Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №4 по курсу**

**«Операционные системы»**

**ДИНАМИЧЕСКИЕ БИБЛИОТЕКИ**

Студент: Салихов Тимур Русланович

Группа: М8О–212Б–22

Вариант: 34

Преподаватель: Соколов Андрей Алексеевич

Оценка: \_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2020.

**Постановка задачи**

## Цель работы

Целью является приобретение практических навыков в:

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

## Задание

Требуется создать динамическую библиотеку, которая реализует определенный функционал. Далее использовать данную библиотеку 2-мя способами:

1. Во время компиляции (на этапе «линковки»/linking)
2. Во время исполнения программы, подгрузив библиотеку в память с помощью системных вызовов

В конечном итоге, программа должна состоять из следующих частей:

* Динамическая библиотека, реализующая заданных вариантом интерфейс;
* Тестовая программа, которая используют библиотеку, используя знания полученные на этапе компиляции;
* Тестовая программа, которая использует библиотеку, используя только местоположение динамической библиотеки и ее интерфейс.

Провести анализ между обоими типами использования библиотеки.

Вариант 34:

7) Подсчет площади плоской геометрической фигуры подвум сторонам

Float Square(float A, float B)

Фигура прямоугольник

Фигура прямоугольный треугольник

8) Перевод числа x из десятичной системы счисления в другую

Char\* translation(long x)

Другая система счисления двоичная

Другая система счисления троичная

**Общие сведения о программе**

Программа компилируется из файлов compile.cpp и runtime.cpp, также используется заголовочный файл functions.hpp. В программе используются следующие системные вызовы:

1. **dlopen** – загружает динамический общий объект (общую библиотеку) из файла, имя которого указано в строке filename (завершается null) и возвращает непрозрачный описатель на загруженный объект.
2. **dlsym** – функция возвращает адрес, по которому символ расположен в памяти(указывается одним из аргументов).
3. **dlclose** – уменьшает счётчик ссылок на динамически загружаемый общий объект, на который ссылается handle. Если счётчик ссылок достигает нуля, то объект выгружается. Все общие объекты, которые были автоматически загружены при вызове dlopen() для объекта, на который ссылается handle, рекурсивно закрываются таким же способом.

**Общий метод и алгоритм решения**.

Первый вариант исполнения реализуется в программе complile.cpp. При помощи cmake создаются 2 библиотеки с разными версиями имплементации функций. Далее одна из библиотек линкуется к исполняемому файлу.

Второй вариант исполнения реализуется в программе runtime.cpp. В нем поключение библиотек происходит при помощи системных вызовов. Также реализована возможность переключения между разными вариантами имплементации.

Различные имплементации представлены в файлах impl1.cpp и impl2.cpp.

**Основные файлы программы**

**compile.cpp:**

#include <iostream>

#include "functions.hpp"

auto main() -> int {

std::cout << "This program is linking libraries during compilation. Instructions: " << std::endl

<< "\"1\" - calculate square of figure" << std::endl

<< "\"2\" - translate number from decimal to other system" << std::endl

<< "\"-1\" - exit" << std::endl

<< "Input instruction: ";

int option;

std::cin >> option;

while (option != -1) {

switch (option) {

case 1:

float A, B;

std::cin >> A >> B;

std::cout << "The square is: " << square(A, B) << std::endl

<< "Input instruction: ";

std::cin >> option;

break;

case 2:

long num;

std::cin >> num;

std::cout << "The num in other system is: " << translation(num) << std::endl

<< "Input instruction: ";

std::cin >> option;

break;

default:

option = -1;

}

}

return 0;

}

**runtime.cpp:**

#include <dlfcn.h>

#include <iostream>

auto main() -> int {

std::cout << "This program is linking libraries during runtime. Instructions: " << std::endl

<< "\"0\" - change implementation" << std::endl

<< "\"1\" - calculate square of figure" << std::endl

<< "\"2\" - translate number from decimal to other system" << std::endl

<< "\"-1\" - exit" << std::endl

<< "Input instruction: ";

int option;

std::cin >> option;

int impl = 1;

auto handle = dlopen("libimpl1.dylib", RTLD\_GLOBAL);

char\* error;

error = dlerror();

if (error) {

std::cerr << error << std::endl;

return 1;

}

float (\*square)(float, float);

square = reinterpret\_cast<decltype(square)>(dlsym(handle, "square"));

error = dlerror();

if (error) {

std::cerr << error << std::endl;

return 1;

}

char\* (\*translation)(long);

translation = reinterpret\_cast<decltype(translation)>(dlsym(handle, "translation"));

error = dlerror();

if (error) {

std::cerr << error << std::endl;

return 1;

}

while (option != -1) {

switch (option) {

case 0:

dlclose(handle);

error = dlerror();

if (error) {

std::cerr << error << std::endl;

return 1;

}

switch (impl) {

case 1:

impl = 2;

handle = dlopen("libimpl2.dylib", RTLD\_GLOBAL);

break;

case 2:

int impl = 1;

handle = dlopen("libimpl1.dylib", RTLD\_GLOBAL);

break;

}

char\* error;

error = dlerror();

if (error) {

std::cerr << error << std::endl;

return 1;

}

square = reinterpret\_cast<decltype(square)>(dlsym(handle, "square"));

error = dlerror();

if (error) {

std::cerr << error << std::endl;

return 1;

}

translation = reinterpret\_cast<decltype(translation)>(dlsym(handle, "translation"));

error = dlerror();

if (error) {

std::cerr << error << std::endl;

return 1;

}

std::cout << "Implementation switched" << std::endl

<< "Input instruction: ";

std::cin >> option;

break;

case 1:

float A, B;

std::cin >> A >> B;

std::cout << "The square is: " << square(A, B) << std::endl

<< "Input instruction: ";

std::cin >> option;

break;

case 2:

long num;

std::cin >> num;

std::cout << "The num in other system is: " << translation(num) << std::endl

<< "Input instruction: ";

std::cin >> option;

break;

default:

option = -1;

}

}

dlclose(handle);

error = dlerror();

if (error) {

std::cerr << error << std::endl;

return 1;

}

return 0;

}

**functions.hpp:**

#pragma once

extern "C" {

float square(float A, float B);

}

extern "C" {

char\* translation(long x);

}

**impl1.cpp:**

#include <iostream>

#include "functions.hpp"

extern "C" {

float square(float A, float B) {

return A \* B;

}

}

extern "C" {

char\* translation(long x) {

char\* num = new char[1000];

char\* rev = new char[1000];

size\_t i = 0;

while (x != 0) {

int digit = x % 2;

rev[i] = digit + 48;

x /= 2;

++i;

}

for (size\_t j = 0; j != i; ++j) {

num[j] = rev[i - j - 1];

}

delete[] rev;

return num;

}

}

**impl2.cpp:**

#include <iostream>

#include "functions.hpp"

extern "C" {

float square(float A, float B) {

return A \* B / 2;

}

}

extern "C" {

char\* translation(long x) {

char\* num = new char[1000];

char\* rev = new char[1000];

size\_t i = 0;

while (x != 0) {

int digit = x % 3;

rev[i] = digit + 48;

x /= 3;

++i;

}

for (size\_t j = 0; j != i; ++j) {

num[j] = rev[i - j - 1];

}

delete[] rev;

return num;

}

}

**CMakeLists.txt:**

cmake\_minimum\_required(VERSION 3.10)

project(Dynamic\_Libraries)

set(CMAKE\_CXX\_STANDARD\_REQUIRED ON)

set(CMAKE\_CXX\_STANDARD 17)

add\_library(impl1 SHARED impl1.cpp)

add\_library(impl2 SHARED impl2.cpp)

add\_executable(comp compile.cpp)

target\_link\_libraries(comp impl1)

add\_executable(runtime runtime.cpp)

**Пример работы**

**Test 1:**

**timursalihov@MacBook-Pro-Timur-2 ~/O/L/b/src (report)>** ./comp

This program is linking libraries during compilation. Instructions:

"1" - calculate square of figure

"2" - translate number from decimal to other system

"-1" - exit

Input instruction: 1 2 3

The square is: 6

Input instruction: 2 13

The num in other system is: 1101

Input instruction: -1

**Test 2:**

**timursalihov@MacBook-Pro-Timur-2 ~/O/L/b/src (report)>** ./comp

This program is linking libraries during compilation. Instructions:

"1" - calculate square of figure

"2" - translate number from decimal to other system

"-1" - exit

Input instruction: 1 2 3

The square is: 3

Input instruction: 2 13

The num in other system is: 111

Input instruction: -1

**Test 3:**

**timursalihov@MacBook-Pro-Timur-2 ~/O/L/b/src (report)>** ./runtime

This program is linking libraries during runtime. Instructions:

"0" - change implementation

"1" - calculate square of figure

"2" - translate number from decimal to other system

"-1" - exit

Input instruction: 1 2 3

The square is: 6

Input instruction: 1 1 3

The square is: 3

Input instruction: 2 13

The num in other system is: 1101

Input instruction: 2 4

The num in other system is: 100

Input instruction: 0

Implementation switched

Input instruction: 1 2 3

The square is: 3

Input instruction: 1 1 3

The square is: 1.5

Input instruction: 2 13

The num in other system is: 111

Input instruction: 2 4

The num in other system is: 11

Input instruction: 0

Implementation switched

Input instruction: 1 2 3

The square is: 6

Input instruction: 1 1 3

The square is: 3

Input instruction: 2 13

The num in other system is: 1101

Input instruction: 2 4

The num in other system is: 100

Input instruction: -1

**Вывод**

Во время выполнения данной работы я познакомился с динамическими библиотеками. Я изучил создание библиотек при помощи CMake, а также мною было изучено подключение библиотек 2 способами: на этапе компиляции и на этапе исполнения программы. Подключение библиотек на этапе компиляции позволяет динамически изменять функции в зависимости от подключенной библиотеки, но замедляет время выполнения, так как требуется время на подключение библиотек.