Пермский филиал федерального государственного автономного образовательного учреждения высшего образования  
«Национальный исследовательский университет   
Высшая школа экономики»

*Факультет социально-экономических и компьютерных наук*

Мусихин Данил Михайлович

**Создание собственной библиотеки**

*Лабораторная работа №6*

студента образовательной программы «Разработка информационных систем для бизнеса» по направлению подготовки *09.03.04 Программная инженерия*

Руководитель, к.т.н., Доцент кафедры ИТБ.

Л.Н. Лядова

Пермь, 2023 год

**Оглавление**

[1 Постановка задачи 4](#_Toc152166223)

[2 Анализ 5](#_Toc152166224)

[3 Код 9](#_Toc152166241)

# Постановка задачи

Цель выполнения – получение навыков разработки библиотек и их подключения с использованием неявного и явного связывания.

Используя руководство по созданию и использованию библиотек, доступное по ссылке:

[**https://learn.microsoft.com/ru-ru/cpp/build/dlls-in-visual-cpp?view=msvc-170**](https://learn.microsoft.com/ru-ru/cpp/build/dlls-in-visual-cpp?view=msvc-170)

необходимо разработать и подключить библиотеку, а также вызвать разработанную функцию, включённую в библиотеку, используя неявное и явное связывание.

Функция может решать любую задачу (как вариант – вывод сообщений на экран и возврат любого кода завершения, значение которого определяется при выполнении функции).

# Анализ

**Создание и подключение библиотеки:**

1. Создать файл.cpp с библиотекой
2. Создать файл.h для использования библиотеки
3. Скомпилировать библиотеку в dll: g++ -shared -o файл.dll файл.cpp -Wl,--out-implib,libexample.a
4. Для работы в vs code использовать launch и tasks для загрузки библиотеки для компиляции
5. Скомпилировать код уже в библиотеке.

**Разница между явным и неявным связываниями:**

Разница между явным (explicit) и неявным (implicit) связываниями в контексте динамической загрузки библиотек в C++ заключается в том, как происходит связывание функций из библиотеки с программой.

1. Неявное связывание (Implicit Linking):

* Определение на этапе компиляции: Функции из библиотеки связываются с программой на этапе компиляции.
* Статическое связывание: используется ключевое слово `extern "C"` для объявления функций из библиотеки в программе.
* Преимущества: простота использования, но требует наличия библиотеки на этапе компиляции.

1. Явное связывание (Explicit Linking):

* Определение на этапе выполнения: Функции из библиотеки связываются с программой динамически, на этапе выполнения программы.
* Динамическое связывание: используются функции из библиотеки Windows API, такие как `LoadLibrary` и `GetProcAddress` для загрузки библиотеки и получения указателя на функцию.
* Преимущества: позволяет избежать статической зависимости от библиотеки на этапе компиляции.

Оба способа имеют свои преимущества и недостатки, и выбор между ними зависит от конкретных требований проекта. Явное связывание предоставляет больше гибкости, но также требует более подробной обработки ошибок и управления памятью.

# Код

## Библиотека

#include <iostream>

extern "C" int showMessage() {

    std::cout << "Hello from DLL!" << std::endl;

    return 52;

}

## Для использования библиотеки

#ifndef LIB\_H

#define LIB\_H

#include <iostream>

extern "C" int showMessage();

#endif // LIB\_H

## Использование

#include <iostream>

#include <windows.h>

#include "lib.h"

typedef int (\*ShowMessageFunction)();

int main() {

    // Неявное связывание

    int resultImplicit = showMessage();

    std::cout << "Returned implicitly: " << resultImplicit << std::endl;

    // Явное связывание

    HMODULE lib = LoadLibraryA("lib.dll");

    if (!lib) {

        std::cerr << "Cannot open library." << std::endl;

        return 1;

    }

    // Получение указателя на функцию

    ShowMessageFunction showMessageExplicit = reinterpret\_cast<ShowMessageFunction>(GetProcAddress(lib, "showMessage"));

    if (!showMessageExplicit) {

        std::cerr << "Cannot load symbol showMessage." << std::endl;

        FreeLibrary(lib);

        return 1;

    }

    // Вызов функции

    int resultExplicit = showMessageExplicit();

    std::cout << "Returned explicitly: " << resultExplicit << std::endl;

    FreeLibrary(lib);

    return 0;

}