Белорусский Национальный Технический Университет

Кафедра «Инновационные процессы»

Лабораторная работа № 4

 «Создание и использование библиотек Linux»

|  |
| --- |
| Выполнил: студент группы 42221  Мельник К.А. Чушпан К.  Проверил:  Станкевич С.Н. |

МИНСК 2023

Цель работы:

Изучить и закрепить на практике создание и использование статически и динамически подключаемых библиотек в операционных системах семейства Linux.

Задания:

**Задание 1**

Произвести рефакторинг проекта предыдущей лабораторной работы по автосборке проекта: вынесите код функций бизнес логики в отдельную статическую библиотеку. Для автосборки проекта используйте утилиту make, обязательно. При помощи программы ldd проверим зависимости, образовавшиеся между библиотеками. Сравните сумму размеров объектных файлов проекта и исполняемого файла. Результаты отобразите скриншотами. Копию исполняемого файла расположите на «Рабочем столе», запустите программу. Какой будет результат?

Листинг кода:

main.cpp:

#include "business\_logic2.h"

#include "business\_logic.h"

using namespace std;

int main() {

int result = calculateResult(5, 7);

PrintResult(result);

return 0;

}

business\_logic.cpp:

#include "business\_logic.h"

int calculateResult(int a, int b) {

return a + b;

}

business\_logic2.cpp:

#include <iostream>

#include "business\_logic2.h"

using namespace std;

void PrintResult(int result) {

cout << "Result: " << result << endl;

}

business\_logic.h:

#pragma once

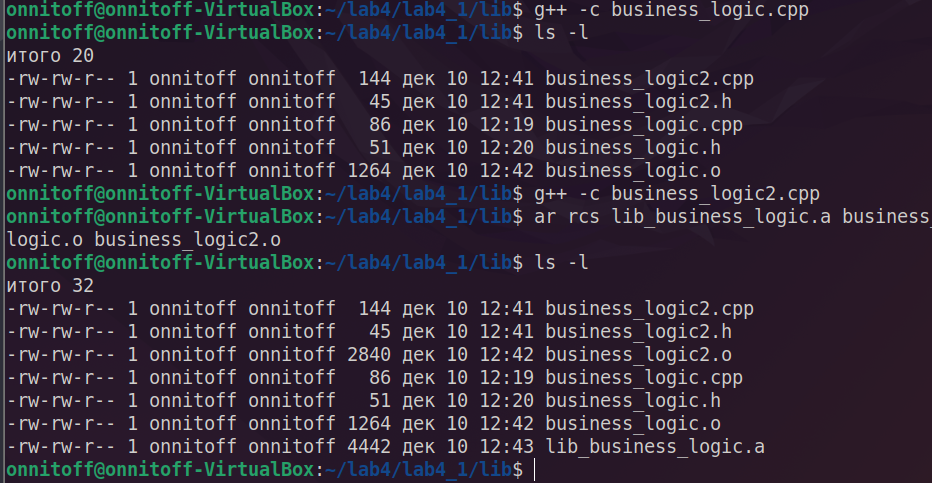
int calculateResult(int a, int b);

business\_logic2.h:

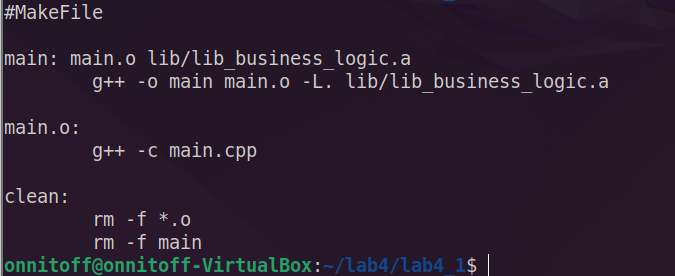
#pragma once

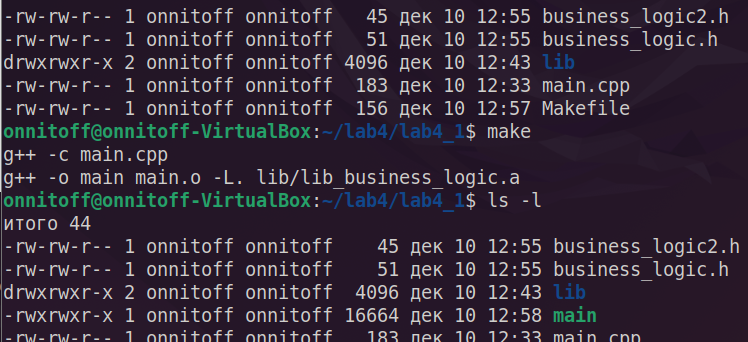
void PrintResult(int result);

Создание статической библиотеки:

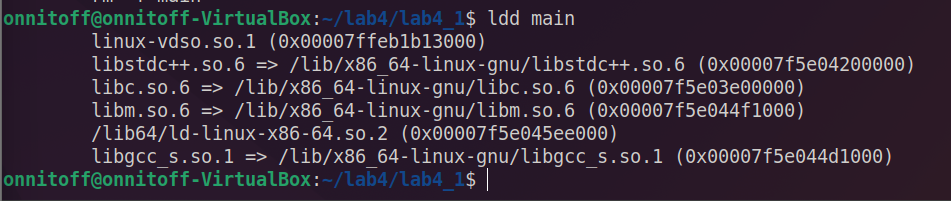


Сборка проекта:

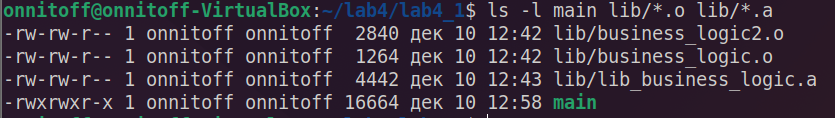


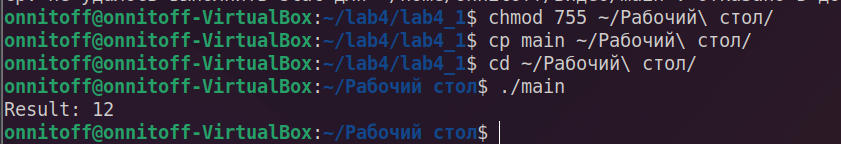


Зависимости между библиотеками:



Сравнение размеров файлов:





**Задание 2**

Произвести рефакторинг проекта предыдущей лабораторной работы по автосборке проекта: вынесите код функций бизнес логики в отдельную динамическую библиотеку со статическим вызовом. Для автосборки проекта используйте утилиту make, обязательно. При помощи программы ldd проверим зависимости, образовавшиеся между библиотеками. Сравните сумму размеров объектных файлов проекта и исполняемого файла. Результаты отобразите скриншотами. Копию исполняемого файла расположите на «Рабочем столе», запустите программу. Какой будет результат? Если возникла проблема, решите ее и поясните каким способом. Сравните результаты первого и вторгог упражнения, сделайте вывод.

Листинг кода:

main.cpp:

#include "business\_logic2.h"

#include "business\_logic.h"

using namespace std;

int main() {

int result = calculateResult(5, 7);

PrintResult(result);

return 0;

}

business\_logic.cpp:

#include "business\_logic.h"

int calculateResult(int a, int b) {

return a + b;

}

business\_logic2.cpp:

#include <iostream>

#include "business\_logic2.h"

using namespace std;

void PrintResult(int result) {

cout << "Result: " << result << endl;

}

business\_logic.h:

#pragma once

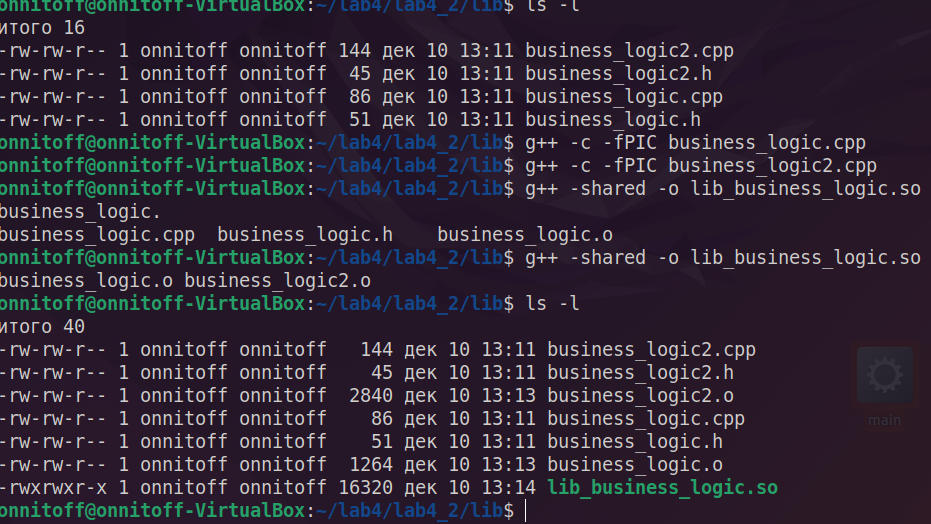
int calculateResult(int a, int b);

business\_logic2.h:

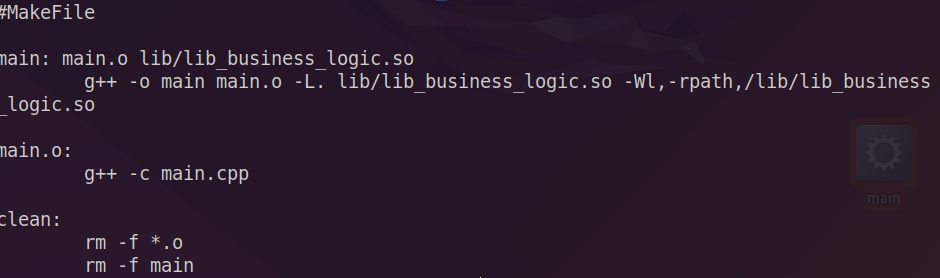
#pragma once

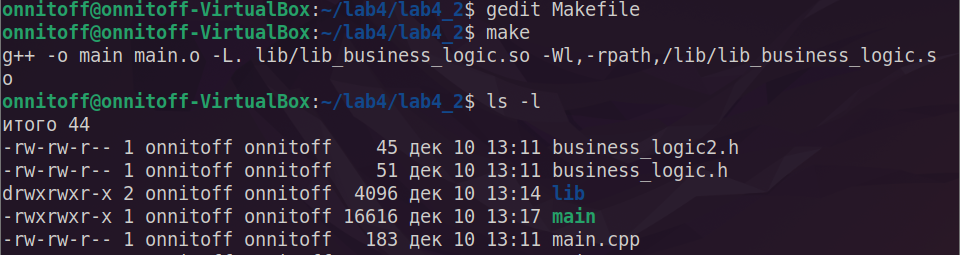
void PrintResult(int result);

Создание динамической библиотеки:

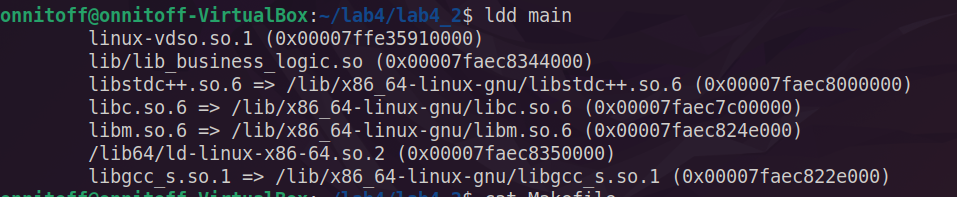


Сборка проекта:

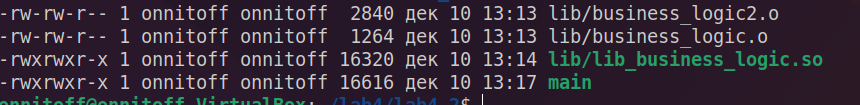




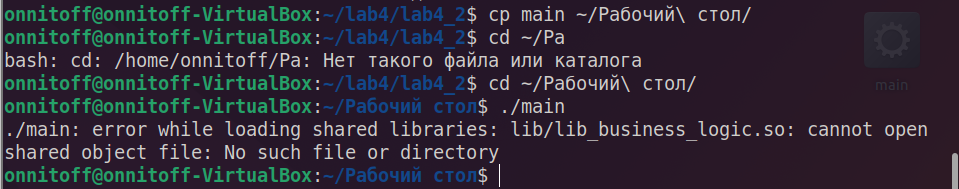
Зависимости:



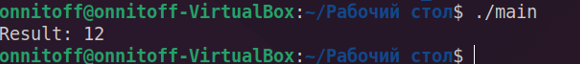
Сравнение размеров файлов:



Перенос и выполнение на рабочем столе:







**Задание 3**

Произвести рефакторинг проекта предыдущей лабораторной работы по автосборке проекта: вынесите код функций бизнес логики в отдельную динамическую библиотеку с динамическим вызовом. Для автосборки проекта используйте утилиту make, обязательно. При помощи программы ldd проверим зависимости, образовавшиеся между библиотеками. Сравните сумму размеров объектных файлов проекта и исполняемого файла. Результаты отобразите скриншотами. Копию исполняемого файла расположите на «Рабочем столе», запустите программу. Какой будет результат? Если возникла проблема, решите ее и поясните каким способом. Сравните результаты с предыдущими упражнения, сделайте вывод.

Листинг кода:

Business\_logic.cpp:

#include "business\_logic.h"

int calculateResult(int a, int b) {

return a + b;

}

Business\_logic2.cpp:

#include <iostream>

#include "business\_logic2.h"

using namespace std;

void PrintResult(int result) {

cout << "Result: " << result << endl;

}

Main.cpp:

#include <iostream>

#include <dlfcn.h>

typedef void (\*PrintResultFunction)(int result);

typedef int (\*CalculateResultFunction)(int a, int b);

int main() {

void \*lib = dlopen("lib/lib\_business\_logic.so", RTLD\_LAZY);

if (!lib) {

std::cerr << "Error opening library: " << dlerror() << std::endl;

return 1;

}

PrintResultFunction printResultFunction = (PrintResultFunction)dlsym(lib, "PrintResult");

if (!printResultFunction) {

std::cerr << "Error loading PrintResult function: " << dlerror() << std::endl;

dlclose(lib);

return 1;

}

CalculateResultFunction calculateResultFunction = (CalculateResultFunction)dlsym(lib, "calculateResult");

if (!calculateResultFunction) {

std::cerr << "Error loading calculateResult function: " << dlerror() << std::endl;

dlclose(lib);

return 1;

}

// Вызов функций

int result = calculateResultFunction(10, 5);

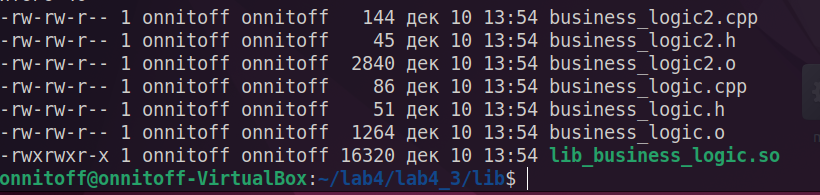
printResultFunction(result);

dlclose(lib);

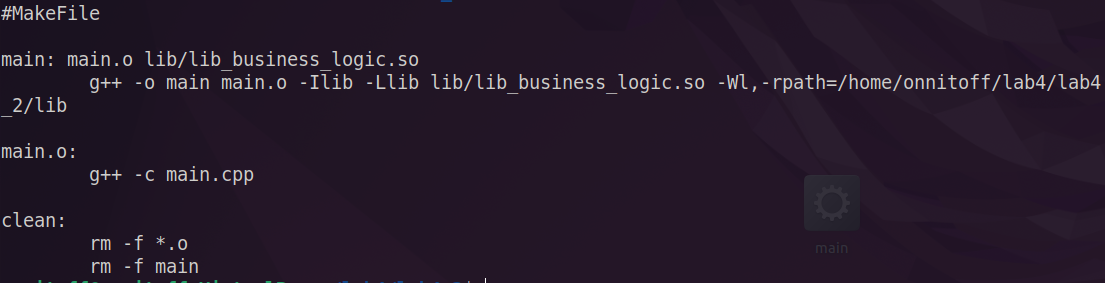
return 0;

}

Создание динамической библиотки:



MakeFile:



Работоспособность:

