

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

Лабораторная работа №5 по курсу
«Операционные системы»

Тема работы
«Динамические библиотеки»

Студент: Дубровин Дмитрий Константинович
Группа: М8О-207Б-21
Вариант: 26
Преподаватель: Миронов Евгений Сергеевич
Оценка: _____
Дата: _____
Подпись: _____

Москва, 2022

Содержание

1. Репозиторий
2. Постановка задачи
3. Общие сведения о программе
4. Общий метод и алгоритм решения
5. Исходный код
6. Демонстрация работы программы
7. Выводы

Репозиторий

https://github.com/1droozd1/os_labs/tree/main/lab_5

Постановка задачи

Требуется создать динамические библиотеки, которые реализуют определенный 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 использует одну из библиотек, используя знания полученные на этапе компиляции;
- Тестовая программа (*программа №2*), которая загружает библиотеки, используя только их местоположение 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*).
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 26

Функция 1:

Подсчёт наибольшего общего делителя для двух натуральных чисел

Int GCF(int A, int B)

Реализация 1:

Алгоритм Евклида

Реализация 2:

Наивный алгоритм. Пытаться разделить числа на все числа, что меньше А и В.

Функция 2:

Отсортировать целочисленный массив

Int * Sort(int * array, int size)

Реализация 1:

Пузырьковая сортировка

Реализация 2:

Сортировка Хоара

Система сборки: CMake.

Вариант 5:

Типы чисел, используемых при вычислениях, в библиотеках меняются в зависимости от операционной системы, на которой происходит сборка.

Windows: без изменений

Linux: int -> long, float -> double

MacOS: int -> short, float -> long double

Сигнатуры функций остаются без изменений.

Общие сведения о программе

Программа состоит из двух интерфейсов (main1.cpp и main2.cpp), каждый из них реализован по-разному, в соответствии с заданием. Также каждая реализация контрактов представляет из себя отдельный файл: lib1.cpp и lib2.cpp. Для объявления необходимых функций также используется заголовочный файл lib.h, а для объявления типов: заголовочный файл types.h. Так как все собирается с помощью CMake, то в проекте присутствует CMakeLists.txt.

Общий метод и алгоритм решения

Объявим необходимые функции внутри файла `lib.h`. Используем спецификатор хранения `extern`, который сообщает компилятору, что находящиеся за ним типы и имена переменных объявляются где-то в другом месте.

Так как по заданию необходимо подключать библиотеки на этапе линковки, то подключать `lib.h` в реализации `lib1.cpp` и `lib2.cpp` не следует. В этих файлах просто напишем логику работы необходимых функций. Важно, чтобы они назывались также, как и те, что объявлены в `lib.h`.

Интерфейс 1:

Подключаем `lib.h` и пользуемся функциями так, как будто библиотека обычная. Различия наступают в сборке программы. Если бы мы собирали такой код в терминале, то прописали бы `g++ -c -fPIC lib1.cpp`. Опция `-fPIC` требует от компилятора, при создании объектных файлов, порождать позиционно-независимый код. Формат позиционно-независимого кода позволяет подключать исполняемые модули к коду основной программы в момент её загрузки. Далее `g++ -shared -o liblib1.so lib1.o -lm`. Опция `-shared` указывает `g++`, что в результате должен быть собран не исполняемый файл, а разделяемый объект — динамическая библиотека.

Интерфейс 2:

Воспользуемся системными вызовами из библиотеки `<dlfcn.h>`.

Функция `dlopen` открывает динамическую библиотеку (объект `.so`) по названию.

Функция `dlsym` - обработчик динамически загруженного объекта вызовом `dlopen` (мы передаём `dlsym` первым аргументом описатель динамического объекта, вызванного в `dlopen`).

Функция `dlclose`, соответственно, закрывает динамическую библиотеку.

Собираем с помощью `g++ -L. -Wall -o main.out main2.c -llib2 -llib1`. Флаг `-L.` Означает, что поиск файлов библиотек будет начинаться с текущей директории.

Исходный код

types.h

```
#include "stdlib.h"
#include "stdio.h"

#if OS == APPLE
typedef short os_int;
typedef long double os_float;
//#endif

#elif OS == UNIX
typedef long os_int;
typedef double os_float;
//#endif

#elif OS == WIN32
typedef int os_int;
typedef float os_float;
#endif
```

lib.h

```
#ifndef __LIB_H__
#define __LIB_H__
#include "stdlib.h"
#include "stdio.h"
#include "types.h"

extern "C" os_int GCD(os_int A, os_int B);
extern "C" os_int *Sort(os_int *array, os_int size);
#endif
```

```
#endif
```

lib1.cpp

```
#include <stdio.h>
#include <iostream>
#include "../src/types.h"

using namespace std;

int gcd(int a, int b) {
    if (a == 0) {
        return b;
    }
    return gcd(b % a, a);
}

extern "C" os_int GCD(os_int A, os_int B)
{
    cout << "\nEuclidean algorithm: ";
    return gcd(A, B);
}
```

```

extern "C" os_int *Sort(os_int *array, os_int size)
{
    int i, j;
    for (i = 0; i < size - 1; i++)
    {
        for (j = 0; j < size - 1 - i; j++)
        {
            if (array[j] > array[j + 1])
            {
                os_int temp = array[j + 1];
                array[j + 1] = array[j];
                array[j] = temp;
            }
        }
    }
    cout << "\nSorting an array with a bubble: ";
    return array;
}

```

lib2.cpp

```

#include <stdio.h>
#include <iostream>
#include <algorithm>
#include "../src/types.h"

```

```

using namespace std;

```

```

extern "C" os_int GCD(os_int A, os_int B)
{
    cout << "\nNaive algorithm: ";
    int result = 1;
    for (int i = 1; i <= min(A, B); i++) {
        if (A % i == 0 && B % i == 0) {
            result = i;
        }
    }
    return result;
}

```

```

void quickSort (os_int *array, os_int low, os_int high)
{
    os_int i = low;
    os_int j = high;
    os_int pivot = array[(i + j) / 2];
    os_int temp;

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

```

```

        array[j] = temp;
        i++;
        j--;
    }
}
if (j > low)
    quickSort (array, low, j);
if (i < high)
    quickSort (array, i, high);
}

extern "C" os_int *Sort(os_int *array, os_int size)
{
    quickSort(array, 0, size-1);
    cout << "\nSorting an array using Hoare sorting: ";
    return array;
}

```

main1.cpp

```

#include <stdio.h>
#include <iostream>
#include "../src/lib.h"

```

```

using namespace std;

```

```

int main(int argc, char const *argv[])
{
    cout << "Введите: [key] [arg1] ... [argN]\n";
    cout << "Если вы хотите найти наибольший общий делитель для двух\n";
    cout << "натуральных чисел: 1 [first number] [second number]\n";
    cout << "Если вы хотите отсортировать массив: 2 [size] [array[0]]\n";
    cout << "[array[1]] ... [array[size-1]]\n";
    int key;
    while(cin >> key){
        if (key == 1){
            os_int A,B;
            cin >> A >> B;;
            cout << GCD(A,B) << "\n";
        }
        if (key == 2){
            os_int size;
            cin >> size;
            os_int a[size];
            os_int *b;
            for (int i=0; i<size; i++)
                cin >> a[i];
            b = Sort(a, size);
            for (int i=0; i<size; i++)
                cout << b[i] << " ";
            cout << "\n";
        }
        if (key != 1 and key != 2){
            cout << "\nНеправильный ключ\n";
        }
        cout << "\nВведите: [key] [arg1] ... [argN]\n";
    }
    cout << "Если вы хотите найти наибольший общий делитель для двух

```



```

натуральных чисел: 1 [first number] [second number]\n";
    cout << "Если вы хотите отсортировать массив: 2 [size] [array[0]]
[array[1]] ... [array[size-1]]\n";
}
}

```

main2.cpp

```

#include <stdio.h>
#include <iostream>
#include <dlfcn.h>
#include "../src/types.h"

```

```

using namespace std;

```

```

const char* lib1 = "./liblib1.so";
const char* lib2 = "./liblib2.so";

```

```

int main(int argc, char const *argv[])
{
    cout << "Введите: [key] [arg1] ... [argN]\n";
    cout << "Если вы хотите поменять метод вычислений, введите 0\n";
    cout << "Если вы хотите найти наибольший общий делитель для двух
натуральных чисел: 1 [first number] [second number]\n";
    cout << "Если вы хотите отсортировать массив: 2 [size] [array[0]]
[array[1]] ... [array[size-1]]\n";
    int key;
    int checker = 0;
    void* CurrentLib = dlopen(lib1, RTLD_LAZY);
    cout << "Библиотека: " << checker << "\n";
    os_int (*GCD)(os_int A, os_int B);
    os_int* (*Sort)(os_int *array, os_int size);
    GCD = (os_int (*)(os_int, os_int)) dlsym(CurrentLib, "GCD");
    Sort = (os_int* (*)(os_int*, os_int)) dlsym(CurrentLib, "Sort");
    while(cin >> key){
        if (key == 0){
            dlclose(CurrentLib);
            if (checker == 0){
                CurrentLib = dlopen(lib2, RTLD_LAZY);
            } else {
                CurrentLib = dlopen(lib1, RTLD_LAZY);
            }
            checker = !checker;
            GCD = (os_int (*)(os_int, os_int)) dlsym(CurrentLib, "GCD");
            Sort = (os_int* (*)(os_int*, os_int)) dlsym(CurrentLib, "Sort");
        }
        if (key == 1){
            os_int A,B;
            cin >> A >> B;
            cout << GCD(A,B) << "\n";
        }
        if (key == 2){
            os_int size;
            cin >> size;
            os_int a[size], *b;
            for (int i=0; i<size; i++)
                cin >> a[i];
        }
    }
}

```

```

        b = Sort(a, size);
        for (int i=0; i<size; i++)
            cout << b[i] << " ";
        cout << "\n";
    }
    if (key != 0 and key != 1 and key != 2){
        cout << "\nНеправильный ключ\n";
    }
    cout << "\nВведите: [key] [arg1] ... [argN]\n";
    cout << "Если вы хотите поменять метод вычислений, введите 0\n";
    cout << "Если вы хотите найти наибольший общий делитель для двух
натуральных чисел: 1 [first number] [second number]\n";
    cout << "Если вы хотите отсортировать список: 2 [size] [array[0]]
[array[1]] ... [array[size-1]]\n";
    cout << "Библиотека: " << checker << "\n";
}
}

```

CMakeLists.txt

```

cmake_minimum_required(VERSION 3.22.0)
project(5 LANGUAGES CXX)

```

```

set(DYNAMIC_SOURCE src/main2.cpp)
set(STATIC_SOURCE src/main1.cpp)

```

```

set(LIBRARIES1_SOURCE src/lib1.cpp)
set(LIBRARIES2_SOURCE src/lib2.cpp)
set(LIBRARIES_INCLUDE src/lib.h)

```

```

add_library(
    lib1 SHARED
    ${LIBRARIES_INCLUDE}
    ${LIBRARIES1_SOURCE}
)

```

```

add_library(
    lib2 SHARED
    ${LIBRARIES_INCLUDE}
    ${LIBRARIES2_SOURCE}
)

```

```

add_library(
    lib_st1 STATIC
    ${LIBRARIES_INCLUDE}
    ${LIBRARIES1_SOURCE}
)

```

```

add_library(
    lib_st2 STATIC
    ${LIBRARIES_INCLUDE}
    ${LIBRARIES2_SOURCE}
)

```

```

if (UNIX)

```

```

    add_definitions(-DOS=UNIX)
endif()

if (APPLE)
    add_definitions(-DOS=APPLE)
endif()

if (WIN32)
    add_definitions(-DOS=WIN32)
endif()

add_executable(main2 ${DYNAMIC_SOURCE})
add_executable(main11 ${STATIC_SOURCE})
add_executable(main12 ${STATIC_SOURCE})

target_link_libraries(main11 PRIVATE lib_st1)
target_link_libraries(main12 PRIVATE lib_st2)

target_link_libraries(main2 PRIVATE dl)

```

Демонстрация работы программы

```

dr0ozd1@Dmitry-Nitro-AN515-45:~/Coding/os_labs/lab_5$ cmake src
-- The CXX compiler identification is GNU 11.3.0
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /home/dr0ozd1/Coding/os_labs/lab_5
dr0ozd1@Dmitry-Nitro-AN515-45:~/Coding/os_labs/lab_5$ make
[ 7%] Building CXX object CMakeFiles/lib1.dir/lib1.cpp.o
[ 14%] Linking CXX shared library liblib1.so
[ 14%] Built target lib1
[ 21%] Building CXX object CMakeFiles/lib2.dir/lib2.cpp.o
[ 28%] Linking CXX shared library liblib2.so
[ 28%] Built target lib2
[ 35%] Building CXX object CMakeFiles/lib_st1.dir/lib1.cpp.o
[ 42%] Linking CXX static library liblib_st1.a
[ 42%] Built target lib_st1
[ 50%] Building CXX object CMakeFiles/lib_st2.dir/lib2.cpp.o
[ 57%] Linking CXX static library liblib_st2.a
[ 57%] Built target lib_st2
[ 64%] Building CXX object CMakeFiles/main2.dir/main2.cpp.o
[ 71%] Linking CXX executable main2
[ 71%] Built target main2
[ 78%] Building CXX object CMakeFiles/main11.dir/main1.cpp.o
[ 85%] Linking CXX executable main11
[ 85%] Built target main11
[ 92%] Building CXX object CMakeFiles/main12.dir/main1.cpp.o
[100%] Linking CXX executable main12
[100%] Built target main12
dr0ozd1@Dmitry-Nitro-AN515-45:~/Coding/os_labs/lab_5$ ./main11

```

Введите: [key] [arg1] ... [argN]

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать массив: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

1 12 24

Euclidean algorithm: 12

Введите: [key] [arg1] ... [argN]

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать массив: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

2 4 6 1 8 -1

Sorting an array with a bubble: -1 1 6 8

Введите: [key] [arg1] ... [argN]

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать массив: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

dr0ozd1@Dmitry-Nitro-AN515-45:~/Coding/os_labs/lab_5\$./main12

Введите: [key] [arg1] ... [argN]

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать массив: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

1 12 24

Naive algorithm: 12

Введите: [key] [arg1] ... [argN]

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать массив: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

2 5 1 3 0 -10 24

Sorting an array using Hoare sorting: -10 0 1 3 24

Введите: [key] [arg1] ... [argN]

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать массив: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

dr0ozd1@Dmitry-Nitro-AN515-45:~/Coding/os_labs/lab_5\$./main12

Введите: [key] [arg1] ... [argN]

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать массив: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

1 12 24

Naive algorithm: 12

12

Введите: [key] [arg1] ... [argN]

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать массив: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

2 5 1 3 0 -10 24

Sorting an array using Hoare sorting: -10 0 1 3 24

Введите: [key] [arg1] ... [argN]

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать массив: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

dr0ozd1@Dmitry-Nitro-AN515-45:~/Coding/os_labs/lab_5\$./main2

Введите: [key] [arg1] ... [argN]

Если вы хотите поменять метод вычислений, введите 0

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать массив: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

Библиотека: 0

1 32 18

Euclidean algorithm: 2

Введите: [key] [arg1] ... [argN]

Если вы хотите поменять метод выислений, введите 0

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать список: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

Библиотека: 0

2 5 -10 -20 10 12 5

Sorting an array with a bubble: -20 -10 5 10 12

Введите: [key] [arg1] ... [argN]

Если вы хотите поменять метод выислений, введите 0

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать список: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

Библиотека: 0

0

Введите: [key] [arg1] ... [argN]

Если вы хотите поменять метод выислений, введите 0

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать список: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

Библиотека: 1

1 32 19

Naive algorithm: 1

Введите: [key] [arg1] ... [argN]

Если вы хотите поменять метод выислений, введите 0

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать список: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

Библиотека: 1

1 35 70

Naive algorithm: 35

Введите: [key] [arg1] ... [argN]

Если вы хотите поменять метод выислений, введите 0

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать список: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

Библиотека: 1

2 5 10 4 32 11 -10

Sorting an array using Hoare sorting: -10 4 10 11 32

Введите: [key] [arg1] ... [argN]

Если вы хотите поменять метод выислений, введите 0

Если вы хотите найти наибольший общий делитель для двух натуральных чисел: 1 [first number] [second number]

Если вы хотите отсортировать список: 2 [size] [array[0]] [array[1]] ... [array[size-1]]

Библиотека: 1

Выводы

В процессе выполнения лабораторной работы я изучил, как создавать динамические библиотеки в операционной системе Linux, а также получил опыт работы с ними. Я узнал, что такие библиотеки могут быть загружены в процессе выполнения программы, что расширяет возможности ее функционирования.