

Московский Авиационный Институт
(Национальный Исследовательский Университет)
Факультет информационных технологий и прикладной математики
Кафедра вычислительной математики и программирования

**Лабораторная работа №5 по курсу
«Операционные системы»**

Работа с динамическими библиотеками

Студент: Семенов Илья Михайлович
Группа: М80 – 206Б-18
Вариант: 15
Преподаватель: Соколов Андрей Алексеевич
Оценка: _____
Дата: _____
Подпись: _____

Москва, 2019

Постановка задачи

Создать динамическую библиотеку, которая реализует определенный функционал. Далее использовать данную библиотеку двумя способами:

- Подключить библиотеку на этапе линковки
- Подгрузив библиотеку в память с помощью системных вызовов

Вариант задания: 1-4. Библиотека должна обеспечивать работу со списком MD5 сумм.

Общие сведения о программе

Программа представляет собой динамическую библиотеку, состоящую из файлов `list.c`, `list.h`, `MD5.c`, `MD5.h`, а так же заголовочного файла, содержащего объявления всех функций и структур, используемых в библиотеке(содержит только директивы для включения в себя файлов `list.h` и `MD5.h`).

Так же в программе есть два исполняемых файла, демонстрирующих работу с динамической библиотекой — `main_link.c`, `main_syscall.c`

Сборка и подключение библиотеки выполняется системой сборки CMake. Используются команды `add_executable`, `add_library`, `target_link_library(SHARED)`.

Используются следующие системные вызовы

1. **dlopen** — загружает динамическую библиотеку и возвращает `handle`
2. **dlclose** — уменьшает счетчик ссылок на динамический объект в памяти, если он становится равным нулю, то объект выгружается из памяти.
3. **dlsym** — позволяет получить указатель на функцию, находящуюся в динамической библиотеке.
4. **dlerror** — возвращает читабельную строку, описывающую последнюю возникшую ошибку, возникшую при взаимодействии с динамической библиотекой.

Общий метод и алгоритм решения

- Создать файлы динамической библиотеки.
- Собрать библиотеку с помощью CMake и получить файл с расширением `.so`

- Написать две программы, одна из которых использует системные вызовы для открытия библиотеки, а другая подключает библиотеку на этапе линковки.
- Собрать эти программы и проверить корректность их работы

Код программы

list.h:

```
#ifndef LIB_TEST_LIST_H
#define LIB_TEST_LIST_H
#include <stdbool.h>
#include <glob.h>
#include <stdlib.h>
#include <stdio.h>
#include "MD5.h"
#define VALUE_TYPE md5
typedef struct ListNode {
    VALUE_TYPE data;
    struct ListNode* next;
    struct ListNode* prev;
} list_node;
typedef struct List {
    struct ListNode* begin;
    struct ListNode* after_end;
} list;
typedef struct ListIterator {
    struct ListNode* node;
    struct List* list;
} list_iterator;
void list_init(list* l);
void list_clear(list* l);
void list_destroy(list* l);
list_iterator list_begin(list* l);
list_iterator list_end(list* l);
list_iterator list_iter_next(list_iterator it);
list_iterator list_iter_prev(list_iterator it);
list_iterator list_make_iter(list_node* node, list* list);
bool list_iter_equal(list_iterator lhs, list_iterator rhs);
VALUE_TYPE* list_iter_get(list_iterator it);
void list_insert(list_iterator it, VALUE_TYPE val);
void list_erase(list_iterator it);
void list_print(list* l);
bool list_empty(list* l);
size_t list_size(list* l);
#endif //LIB_TEST_LIST_H
```

list.c:

```
#include "list.h"
void list_init(list* l) {
    l->after_end = malloc(sizeof(list_node));
    l->begin = l->after_end;
    l->begin->prev = NULL;
}
void list_print(list* l) {
```

```

    list_iterator it = list_begin(l);
    while (!list_iter_equal(it, list_end(l))) {
        char* str = hash_to_string(*list_iter_get(it));
        printf("%s ", str);
        free(str);
        it = list_iter_next(it);
    }
    printf("\n");
}

void list_clear(list* l) {
    while (l->begin != l->after_end) {
        list_node* temp = l->begin;
        l->begin = l->begin->next;
        free(temp);
    }
}

void list_destroy(list* l) {
    list_clear(l);
    free(l->after_end);
    l->begin = NULL;
    l->after_end = NULL;
}

list_iterator list_begin(list* l) {
    return list_make_iter(l->begin, l);
}

list_iterator list_end(list* l) {
    return list_make_iter(l->after_end, l);
}

list_iterator list_iter_next(list_iterator it) {
    return list_make_iter(it.node->next, it.list);
}

list_iterator list_iter_prev(list_iterator it) {
    return list_make_iter(it.node->prev, it.list);
}

list_iterator list_make_iter(list_node* node, list* list) {
    return (list_iterator){.node = node, .list = list};
}

bool list_iter_equal(list_iterator lhs, list_iterator rhs) {
    return lhs.list == rhs.list && rhs.node == lhs.node;
}

VALUE_TYPE* list_iter_get(list_iterator it) {
    return &it.node->data;
}

void list_insert(list_iterator it, VALUE_TYPE val) {
    list_node* node = it.node;
    list_node* new_node = malloc(sizeof(list_node));
    new_node->data = val;
    if (node->prev == NULL) { // вставка в начало
        node->prev = new_node;
        new_node->next = node;
        new_node->prev = NULL;
        it.list->begin = new_node;
        return;
    }
    new_node->prev = node->prev;
    new_node->next = node;
    node->prev->next = new_node;
}

```

```

        node->prev = new_node;
    }
    void list_erase(list_iterator it) {
        if (it.node == it.list->after_end) {
            return;
        }
        if (it.node->prev == NULL) {
            it.node->next->prev = NULL;
            it.list->begin = it.node->next;
            free(it.node);
            return;
        }
        list_node* temp = it.node;
        it.node->prev->next = it.node->next;
        it.node->next->prev = it.node->prev;
        free(temp);
    }
    bool list_empty(list* l) {
        return l->begin == l->after_end;
    }
    size_t list_size(list* l) {
        size_t size = 0;
        list_node* node = l->begin;
        while (node != l->after_end) {
            size++;
            node = node->next;
        }
        return size;
    }

```

MD5.h:

```

#ifndef LIB_TEST_MD5_H
#define LIB_TEST_MD5_H
#include <stdlib.h>
#include <stdio.h>
typedef struct MD5 {
    unsigned long long part1;
    unsigned long long part2;
} md5;
char get_hex_digit(unsigned number);
md5 make_hash(unsigned long long p1, unsigned long long p2);
char* hash_to_string(md5 hash);
#endif //LIB_TEST_MD5_H

```

MD5.c:

```

#include "MD5.h"
md5 make_hash(unsigned long long p1, unsigned long long p2) {
    return (md5){.part1 = p1, .part2 = p2};
}
char get_hex_digit(unsigned number) {
    if (number < 10) {
        return '0' + number;
    } else {
        return 'A' + number - 10;
    }
}

```

```

    }
}
char* hash_to_string(md5 hash) {
    char* result = malloc(sizeof(char) * 33);
    result[32] = '\0';
    for (int i = 0; i < 32; ++i) {
        unsigned long long* number = (i < 16 ? &hash.part1 : &hash.part2);
        unsigned cur_bits = ((*number) >> ((15 - i) * 4)) & 15u;
        result[i] = get_hex_digit(cur_bits);
    }
    return result;
}

```

main_syscall.c:

```

#include <stdio.h>
#include <stdlib.h>
#include <dlfcn.h>
#include "lib_interface.h"
int main() {
    void* ptr = dlopen("libmy_list.so", RTLD_NOW);
    if(!ptr) {
        printf("%s", dlerror());
        exit(1);
    }
    void (*init)(list*) = dlsym(ptr, "list_init");
    void (*clear)(list*) = dlsym(ptr, "list_clear");
    void (*destroy)(list*) = dlsym(ptr, "list_destroy");
    list_iterator (*begin)(list*) = dlsym(ptr, "list_begin");
    list_iterator (*end)(list*) = dlsym(ptr, "list_end");
    list_iterator (*iter_next)(list_iterator) = dlsym(ptr, "list_iter_next");
    list_iterator (*iter_prev)(list_iterator) = dlsym(ptr, "list_iter_prev");
    void (*insert)(list_iterator, VALUE_TYPE) = dlsym(ptr, "list_insert");
    void (*erase)(list_iterator) = dlsym(ptr, "list_erase");
    void (*print)(list*) = dlsym(ptr, "list_print");
    md5 (*hash)(unsigned long long, unsigned long long) = dlsym(ptr,
"make_hash");
    void* pointers[9] =
{init,clear,begin,end,iter_next,iter_prev,insert,erase,print};
    for (int i = 0; i < 9; ++i) {
        if (!pointers[i]) {
            printf("function is not loaded correctly\n");
            exit(1);
        }
    }
    list l;
    init(&l);
    insert(begin(&l), hash(10,10));
    insert(begin(&l), hash(20,10));
    insert(begin(&l), hash(30,10));
    insert(begin(&l), hash(40,10));
    insert(begin(&l), hash(50,10));
    erase(iter_prev(end(&l)));
    print(&l);
    dlclose(ptr);
}

```

main_link.c:

```
#include <stdio.h>
#include "lib_interface.h"
int main() {
    list ll;
    list_init(&ll);
    list_insert(list_begin(&ll), make_hash(10,10));
    list_insert(list_begin(&ll), make_hash(20,20));
    list_insert(list_begin(&ll), make_hash(30,30));
    list_insert(list_begin(&ll), make_hash(40,40));
    list_insert(list_begin(&ll), make_hash(50,50));
    list_print(&ll);
}
```

lib_interface.h:

```
#ifndef LIB_TEST_LIB_INTERFACE_H
#define LIB_TEST_LIB_INTERFACE_H
#include "MD5.h"
#include "list.h"
#endif //LIB_TEST_LIB_INTERFACE_H
```

Демонстрация работы программы

```
ilya@ilya-lenovo:~/CLionProjects/os_lab_05/cmake-build-debug$  
./lab_compile_link
```

```
0000000000000003200000000000000032 000000000000000280000000000000028  
0000000000000001E000000000000001E 000000000000000140000000000000014  
0000000000000000A000000000000000A
```

```
ilya@ilya-lenovo:~/CLionProjects/os_lab_05/cmake-build-debug$  
./lab_syscall_link
```

```
00000000000000032000000000000000A 00000000000000028000000000000000A  
0000000000000001E000000000000000A 00000000000000014000000000000000A
```

```
ilya@ilya-lenovo:~/CLionProjects/os_lab_05/cmake-build-debug$ strace  
./lab_syscall_link
```

```
execve("./lab_syscall_link", [".lab_syscall_link"], 0x7ffeae6a5c20 /* 53 vars */) =  
0
```

```
brk(NULL)                                = 0x562fb18df000
```

```
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
```

```
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
```

```
openat(AT_FDCWD,  
"/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/tls/x86_64/x86_64/  
libdl.so.2", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
```

```
stat("/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/tls/x86_64/x86_64",  
0x7ffd30f69db0) = -1 ENOENT (No such file or directory)
```

```
openat(AT_FDCWD,  
"/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/tls/x86_64/libdl.so.2",  
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
```

```
stat("/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/tls/x86_64",  
0x7ffd30f69db0) = -1 ENOENT (No such file or directory)
```

```
openat(AT_FDCWD,  
"/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/tls/x86_64/libdl.so.2",  
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
```

```
stat("/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/tls/x86_64",  
0x7ffd30f69db0) = -1 ENOENT (No such file or directory)
```



```

openat(AT_FDCWD,
"/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/tls/libdl.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/tls",
0x7ffd30f69db0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD,
"/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/x86_64/x86_64/
libdl.so.2", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/x86_64/x86_64",
0x7ffd30f69db0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD,
"/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/x86_64/libdl.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/x86_64",
0x7ffd30f69db0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD,
"/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/x86_64/libdl.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/x86_64",
0x7ffd30f69db0) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD,
"/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/libdl.so.2",
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)

stat("/home/ilya/CLionProjects/os_lab_05/cmake-build-debug",
{st_mode=S_IFDIR|0755, st_size=4096, ...}) = 0

openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3

fstat(3, {st_mode=S_IFREG|0644, st_size=152516, ...}) = 0

mmap(NULL, 152516, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f1c86ac1000

close(3) = 0

access("/etc/ld.so.nohwcap", F_OK) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libdl.so.2", O_RDONLY|
O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\16\0\0\0\0\0"..., 832) =
832

```

```

fstat(3, {st_mode=S_IFREG|0644, st_size=14560, ...}) = 0

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
MAP_ANONYMOUS, -1, 0) = 0x7f1c86abf000

mmap(NULL, 2109712, PROT_READ|PROT_EXEC, MAP_PRIVATE|
MAP_DENYWRITE, 3, 0) = 0x7f1c866bc000

mprotect(0x7f1c866bf000, 2093056, PROT_NONE) = 0

mmap(0x7f1c868be000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0x7f1c868be000

close(3)
= 0

openat(AT_FDCWD,
"/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/libc.so.6", O_RDONLY|
O_CLOEXEC) = -1 ENOENT (No such file or directory)

access("/etc/ld.so.nohwcap", F_OK) = -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\3\0>\0\1\0\0\0\260\34\2\0\0\0\0\0"...,
832) = 832

fstat(3, {st_mode=S_IFREG|0755, st_size=2030544, ...}) = 0

mmap(NULL, 4131552, PROT_READ|PROT_EXEC, MAP_PRIVATE|
MAP_DENYWRITE, 3, 0) = 0x7f1c862cb000

mprotect(0x7f1c864b2000, 2097152, PROT_NONE) = 0

mmap(0x7f1c866b2000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|
MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7f1c866b2000

mmap(0x7f1c866b8000, 15072, PROT_READ|PROT_WRITE, MAP_PRIVATE|
MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f1c866b8000

close(3)
= 0

mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|
MAP_ANONYMOUS, -1, 0) = 0x7f1c86abc000

arch_prctl(ARCH_SET_FS, 0x7f1c86abc740) = 0

mprotect(0x7f1c866b2000, 16384, PROT_READ) = 0

mprotect(0x7f1c868be000, 4096, PROT_READ) = 0

mprotect(0x562fb0ae2000, 4096, PROT_READ) = 0

```

```

mprotect(0x7f1c86ae7000, 4096, PROT_READ) = 0
munmap(0x7f1c86ac1000, 152516) = 0
openat(AT_FDCWD,
"/home/ilya/CLionProjects/os_lab_05/cmake-build-debug/libmy_list.so",
O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\n\0\0\0\0\0"..., 832)
= 832
brk(NULL) = 0x562fb18df000
brk(0x562fb1900000) = 0x562fb1900000
fstat(3, {st_mode=S_IFREG|0775, st_size=19000, ...}) = 0
mmap(NULL, 2105488, PROT_READ|PROT_EXEC, MAP_PRIVATE|
MAP_DENYWRITE, 3, 0) = 0x7f1c860c8000
mprotect(0x7f1c860ca000, 2093056, PROT_NONE) = 0
mmap(0x7f1c862c9000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|
MAP_FIXED|MAP_DENYWRITE, 3, 0x1000) = 0x7f1c862c9000
close(3) = 0
mprotect(0x7f1c862c9000, 4096, PROT_READ) = 0
fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
write(1, "00000000000000032000000000000000A"...,
1330000000000000032000000000000000A
00000000000000280000000000000000A 000000000000001E0000000000000000A
00000000000000014000000000000000A
) = 133
munmap(0x7f1c860c8000, 2105488) = 0
exit_group(0) = ?
+++ exited with 0 +++

```

Вывод

В результате данной работы я научился создавать статические и динамические библиотеки, ознакомился с тем, как они размещаются в памяти и каким образом программы могут подключать их(с помощью системных вызовов или на этапе линковки).