***Статические библиотеки: достоинства и недостатки, создание и установка.***

***Разработать программу с использование словаря. Программа получает на вход N пар синонимов. Каждая строка содержит ровно два слова синонима через пробел. После всех пар слов следует слово, для которого требуется определить синоним.***

**Статические библиотеки** — это коллекции объектных файлов (скомпилированного кода), которые связаны (merged) с исполняемым файлом во время компиляции. Это означает, что код библиотеки становится частью исполняемого файла. В отличие от динамических библиотек, статические библиотеки не требуют присутствия отдельного файла библиотеки во время выполнения программы.

**Достоинства статических библиотек:**

Производительность: Поскольку код библиотеки интегрируется непосредственно в исполняемый файл, нет накладных расходов на поиск и загрузку библиотеки во время выполнения, что может повысить производительность.

Простота распространения: Исполняемый файл со статической библиотекой не зависит от наличия отдельных файлов библиотеки, что упрощает распространение программы.

Отсутствие конфликтов версий: При использовании статических библиотек исключены проблемы с несовместимостью версий, поскольку код библиотеки уже интегрирован в исполняемый файл.

Самодостаточность: Исполняемый файл полностью самодостаточен и не требует внешних зависимостей (за исключением системных библиотек).

**Недостатки статических библиотек:**

Увеличение размера исполняемого файла: Код статической библиотеки включается в исполняемый файл, что может существенно увеличить его размер, особенно если используется несколько больших библиотек.

Проблемы с обновлением: При обновлении статической библиотеки необходимо перекомпилировать и перераспространить все исполняемые файлы, которые ее используют.

Дублирование кода: Если несколько программ используют одну и ту же статическую библиотеку, код библиотеки будет скопирован в каждый исполняемый файл, что приведет к избыточному использованию памяти и дискового пространства.

Сложность отладки: Отладка статического кода, встроенного в исполняемый файл, может быть несколько сложнее, чем отладка кода, находящегося в отдельной динамической библиотеке.

**Создание и установка статической библиотеки в C++**

1. Создание исходных файлов библиотеки (.h, .cpp):

Предположим, у нас есть простая библиотека с двумя файлами:

mylib.h (заголовочный файл):

#ifndef MYLIB\_H

#define MYLIB\_H

int add(int a, int b);

int multiply(int a, int b);

#endif

mylib.cpp (исходный файл):

#include "mylib.h"

int add(int a, int b) {

return a + b;

}

int multiply(int a, int b) {

return a \* b;

}

**Добавление класса в статическую библиотеку**

Чтобы создать файл заголовка для нового класса откройте контекстное меню проекта MathLibrary в обозревателе решений, а затем выберите Добавить>Новый элемент.

В диалоговом окне Добавление нового элемента выберите пункт Visual C++>Код. В центральной области выберите Заголовочный файл (.h). Укажите имя для файла заголовка, например MathLibrary.h, и нажмите кнопку Добавить. Отобразится почти пустой файл заголовка.

Добавьте объявление класса с именем Arithmetic для выполнения обычных арифметических операций, таких как сложение, вычитание, умножение и деление. Код должен выглядеть примерно так:

**C++**

// MathLibrary.h

#pragma once

namespace MathLibrary

{

class Arithmetic

{

public:

// Returns a + b

static double Add(double a, double b);

// Returns a - b

static double Subtract(double a, double b);

// Returns a \* b

static double Multiply(double a, double b);

// Returns a / b

static double Divide(double a, double b);

};

}

Чтобы создать исходный файл для нового класса, откройте контекстное меню проекта MathLibrary в обозревателе решений, а затем выберите Добавить>Новый элемент.

В диалоговом окне Добавление нового элемента в центральной области выберите Файл C++ (.cpp). Укажите имя исходного файла, например MathLibrary.cpp, и нажмите кнопку Добавить. Отобразится пустой исходный файл.

Используйте этот исходный файл для реализации функций класса Arithmetic. Код должен выглядеть примерно так:

**C++**

// MathLibrary.cpp

// compile with: cl /c /EHsc MathLibrary.cpp

// post-build command: lib MathLibrary.obj

#include "MathLibrary.h"

namespace MathLibrary

{

double Arithmetic::Add(double a, double b)

{

return a + b;

}

double Arithmetic::Subtract(double a, double b)

{

return a - b;

}

double Arithmetic::Multiply(double a, double b)

{

return a \* b;

}

double Arithmetic::Divide(double a, double b)

{

return a / b;

}

}

Чтобы выполнить сборку статической библиотеки, выберите в строке меню команду Сборка>Собрать решение. В результате будет создана статическая библиотека MathLibrary.lib, которая может использоваться другими программами.

**Использование функциональности из статической библиотеки в приложении**

Для использования математических процедур из статической библиотеки необходимо сослаться на эту библиотеку. В обозревателе решений откройте контекстное меню проекта MathClient, а затем выберите команду Добавить>Ссылка.

В диалоговом окне Добавление ссылки перечислены библиотеки, на которые можно создать ссылку. На вкладке Проекты перечислены проекты текущего решения и все библиотеки, на которые они ссылаются. На вкладке Проекты установите флажок MathLibrary, а затем нажмите кнопку ОК.

Для создания ссылки на файл заголовка MathLibrary.h необходимо изменить путь к каталогам включаемых файлов. В обозревателе решений щелкните правой кнопкой мыши проект MathClient, чтобы открыть контекстное меню. Выберите пункт Свойства, чтобы открыть диалоговое окно Страницы свойств MathClient.

В диалоговом окне Страницы свойств MathClient в раскрывающемся списке Конфигурация выберите пункт Все конфигурации. В раскрывающемся списке Платформа выберите пункт Все платформы.

Перейдите на страницу свойств Свойства конфигурации>C/C++>Общие. В свойстве Дополнительные каталоги включаемых файлов укажите путь к каталогу MathLibrary или найдите этот каталог.

**Чтобы найти путь к каталогу, выполните указанные ниже действия.**

Откройте раскрывающийся список значений свойства Дополнительные каталоги включаемых файлов, а затем выберите Изменить.

В диалоговом окне Дополнительные каталоги включаемых файлов дважды щелкните в верхней части текстового поля. Нажмите кнопку с многоточием (...) в конце строки.

В диалоговом окне Выбор каталога перейдите на уровень вверх и выберите каталог MathLibrary. Затем нажмите кнопку Выбрать папку, чтобы сохранить выбор.

В диалоговом окне Дополнительные каталоги включаемых файлов нажмите кнопку ОК.

В диалоговом окне Страницы свойств нажмите кнопку OK, чтобы сохранить изменения в проекте.

Теперь в этом приложении можно использовать класс Arithmetic, включив в код заголовок #include "MathLibrary.h". Замените содержимое MathClient.cpp на следующий код:

**C++**

// MathClient.cpp

// compile with: cl /EHsc MathClient.cpp /link MathLibrary.lib

#include <iostream>

#include "MathLibrary.h"

int main()

{

double a = 7.4;

int b = 99;

std::cout << "a + b = " <<

MathLibrary::Arithmetic::Add(a, b) << std::endl;

std::cout << "a - b = " <<

MathLibrary::Arithmetic::Subtract(a, b) << std::endl;

std::cout << "a \* b = " <<

MathLibrary::Arithmetic::Multiply(a, b) << std::endl;

std::cout << "a / b = " <<

MathLibrary::Arithmetic::Divide(a, b) << std::endl;

return 0;

}