000",
"VSCODE_GIT_ASKPASS_MAIN=/vscode/vscode-server/bin/linux-x64/618725e67565b290ba4da6fe2
d29f8fa1d4e3622"...,
"BROWSER=/vscode/vscode-server/bin/linux-x64/618725e67565b290ba4da6fe2d29f8fa1d4e3622/bin/h
elpers/bro"...,
"PATH=/home/vscode/.vscode-server/data/User/globalStorage/github.copilot-chat/copilotCli:/home/vsco
de"..., "REMOTE_CONTAINERS_DISPLAY_SOCK=/tmp/.X11-unix/X1",
"OLDPWD=/workspaces/OS/lab4", "TERM_PROGRAM=vscode",
"VSCODE_IPC_HOOK_CLI=/tmp/vscode-ipc-35da8e30-cac2-4f3e-978f-aec7c27f7e1c.sock",
"_=/usr/bin/strace"]) = 0 <0.004541>

17:26:57.639242 brk(NULL)   = 0x5616f1ea3000 <0.000064>

17:26:57.640049 mmap(NULL, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f80fbbb3000 <0.000080>

17:26:57.640921 access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
<0.000100>

17:26:57.642099 openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3 <0.000137>

**17:26:57.643145 fstat(3, {st_dev=makedev(0, 0x91), st_ino=250030, st_mode=S_IFREG|0644,
st_nlink=1, st_uid=0, st_gid=0, st_blksize=4096, st_blocks=24, st_size=11939, st_atime=1760611512
/* 2025-10-16T10:45:12+0000 */, st_atime_nsec=0, st_mtime=1760611512 /*
2025-10-16T10:45:12+0000 */, st_mtime_nsec=0, st_ctime=1763922925 /*
2025-11-23T18:35:25.719300012+0000 */, st_ctime_nsec=719300012}) = 0 <0.000062>**

**17:26:57.644108 mmap(NULL, 11939, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f80fbbb0000
<0.000076>**

17:26:57.644823 close(3)   = 0 <0.000066>

17:26:57.645601 openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3 <0.000099>

17:26:57.646480 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@\0\0\0\0\0\0\0\20^\0\0\0\0\0\0\0\0@\08\0\16\0@\0@\0?\0\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\20\3\0\0"..., 832) = 832 <0.000468>

17:26:57.647865 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\20\3\0\0\0\0\0\0\20\3\0\0\0\0\0\0\10\0\0\0\0\0\0\0\3\0\0\0\4\0\0\0\240g\35\0\0\0\0\0\240g\35\0\0\0\0\0\240g\35\0\0\0\0\0\34\0\0\0\0\0\0\0\34\0\0\0"..., 784, 64) = 784 <0.000066>

17:26:57.648695 fstat(3, {st_dev=makedev(0, 0x91), st_ino=236312, st_mode=S_IFREG|0755, st_nlink=1, st_uid=0, st_gid=0, st_blksize=4096, st_blocks=4152, st_size=2125328, st_atime=1758120942 /* 2025-09-17T14:55:42+0000 */, st_atime_nsec=0, st_mtime=1758120942 /* 2025-09-17T14:55:42+0000 */, st_mtime_nsec=0, st_ctime=1763922845 /* 2025-11-23T18:34:05.458151251+0000 */, st_ctime_nsec=458151251}) = 0 <0.000067>

17:26:57.649481 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\20\3\0\0\0\0\0\0\20\3\0\0\0\0\0\0\10\0\0\0\0\0\0\0\3\0\0\0\4\0\0\0\240g\35\0\0\0\0\0\240g\35\0\0\0\0\0\240g\35\0\0\0\0\0\34\0\0\0\0\0\0\0\34\0\0\0"..., 784, 64) = 784 <0.000058>

17:26:57.650140 mmap(NULL, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f80fb99e000 <0.000066>

17:26:57.650811 mmap(0x7f80fb9c6000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7f80fb9c6000 <0.000104>

17:26:57.651562 mmap(0x7f80fbb4e000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b0000) = 0x7f80fbb4e000 <0.000117>

17:26:57.652418 mmap(0x7f80fbb9d000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000) = 0x7f80fbb9d000 <0.000098>

17:26:57.653266 mmap(0x7f80fbba3000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f80fbba3000 <0.000095>

17:26:57.654248 close(3) = 0 <0.000090>

17:26:57.655217 mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f80fb99b000 <0.000071>

17:26:57.655997 arch_prctl(ARCH_SET_FS, 0x7f80fb99b740) = 0 <0.000069>

17:26:57.656986 set_tid_address(0x7f80fb99ba10) = 31769 <0.000319>

17:26:57.658178 set_robust_list(0x7f80fb99ba20, 24) = 0 <0.000138>

17:26:57.659391 rseq(0x7f80fb99c060, 0x20, 0, 0x53053053) = 0 <0.000075>

17:26:57.660797 mprotect(0x7f80fbb9d000, 16384, PROT_READ) = 0 <0.000075>

17:26:57.661515 mprotect(0x5616c6120000, 4096, PROT_READ) = 0 <0.000077>

17:26:57.662326 mprotect(0x7f80fbbeb000, 8192, PROT_READ) = 0 <0.000111>

17:26:57.663223 prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0 <0.000097>

17:26:57.664100 munmap(0x7f80fbbb0000, 11939) = 0 <0.000126>

17:26:57.665111 fstat(0, {st_dev=makedev(0, 0x9c), st_ino=4, st_mode=S_IFCHR|0620, st_nlink=1, st_uid=1000, st_gid=5, st_blksize=1024, st_blocks=0, st_rdev=makedev(0x88, 0x1), st_atime=1765733217 /* 2025-12-14T17:26:57.534358669+0000 */, st_atime_nsec=534358669, st_mtime=1765733217 /* 2025-12-14T17:26:57.534358669+0000 */, st_mtime_nsec=534358669, st_ctime=1765718984 /* 2025-12-14T13:29:44.534358669+0000 */, st_ctime_nsec=534358669}) = 0 <0.000081>

17:26:57.665979 getrandom("\x77\x4a\x8b\xd7\x34\x94\xd8\x9a", 8, GRND_NONBLOCK) = 8 <0.000101>

17:26:57.666876 brk(NULL)   = 0x5616f1ea3000 <0.000062>

17:26:57.667551 brk(0x5616f1ec4000)   = 0x5616f1ec4000 <0.000073>

17:26:57.668219 read(0, "1 2 3\n", 1024) = 6 <2.037120>

17:26:59.706542 fstat(1, {st_dev=makedev(0, 0x9c), st_ino=4, st_mode=S_IFCHR|0620, st_nlink=1, st_uid=1000, st_gid=5, st_blksize=1024, st_blocks=0, st_rdev=makedev(0x88, 0x1), st_atime=1765733217 /* 2025-12-14T17:26:57.534358669+0000 */, st_atime_nsec=534358669, st_mtime=1765733217 /* 2025-12-14T17:26:57.534358669+0000 */, st_mtime_nsec=534358669, st_ctime=1765718984 /* 2025-12-14T13:29:44.534358669+0000 */, st_ctime_nsec=534358669}) = 0 <0.000083>

17:26:59.707439 write(1, "6.000000\n", 9) = 9 <0.000118>

17:26:59.708324 read(0, "2 3 5 6 3\n", 1024) = 10 <11.568834>

17:27:11.278274 write(1, "3 5 6\n", 6) = 6 <0.000113>

17:27:11.279118 read(0, "q\n", 1024)   = 2 <3.471041>

17:27:14.751225 lseek(0, -1, SEEK_CUR) = -1 ESPIPE (Illegal seek) <0.000076>

17:27:14.752018 exit_group(0)   = ?

17:27:14.753801 +++ exited with 0 +++

prog_dynamic

17:27:23.409279 execve("./prog_dynamic", ["./prog_dynamic"], 0x7ffc141898b8 /* 35 vars */) = 0 <0.004962>

17:27:23.416432 brk(NULL)   = 0x55b6ab6cf000 <0.000071>

17:27:23.417289 mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f41f8e44000 <0.000071>

17:27:23.417986 access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory) <0.000086>

17:27:23.419018 openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v4/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory) <0.000480>

17:27:23.420778 openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v3/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory) <0.000370>

17:27:23.421852 openat(AT_FDCWD, "./glibc-hwcaps/x86-64-v2/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory) <0.000401>

17:27:23.422956 openat(AT_FDCWD, "./libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory) <0.000359>

17:27:23.423912 openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3 <0.000068>

17:27:23.424608 fstat(3, {st_mode=S_IFREG|0644, st_size=11939, ...}) = 0 <0.000107>

17:27:23.425413 mmap(NULL, 11939, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f41f8e41000 <0.000087>

17:27:23.426255 close(3)   = 0 <0.000080>

17:27:23.427103 openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3 <0.000120>

17:27:23.428061 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@\0\0\0\0\0\0\0\20^
\0\0\0\0\0\0\0\0@\08\0\16\0@\0@\0?\0\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
\20\3\0\0\0\0\0\0\20\3\0\0\0\0\0\0\10\0\0\0\0\0\0\0\3\0\0\0\4\0\0\0\240g\35\0\0\0\0\0\240g\35\0\0\0\0\0\2
40g\35\0\0\0\0\0\34\0\0\0\0\0\0\0\34\0\0\0\0\0\0\0\20\0\0\0\0\0\0\0\1\0\0\0\4\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0"..., 832) = 832 <0.000094>

17:27:23.428923 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\20\3\0\0\0\0\0\0\20\3\0\0\0\0\0\0\10\0\0
\0\0\0\0\0\3\0\0\0\4\0\0\0\240g\35\0\0\0\0\0\240g\35\0\0\0\0\0\240g\35\0\0\0\0\0\34\0\0\0\0\0\0\0\34\0\0
\0\0\0\0\0\20\0\0\0\0\0\0\0\1\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\0\0\0\0\0\220s\2\0\0\0\0\
0\220s\2\0\0\0\0\0\0\20\0\0\0\0\0\0\0\1\0\0\0\5\0\0\0\0\200\2\0\0\0\0\0\0\200\2\0\0\0\0\0\0\200\2\0\0\0\0\0
"..., 784, 64) = 784 <0.000091>

17:27:23.429859 fstat(3, {st_mode=S_IFREG|0755, st_size=2125328, ...}) = 0 <0.000065>

17:27:23.430612 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\20\3\0\0\0\0\0\0\20\3\0\0\0\0\0\0\10\0\0
\0\0\0\0\0\3\0\0\0\4\0\0\0\240g\35\0\0\0\0\0\240g\35\0\0\0\0\0\240g\35\0\0\0\0\0\34\0\0\0\0\0\0\0\34\0\0
\0\0\0\0\0\20\0\0\0\0\0\0\0\1\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\0\0\0\0\0\220s\2\0\0\0\0\
0\220s\2\0\0\0\0\0\0\20\0\0\0\0\0\0\0\1\0\0\0\5\0\0\0\0\200\2\0\0\0\0\0\0\200\2\0\0\0\0\0\0\200\2\0\0\0\0\0
"..., 784, 64) = 784 <0.000087>

17:27:23.431504 mmap(NULL, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f41f8c2f000 <0.000100>

17:27:23.432377 mmap(0x7f41f8c57000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7f41f8c57000 <0.000144>

17:27:23.433129 mmap(0x7f41f8ddf000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b0000) = 0x7f41f8ddf000 <0.000083>

17:27:23.433811 mmap(0x7f41f8e2e000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000) = 0x7f41f8e2e000 <0.000080>

17:27:23.434624 mmap(0x7f41f8e34000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f41f8e34000 <0.000497>

17:27:23.435904 close(3)   = 0 <0.000064>

17:27:23.436698 mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f41f8c2c000 <0.000074>

17:27:23.437372 arch_prctl(ARCH_SET_FS, 0x7f41f8c2c740) = 0 <0.000060>

17:27:23.438017 set_tid_address(0x7f41f8c2ca10) = 31957 <0.000056>

17:27:23.438636 set_robust_list(0x7f41f8c2ca20, 24) = 0 <0.000064>

17:27:23.439642 rseq(0x7f41f8c2d060, 0x20, 0, 0x53053053) = 0 <0.000139>

17:27:23.440761 mprotect(0x7f41f8e2e000, 16384, PROT_READ) = 0 <0.000096>

17:27:23.441607 mprotect(0x55b6764e6000, 4096, PROT_READ) = 0 <0.000084>

17:27:23.442595 mprotect(0x7f41f8e7c000, 8192, PROT_READ) = 0 <0.000086>

17:27:23.443782 prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0 <0.000081>

17:27:23.444802 munmap(0x7f41f8e41000, 11939) = 0 <0.000137>

17:27:23.445834 getrandom("\x92\xc2\xf1\x3e\x30\xe8\xc0\xe1", 8, GRND_NONBLOCK) = 8 <0.000082>

17:27:23.446666 brk(NULL)   = 0x55b6ab6cf000 <0.000091>

17:27:23.447648 brk(0x55b6ab6f0000)   = 0x55b6ab6f0000 <0.000088>

17:27:23.448730 openat(AT_FDCWD, "./libimpl1.so", O_RDONLY|O_CLOEXEC) = 3 <0.001956>

17:27:23.451549 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@\0\0\0\0\0\0\0\3404\0\0\0\0\0\0\0\0\0@\08\0\v\0@\0\31\0\30\0\1\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\0\0\0\0\0\0\320\4\0\0\0\0\0\0\320\4\0\0\0\0\0\0\0\20\0\0\0\0\0\0\1\0\0\0\5\0\0\0\0\20\0\0\0\0\0\0\0\20\0\0\0\0\0\0\0\20\0\0\0\0\0\0\0\25\2\0\0\0\0\0\0\25\2\0\0\0\0\0\0\0\20\0\0\0\0\0\0\1\0\0\0\4\0\0\0\0 \0\0\0\0\0\0\0 \0\0\0\0\0\0"..., 832) = 832 <0.000371>

17:27:23.452597 fstat(3, {st_mode=S_IFREG|0755, st_size=15136, ...}) = 0 <0.000351>

17:27:23.453648 getcwd("/workspaces/OS/lab4/src", 128) = 24 <0.000093>

17:27:23.454613 mmap(NULL, 16400, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f41f8c27000 <0.000111>

17:27:23.455438 mmap(0x7f41f8c28000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7f41f8c28000 <0.000087>

17:27:23.456067 mmap(0x7f41f8c29000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f41f8c29000 <0.000062>

17:27:23.456670 mmap(0x7f41f8c2a000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f41f8c2a000 <0.000078>

17:27:23.458568 close(3) = 0 <0.000090>

17:27:23.459347 mprotect(0x7f41f8c2a000, 4096, PROT_READ) = 0 <0.000080>

17:27:23.460110 openat(AT_FDCWD, "./libimpl2.so", O_RDONLY|O_CLOEXEC) = 3 <0.001606>

17:27:23.462408 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\0\0\0\0\0P5\0\0\0\0\0\0\0\0@\08\0\v\0@\0\34\0\33\0\1\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\0\0\0\0\0\10\5\0\0\0\0\0\0\10\5\0\0\0\0\0\0\0\20\0\0\0\0\0\0\1\0\0\0\5\0\0\0\0\20\0\0\0\0\0\0\20\0\0\0\0\0\0\20\0\0\0\0\0\0\305\2\0\0\0\0\0\0\305\2\0\0\0\0\0\0\20\0\0\0\0\0\0\1\0\0\0\4\0\0\0\0 \0\0\0\0\0\0\0 \0\0\0\0\0\0"..., 832) = 832 <0.000437>

17:27:23.463545 fstat(3, {st_mode=S_IFREG|0755, st_size=15440, ...}) = 0 <0.000295>

17:27:23.464415 getcwd("/workspaces/OS/lab4/src", 128) = 24 <0.000057>

17:27:23.465025 mmap(NULL, 16408, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f41f8c22000 <0.000078>

17:27:23.465806 mmap(0x7f41f8c23000, 4096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7f41f8c23000 <0.000109>

17:27:23.466673 mmap(0x7f41f8c24000, 4096, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f41f8c24000 <0.000099>

17:27:23.467476 mmap(0x7f41f8c25000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f41f8c25000 <0.000068>

17:27:23.469707 close(3) = 0 <0.000069>

17:27:23.470278 mprotect(0x7f41f8c25000, 4096, PROT_READ) = 0 <0.000067>

17:27:23.470841 fstat(0, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x1), ...}) = 0 <0.000060>

17:27:23.471323 read(0, "1 2 3\n", 1024) = 6 <3.222835>

17:27:23.965502 fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x1), ...}) = 0 <0.000062>

17:27:23.966326 write(1, "6.000000\n", 9) = 9 <0.000145>

17:27:23.967195 read(0, "2 3 5 6 3\n", 1024) = 10 <9.926320>

17:27:33.894593 write(1, "3 5 6\n", 6) = 6 <0.000088>

17:27:33.895506 read(0, "0\n", 1024)   = 2 <4.626970>

17:27:38.523633 read(0, "1 2 3\n", 1024) = 6 <2.362873>

17:27:40.887683 write(1, "3.000000\n", 9) = 9 <0.000095>

17:27:40.888532 read(0, "2 3 5 6 3\n", 1024) = 10 <6.684968>

17:27:47.574525 write(1, "3 5 6\n", 6) = 6 <0.000092>

17:27:47.575390 read(0, "q\n", 1024)   = 2 <2.453838>

17:27:50.030268 munmap(0x7f41f8c27000, 16400) = 0 <0.000680>

17:27:50.031689 munmap(0x7f41f8c22000, 16408) = 0 <0.000633>

17:27:50.033081 lseek(0, -1, SEEK_CUR) = -1 ESPIPE (Illegal seek) <0.000088>

17:27:50.033922 exit_group(0)   = ?

17:27:50.035516 +++ exited with 0 +++

Вывод

Я научился писать динамические библиотеки, подключать их как на этапе компиляции, так и при помощи интерфейса ос для работы с динамическими библиотеками. Узнал, какие структуры данных существуют для хранения информации о библиотеках, как работать с ними. Узнал, какие системные вызовы необходимы для работы с динамическими библиотеками.