

ФГАОУ ВО «Санкт-Петербургский политехнический университет Петра Великого» Институт компьютерных наук и технологий

Высшая школа киберфизических систем и управления

по дисциплине Практикум по программированию

Выполнил студент группы 23533/2

Руководитель

ассистент

В.Э. Ковалевский

Санкт-Петербург 2019

30.05.2019

А.А. Тарасов

Отчет по лабораторной работе

DLL и приложение из нескольких проектов

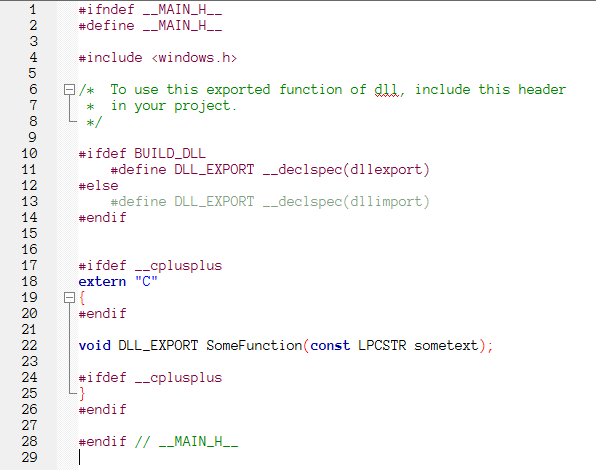
Задание:

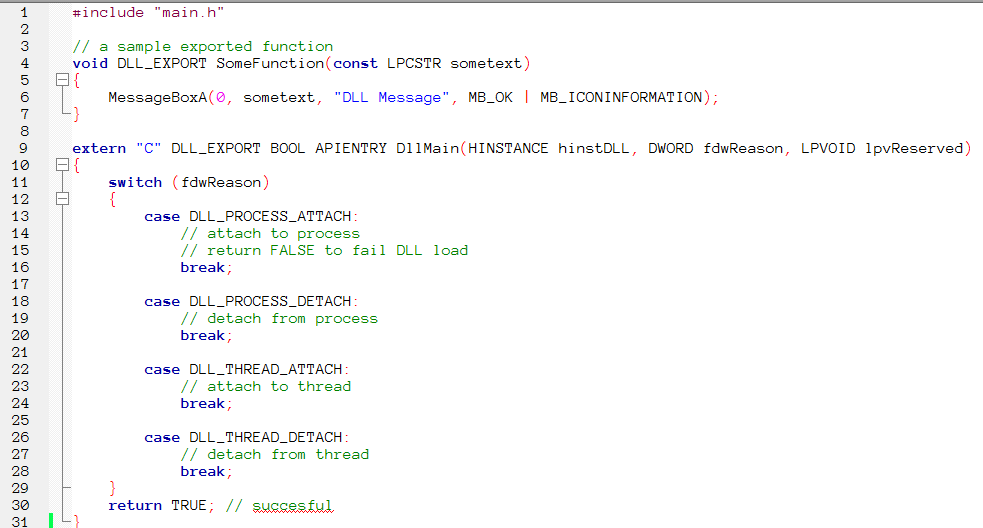
Разработать DLL файл, в который поместить необходимые в работе программы функции, которые могут применяться и в других программах.

Одна из основных особенностей Windows приложений – это их возможность подключать во время работы необходимые функции и ресурсы. Подключаемый материал размещён в специальных контейнерах – библиотеках динамической компоновки (Dynamic-Link Libraries, DLL). Функции, необходимые для работы Windows хранятся в dll файлах. Библиотеки дают преимущество модульности, помогают уменьшить размер программ, в общем случае повышают производительность.

Для начала необходимо создать DLL. В данной работе DLL создавался в среде Code::Blocks с компилятором GNU GCC. При создании нового проекта в Code::Blocks есть возможность выбрать Dynamic Link Library проект. В этом случае не придётся создавать исходные и заголовочные файлы с нуля, а остаётся только добавить нужные функции.

Ниже приведён шаблон для dll (main.h и main.cpp соответственно)





Здесь директива \_\_delspec(dllexport) используется компоновщиком для построения раздела экспорта dll. Эту директиву необходимо помещать перед объявлением или определением любой функции в DLL. Для удобного чтения имени функции в программе, написанной не на С++ нужно использовать спецификатор компоновки для языка С, который не исказит имя функции в процессе сборки.

extern “C” //далее прототип функции

Была выбрана цель доработать программу тестирования сортировок из курса вычислительной математики. Программа должна была сортировать массив quicksort и bubble sort, а затем сравнить время выполнения сортировок. Функции сортировок будут вынесены в DLL, которая в свою очередь, будет динамически подключена к программе.

Для взаимодействия моей программы с DLL (и написания DLL) используется WIN32 API, подключаемое в windows.h

После добавления кода сортировок в исходники и их сборки, получается 3 файла, один из которых - DLL. Dll нужно поместить в корень нового проекта.

Теперь явно подключим DLL. Для этого необходимо получить дескриптор подключаемого DLL. Дескриптор вернёт функция LoadLibrary(const char\*).

HMODULE hLib = LoadLibrary("TestingDLL.dll");

Теперь, когда библиотека успешно подключена, дескриптор её известен. Получим указатель на функции следующими командами:

void (\*p\_bubble) (double\*,int);

(FARPROC &)p\_bubble = GetProcAddress(hLib, "bubble");

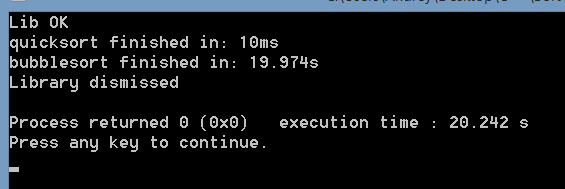
void (\*p\_qsort) (double\*, int);

(FARPROC &)p\_qsort = GetProcAddress(hLib, "quicksort");

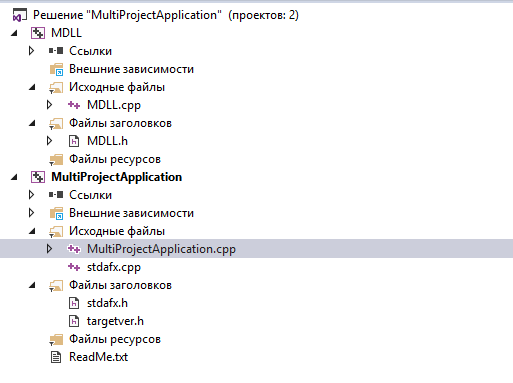
Теперь фунции можно использовать в программе. Если они не будут использоваться, есть смысл отключить файл DLL командой

FreeLibrary(hLib);

Результат сортировки массива double на 100000 элементов



Можно создать одно большое решение, которое будет содержать несколько проектов – проект с DLL и приложение, которое будет этот DLL использовать.



Такой способ организации решения помогает структурировать рабочий процесс и не отвлекаться на поиск местонахождения других проектов.

Исходный код: основная программа

#include <iostream>

#include <ctime>

#include "cstdlib"

#include "windows.h"

using namespace std;

int main()

{

HMODULE hLib = LoadLibrary("TestingDLL.dll");

void (\*p\_bubble) (double\*,int);

(FARPROC &)p\_bubble = GetProcAddress(hLib, "bubble");

void (\*p\_qsort) (double\*, int);

(FARPROC &)p\_qsort = GetProcAddress(hLib, "quicksort");

cout << "Lib OK" << endl;

srand(time(0));

double arr[100000];

for(int i =0; i< 100000; i++)

arr[i] = rand()%10000 + (double) rand()/RAND\_MAX;

int start = clock();

p\_qsort(arr, sizeof(arr)/8);

int ends = clock();

cout<<"quicksort finished in: "<< ends - start <<"ms"<<endl;

for(int i =0; i< 100000; i++)

arr[i] = rand()%10000 + (double) rand()/RAND\_MAX;

start = clock();

p\_bubble(arr, sizeof(arr)/8);

ends = clock();

cout <<"bubblesort finished in: "<< (double) (ends - start)/1000<< "s" << endl;

FreeLibrary(hLib);

cout << "Library dismissed" << endl;

}

Заголовочный файл DLL

#include <iostream>

#include <ctime>

#include "cstdlib"

#include "windows.h"

using namespace std;

int main()

{

HMODULE hLib = LoadLibrary("TestingDLL.dll");

void (\*p\_bubble) (double\*,int);

(FARPROC &)p\_bubble = GetProcAddress(hLib, "bubble");

void (\*p\_qsort) (double\*, int);

(FARPROC &)p\_qsort = GetProcAddress(hLib, "quicksort");

cout << "Lib OK" << endl;

srand(time(0));

double arr[100000];

for(int i =0; i< 100000; i++)

arr[i] = rand()%10000 + (double) rand()/RAND\_MAX;

int start = clock();

p\_qsort(arr, sizeof(arr)/8);

int ends = clock();

cout<<"quicksort finished in: "<< ends - start <<"ms"<<endl;

for(int i =0; i< 100000; i++)

arr[i] = rand()%10000 + (double) rand()/RAND\_MAX;

start = clock();

p\_bubble(arr, sizeof(arr)/8);

ends = clock();

cout <<"bubblesort finished in: "<< (double) (ends - start)/1000<< "s" << endl;

FreeLibrary(hLib);

cout << "Library dismissed" << endl;

}

cpp файл DLL

#include "main.h"

// a sample exported function

void DLL\_EXPORT SomeFunction(const LPCSTR sometext)

{

MessageBoxA(0, sometext, "DLL Message", MB\_OK | MB\_ICONINFORMATION);

}

extern "C" DLL\_EXPORT BOOL APIENTRY DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpvReserved)

{

switch (fdwReason)

{

case DLL\_PROCESS\_ATTACH:

// attach to process

// return FALSE to fail DLL load

break;

case DLL\_PROCESS\_DETACH:

// detach from process

break;

case DLL\_THREAD\_ATTACH:

// attach to thread

break;

case DLL\_THREAD\_DETACH:

// detach from thread

break;

}

return TRUE; // succesful

}

extern "C" DLL\_EXPORT void bubble(double\* array, int quantity)

{

double temp;

bool ch;

while (quantity != 1)

{

ch = 0;

for (int i = 1; i<quantity; i++)

{

if (array[i]<array[i - 1])

{

temp = array[i];

array[i] = array[i - 1];

array[i - 1] = temp;

ch = 1;

}

}

quantity--;

if (ch == 0) break;

}

}

extern "C" DLL\_EXPORT

void quicksort(double\* array, long int quantity)

{

//quantity--;

double temp, p;

int i = 0, j = quantity - 1;

p = array[(quantity - 1) / 2];

do

{

while (array[i] < p)

i++;

while (array[j]>p)

j--;

if (i <= j)

{

temp = array[i];

array[i] = array[j];

array[j] = temp;

i++;

j--;

}

} while (i <= j);

if (j>0) quicksort(array, j + 1);

if (quantity>i) quicksort(array + i, quantity - i);

}