

# Лабораторна робота № 6

Виконав:

студент ПМІ-44

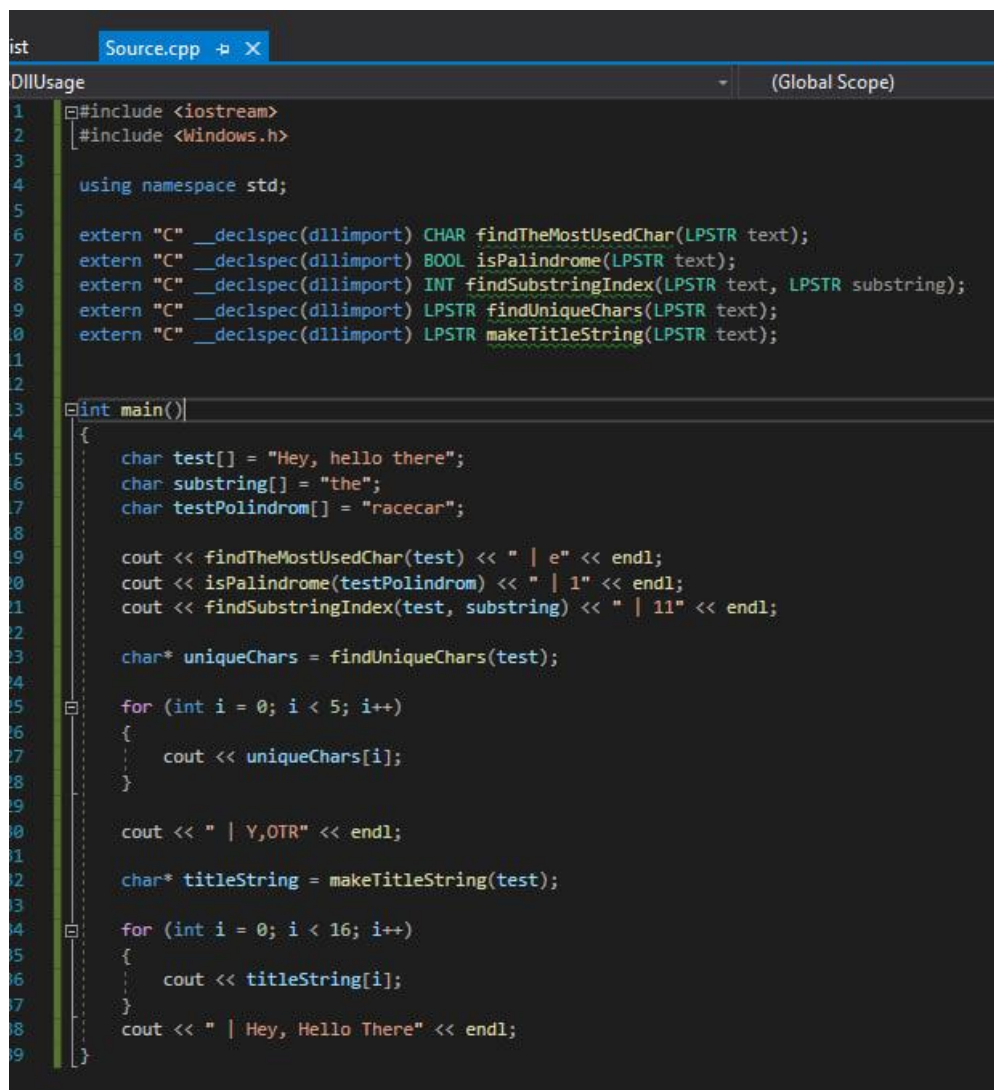
Копина Ілля

**Тема:** Використання бібліотеки DLL в програмах C++

## Хід Роботи:

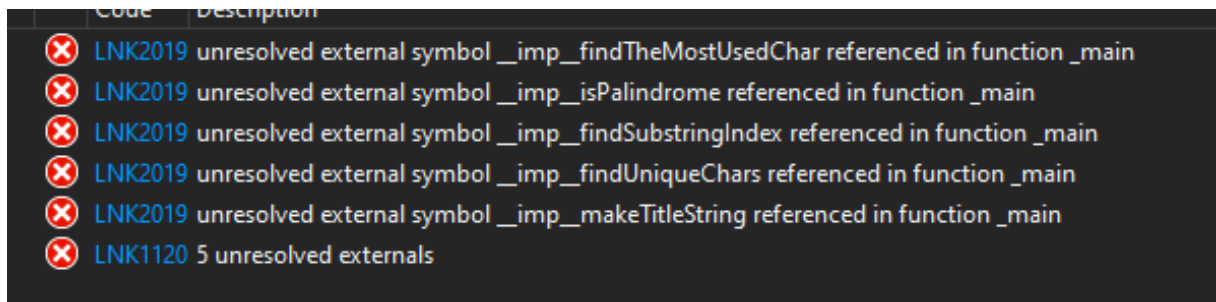
За основу DLL була використана уже створена в контексті минулої лабораторної DLL бібліотека. А саме: будуть використовуватись **MyLabDll.dll** та **MyLabDll.lib** файли методами неявного та явного зв'язування.

Розпочнемо з **Неявного зв'язування**. Для початку був створений Пустий Проект куди були імпортовані прототипи функцій з DLL, а також була перенесена головна функція з тестами з минулої лабораторної роботи.



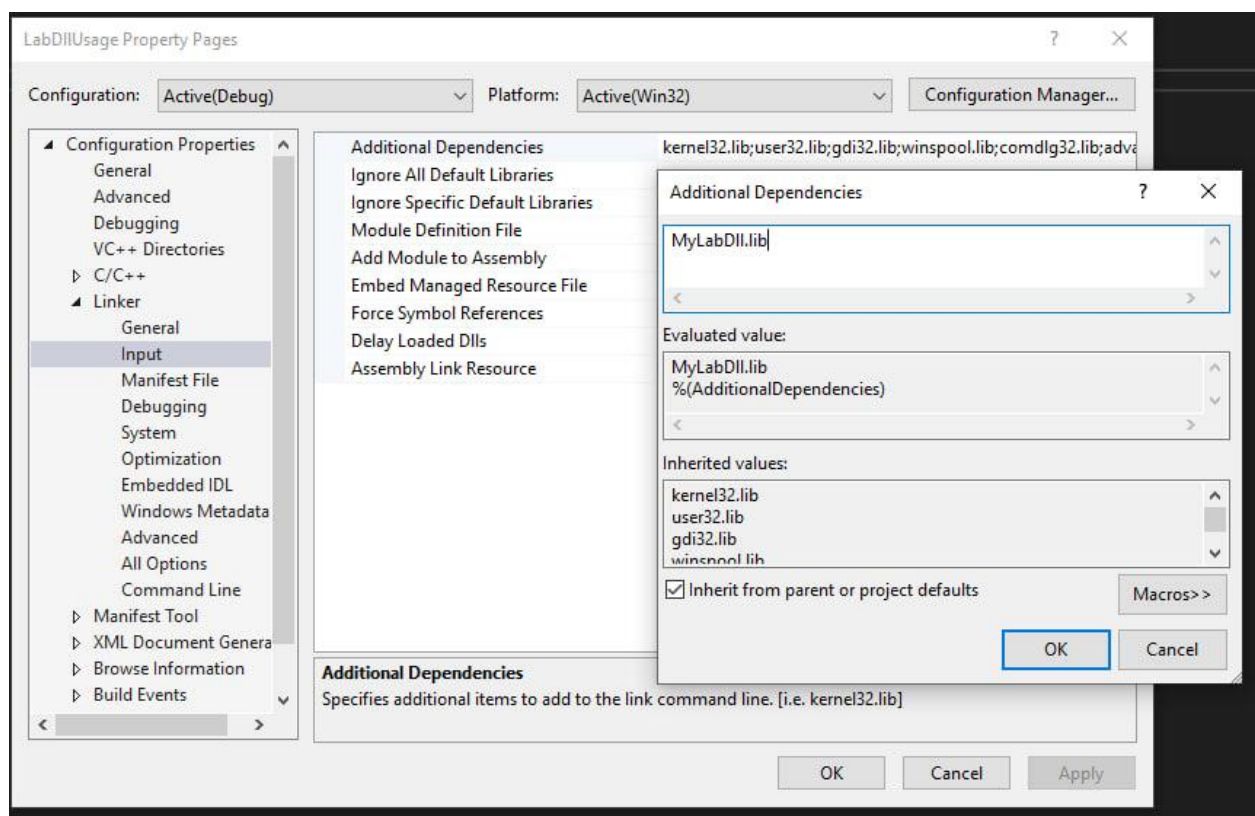
```
1 #include <iostream>
2 #include <Windows.h>
3
4 using namespace std;
5
6 extern "C" __declspec(dllimport) CHAR findTheMostUsedChar(LPSTR text);
7 extern "C" __declspec(dllimport) BOOL isPalindrome(LPSTR text);
8 extern "C" __declspec(dllimport) INT findSubstringIndex(LPSTR text, LPSTR substring);
9 extern "C" __declspec(dllimport) LPSTR findUniqueChars(LPSTR text);
10 extern "C" __declspec(dllimport) LPSTR makeTitleString(LPSTR text);
11
12
13 int main()
14 {
15     char test[] = "Hey, hello there";
16     char substring[] = "the";
17     char testPolindrom[] = "racecar";
18
19     cout << findTheMostUsedChar(test) << " | e" << endl;
20     cout << isPalindrome(testPolindrom) << " | 1" << endl;
21     cout << findSubstringIndex(test, substring) << " | 11" << endl;
22
23     char* uniqueChars = findUniqueChars(test);
24
25     for (int i = 0; i < 5; i++)
26     {
27         cout << uniqueChars[i];
28     }
29
30     cout << " | Y,OTR" << endl;
31
32     char* titleString = makeTitleString(test);
33
34     for (int i = 0; i < 16; i++)
35     {
36         cout << titleString[i];
37     }
38     cout << " | Hey, Hello There" << endl;
39 }
```

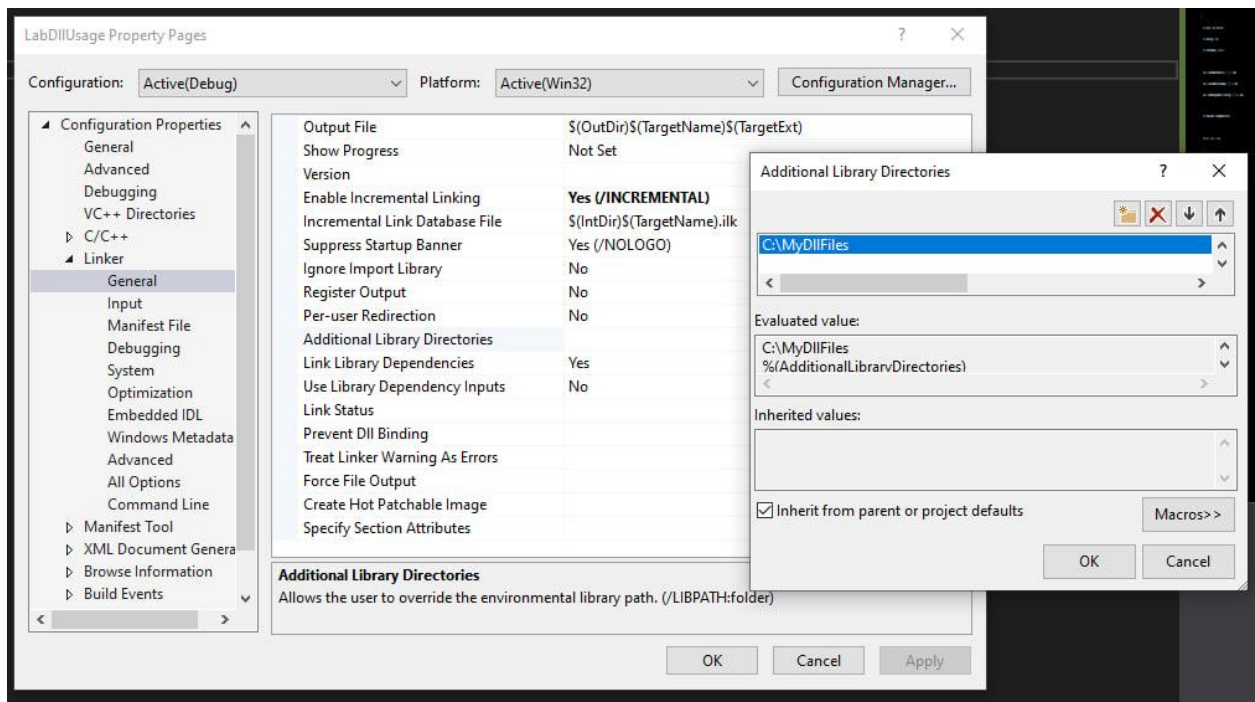
На цьому етапі помилок у самому коді немає, проте після запуску компілятор покаже декілька помилок зв'язані з поганими зовнішніми посиланнями на функції.



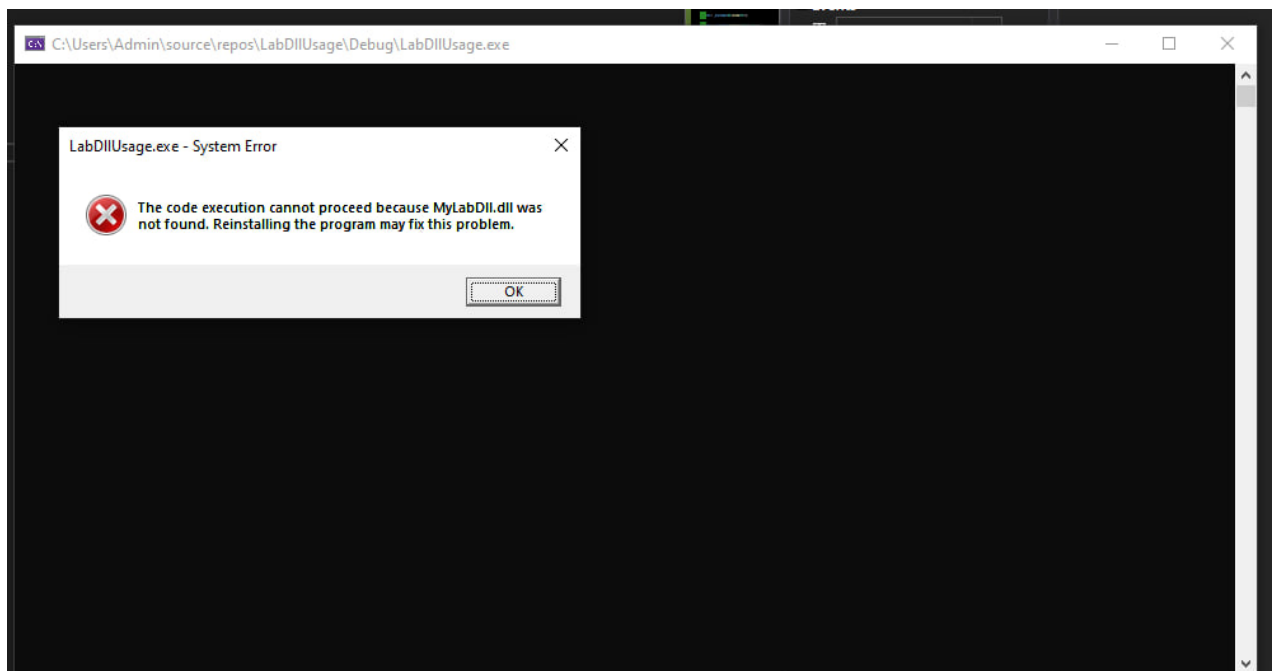
Причиною цих помилок є те, що компонувальнику потрібний файл **MyLabDll.lib**. Отже для виконання налаштування для пошуку цього файлу були виконані наступні дії:

- Зайшли у вікно Property Pages через Visual Studio (Project → Properties).
- На лівій панелі вибрали Configuration Properties → Linker → Input та на панелі властивостей обрали Additional Dependencies де і додали наш файл.
- Після цього зайшли у вкладку Configuration Properties → Linker → General та на панелі властивостей обрали Additional Dependencies де записали шлях до розташування файлу імпорту (У моєму випадку файл був перенесений у папку C:\MyDllFiles).





Тепер ми можемо запуснути програму без помилок від компілятора, проте після запуску програма вилітає з наступною помилкою:



Це говорить про те, що під час виконання операційна система намагалася знайти саме бібліотеку **MyLabDII.dll**, але не змогла. Найпростіший спосіб уникнути цю проблему – це помістити цей файл у папку де знаходиться exe файл проекту, але я вирішив додати її безпосередньо в папку Windows, що дозволить системі одразу знайти її.

This PC > OS (C:) > Windows				
	Name	Date modified	Type	Size
	bootstat	5/9/2023 4:40 PM	DAT File	66 KB
	Core	12/7/2019 11:10 AM	XML Document	30 KB
	CoreSingleLanguage	4/12/2018 2:33 AM	XML Document	35 KB
	csup	11/17/2018 4:51 AM	Text Document	1 KB
	diagerr	10/14/2020 12:49 AM	XML Document	8 KB
	diagwrn	10/14/2020 12:49 AM	XML Document	8 KB
	DirectX	5/5/2023 8:13 PM	Text Document	1 KB
	explorer	4/12/2023 1:57 PM	Application	5,127 KB
	forevermopt	3/23/2021 8:23 PM	Configuration sett...	1 KB
	HelpPane	10/13/2022 5:07 PM	Application	1,051 KB
ial	hh	12/7/2019 11:09 AM	Application	18 KB
	Info	10/22/2021 3:06 PM	XML Document	5 KB
	mib.bin	12/7/2019 11:08 AM	BIN File	43 KB
	MyLabDll.dll	5/9/2023 4:45 AM	Application exten...	63 KB
	notepad	8/10/2022 6:24 PM	Application	197 KB
	NvContainerRecovery	3/15/2022 5:43 PM	Windows Batch File	2 KB
	ODBCINST	1/25/2023 7:19 PM	Configuration sett...	1 KB

Після запуску програми бачимо результат тестів:

The screenshot shows the Visual Studio IDE with a C++ source file named 'Source.cpp' and a 'Microsoft Visual Studio Debug Console' window open. The source code includes headers for `<iostream>` and `<windows.h>`, uses the `std` namespace, and declares several external functions from a DLL: `findTheMostUsedChar`, `isPalindrome`, `findSubstringIndex`, `findUniqueChars`, and `makeTitleString`. The `main` function tests these functions with various inputs and prints the results.

The debug console output shows the following results:

```
e | e
1 | 1
11 | 11
Y,OTR | Y,OTR
Hey, Hello There | Hey, Hello There
C:\Users\Admin\source\repos\LabDllUsage\Debug\LabDllUsage.exe (process 11920) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Тепер спробуємо виконати **Явне зв'язування**. Для цього створимо ще один Пустий Проект і перенесемо туди прототипи функцій з додаванням ключового слова **typedef**. Завдяки цьому ми зможемо оголосити спеціальні типи вказівників на зовнішні функції DLL. Після цього потрібно вказати в програму шлях до файлу **MyLabDll.dll** і дати цей шлях як параметр в функцію **LoadLibrary**, яка зможе імпортувати бібліотеку у цю програму. Після цього потрібно побудувати вказівник на певну функцію завдяки **GetProcAddress**, яка може імпортувати потрібні нам функцію з DLL. Також варто додати відповідні перевірки як для бібліотеки так і для окремих функцій. У результаті код буде виглядати наступним чином:

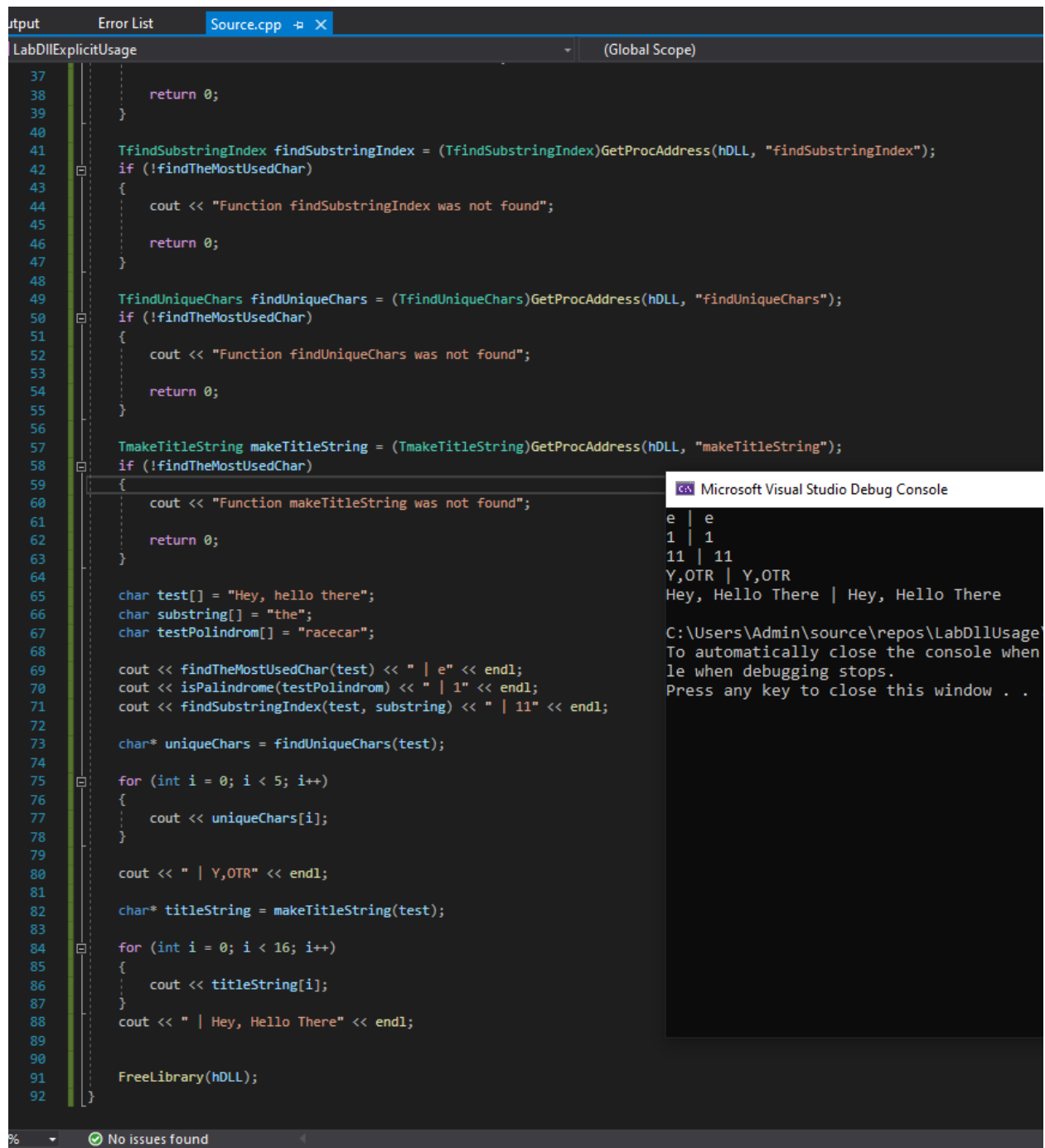
```
put      Error List      Source.cpp  -> X
LabDllExplicitUsage      (Global Scope)

4      using namespace std;
5
6      typedef CHAR (*TfindTheMostUsedChar)(LPSTR text);
7      typedef BOOL (*TisPalindrome)(LPSTR text);
8      typedef INT (*TfindSubstringIndex)(LPSTR text, LPSTR substring);
9      typedef LPSTR (*TfindUniqueChars)(LPSTR text);
10     typedef LPSTR (*TmakeTitleString)(LPSTR text);
11
12
13     int main()
14     {
15         const wchar_t dllPath[] = L"C:\\MyDllFiles\\MyLabDll.dll";
16
17         HINSTANCE hDLL = LoadLibrary(dllPath);
18         if (hDLL == NULL)
19         {
20             cout << "DLL was not found";
21
22             return 0;
23         }
24
25         TfindTheMostUsedChar findTheMostUsedChar = (TfindTheMostUsedChar)GetProcAddress(hDLL, "findTheMostUsedChar");
26         if (!findTheMostUsedChar)
27         {
28             cout << "Function findTheMostUsedChar was not found";
29
30             return 0;
31         }
32
33         TisPalindrome isPalindrome = (TisPalindrome)GetProcAddress(hDLL, "isPalindrome");
34         if (!isPalindrome)
35         {
36             cout << "Function isPalindrome was not found";
37
38             return 0;
39         }
40
41         TfindSubstringIndex findSubstringIndex = (TfindSubstringIndex)GetProcAddress(hDLL, "findSubstringIndex");
42         if (!findSubstringIndex)
43         {
44             cout << "Function findSubstringIndex was not found";
45
46             return 0;
47         }
48
49         TfindUniqueChars findUniqueChars = (TfindUniqueChars)GetProcAddress(hDLL, "findUniqueChars");
50         if (!findUniqueChars)
51         {
52             cout << "Function findUniqueChars was not found";
53
54             return 0;
55         }
56
57         TmakeTitleString makeTitleString = (TmakeTitleString)GetProcAddress(hDLL, "makeTitleString");
58         if (!makeTitleString)
59         {
60             cout << "Function makeTitleString was not found";
61         }
62     }
63 }
```

No issues found



Також потрібно не забути видалити створені вказівники за допомогою функції **FreeLibrary**. Тепер можна перенести і тести з минулої лабораторної. Після запуску бачимо, що все працює.



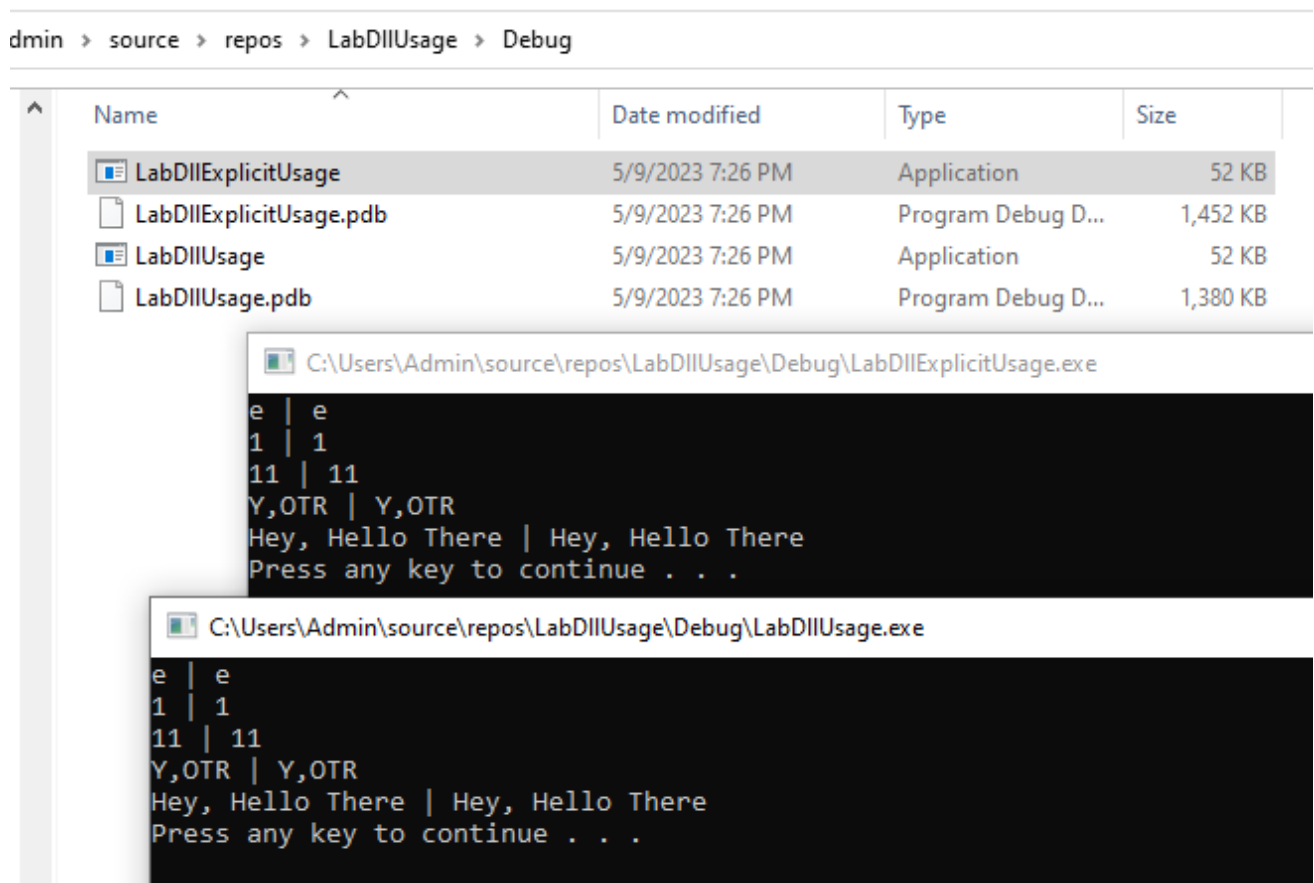
```
37
38     return 0;
39 }
40
41 TfindSubstringIndex findSubstringIndex = (TfindSubstringIndex)GetProcAddress(hDLL, "findSubstringIndex");
42 if (!findTheMostUsedChar)
43 {
44     cout << "Function findSubstringIndex was not found";
45
46     return 0;
47 }
48
49 TfindUniqueChars findUniqueChars = (TfindUniqueChars)GetProcAddress(hDLL, "findUniqueChars");
50 if (!findTheMostUsedChar)
51 {
52     cout << "Function findUniqueChars was not found";
53
54     return 0;
55 }
56
57 TmakeTitleString makeTitleString = (TmakeTitleString)GetProcAddress(hDLL, "makeTitleString");
58 if (!findTheMostUsedChar)
59 {
60     cout << "Function makeTitleString was not found";
61
62     return 0;
63 }
64
65 char test[] = "Hey, hello there";
66 char substring[] = "the";
67 char testPolindrom[] = "racecar";
68
69 cout << findTheMostUsedChar(test) << " | e" << endl;
70 cout << isPalindrome(testPolindrom) << " | 1" << endl;
71 cout << findSubstringIndex(test, substring) << " | 11" << endl;
72
73 char* uniqueChars = findUniqueChars(test);
74
75 for (int i = 0; i < 5; i++)
76 {
77     cout << uniqueChars[i];
78 }
79
80 cout << " | Y,OTR" << endl;
81
82 char* titleString = makeTitleString(test);
83
84 for (int i = 0; i < 16; i++)
85 {
86     cout << titleString[i];
87 }
88 cout << " | Hey, Hello There" << endl;
89
90
91 FreeLibrary(hDLL);
92 }
```

Microsoft Visual Studio Debug Console

```
e | e
1 | 1
11 | 11
Y,OTR | Y,OTR
Hey, Hello There | Hey, Hello There

C:\Users\Admin\source\repos\LabDllUsage
To automatically close the console when
le when debugging stops.
Press any key to close this window . .
```

Тепер спробуємо запустити наші програми з папки Debug та перевіримо, що все працює. Спочатку потрібно додати функцію **system("pause")** адже тепер середовище не зможе зупиняти процес за нас. Також варто сказати, що сам Dll файл не потрібно додавати у папку програми, адже він знаходиться у системній папці Windows і тому перша програма сама зможе його знайти, а у другій програмі ми явно вказуємо шлях до файлу.



Переконавшись, що все підключено правильно, можна використовувати бібліотеку. У нашому випадку створимо ще декілька тестів для кожної з функцій та належно оформимо результати тестів у консолі. Окрім цього отриманий тип даних у деяких функцій є типу **LPSTR**, який заповняє невикористану пам'ять різними не символьними даними і тому виводити його в консоль без використання додаткових бібліотек досить незручно. Тому була створена додаткова функція **convertLPSTR**, яка валідує символи з цього типу даних. Ця функція застосовується в тестах двох останніх функцій.

### Результат Роботи:

Після запуску програми бачимо, що усі тести пройшли і сама програма працює. Код програми є доданий до завдання

C:\Users\Admin\source\repos\LabDIIIUsage\Debug\LabDIIIUsage.exe

-----  
Tests for 'findTheMostUsedChar' function:

    Data: First test with different letters  
Expected: t  
Actual: t

    Data: what\_about\_unusual\_chars\_like\_this?  
Expected: \_  
Actual: \_

    Data: Hm? What? How?  
Expected: ?  
Actual: ?  
-----

Tests for 'isPalindrome' function:

    Data: accruent  
Expected: 0  
Actual: 0

    Data: Madam  
Expected: 0  
Actual: 0

    Data: madam  
Expected: 1  
Actual: 1

    Data: neverodddoreven  
Expected: 1  
Actual: 1  
-----

Tests for 'findSubstringIndex' function:

    Data: Hello, how are you? | are  
Expected: 11  
Actual: 11

    Data: Im doing well. How about you? | m doin  
Expected: 1  
Actual: 1

    Data: I am glad to hear that. I am a little busy | am  
Expected: 2  
Actual: 2

    Data: Ok. I wont disturb you | Iwont  
Expected: -1  
Actual: -1  
-----

Tests for 'findUniqueChars' function:

    Data: Jack jogged off, often lost  
Expected: ACKD,NLS  
Actual: ACKD,NLS

    Data: This cat is full of fur  
Expected: HCAOR  
Actual: HCAOR

Actual: JIG,FN!  
-----

Tests for 'makeTitleString' function:

    Data: Here is my topic of the day.What do you?think.  
Expected: Here Is My Topic Of The Day.What Do You?Think.  
Actual: Here Is My Topic Of The Day.What Do You?Think.

    Data: what\_about\_unusual\_chars\_like\_this? it will not Do oNly second part  
Expected: What\_About\_Unusual\_Chars\_Like\_This? It Will Not Do ONLY Second Part  
Actual: What\_About\_Unusual\_Chars\_Like\_This? It Will Not Do ONLY Second Part

    Data: Okay, what About?This!anD this  
Expected: Okay, What About?This!AnD This  
Actual: Okay, What About?This!AnD This  
-----

Press any key to continue . . .