Table of Contents

1. Instalacja [rodowiska Visual Studio Community 2019 i konß guracja
bibliotek dll napisanych w MASM 2
1.1. Instalacja 2
1.2. Utworzenie projektu z asemblerow bibliotek DLL 4
1.2.1. Utworzenie nowego projektu 5
1.2.2. Dodanie do rozwi zania cz [ci asemblerowej 7
1.2.3. Ustawienie zale|no[ci w rozwi zaniu 10
1.2.4. Zamiana biblioteki z C++ na asemblerow 12
1.2.5. Ustawienie trybu kompilacji 18
1.2.6. KoDcowa konß guracja biblioteki asemblerowej 25
1.3. Wykorzystanie biblioteki dll napisanej w asemblerze 29
1.3.1. Biblioteka asemblerowa dll 29
1.3.2. OdwoBanie do biblioteki dll z programu C++ 30
1.3.3. Zadanie do zrealizowania w ramach laboratorium 31

1.Instalacja [rodowiska Visual Stud Community 2019 i konß guracja do t bibliotek dll napisanych w MASM.

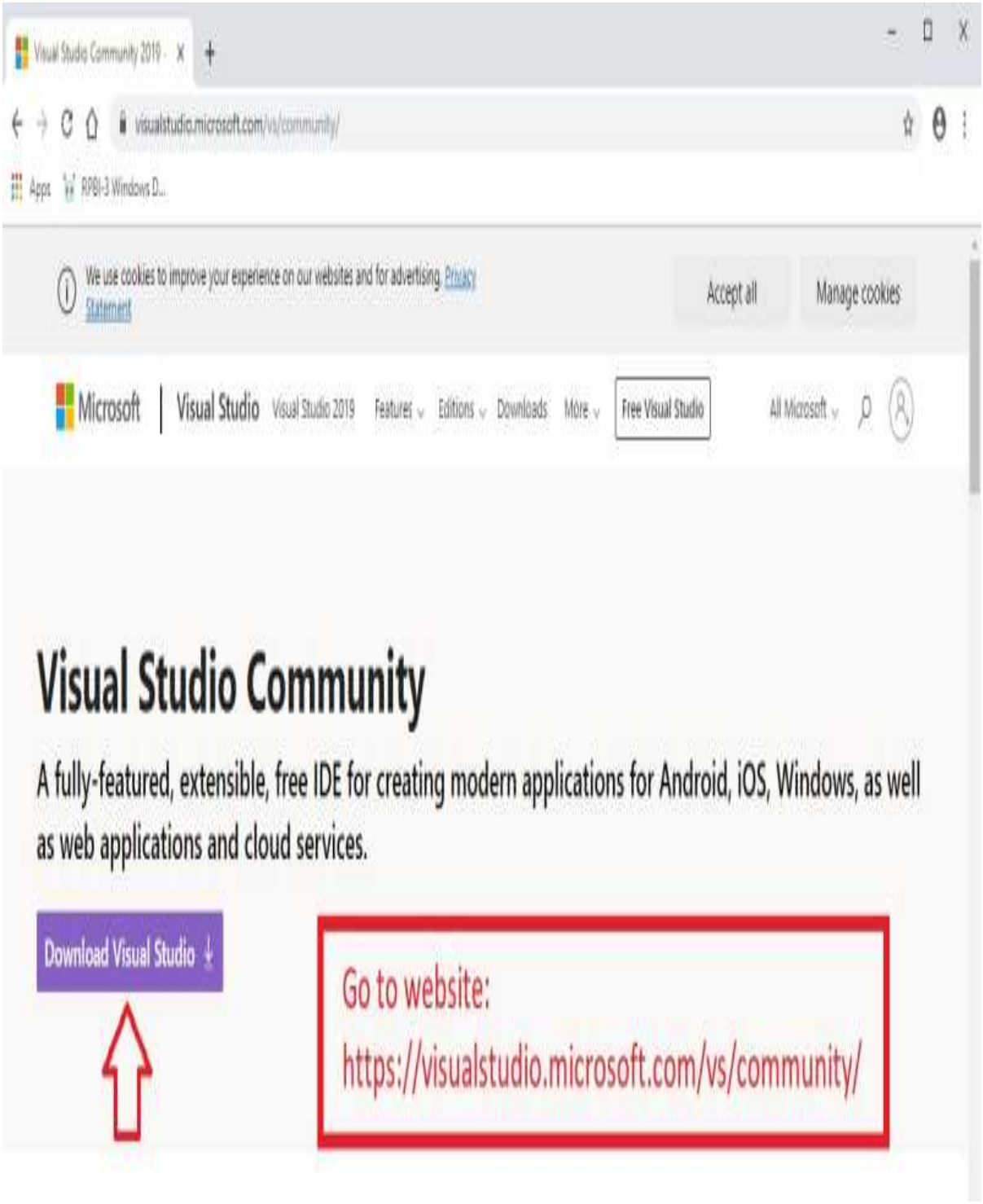
1.1Instalacja

Rozpoczynamy od [ci gni cia instalatora z Microsoft te strony

Po uruchomieniu instalatora wybieramy opcje .NET desktop developme with C++ i rozpoczynamy instalacj .

Instalacja przebiegnie automatycznie.

Po zainstalowaniu, przy pierwszym uruchomieniu [rodowiska, mo|na us



1.2 Utworzenie projektu z asemblerow bibliotek

Po zainstalowaniu mo|emy utworzy projekt, w którym gBówny program napisany w wysokiego poziomu (np. C#, C++), a biblioteka doB czana dynamicznie asemblerze.



Tworzenie bibliotek w asemblerze, wywoBywanych z pro wybranym j zyku wysokiego poziomu (HLL) jest wykorzystaniem j zyka asemblerowego.

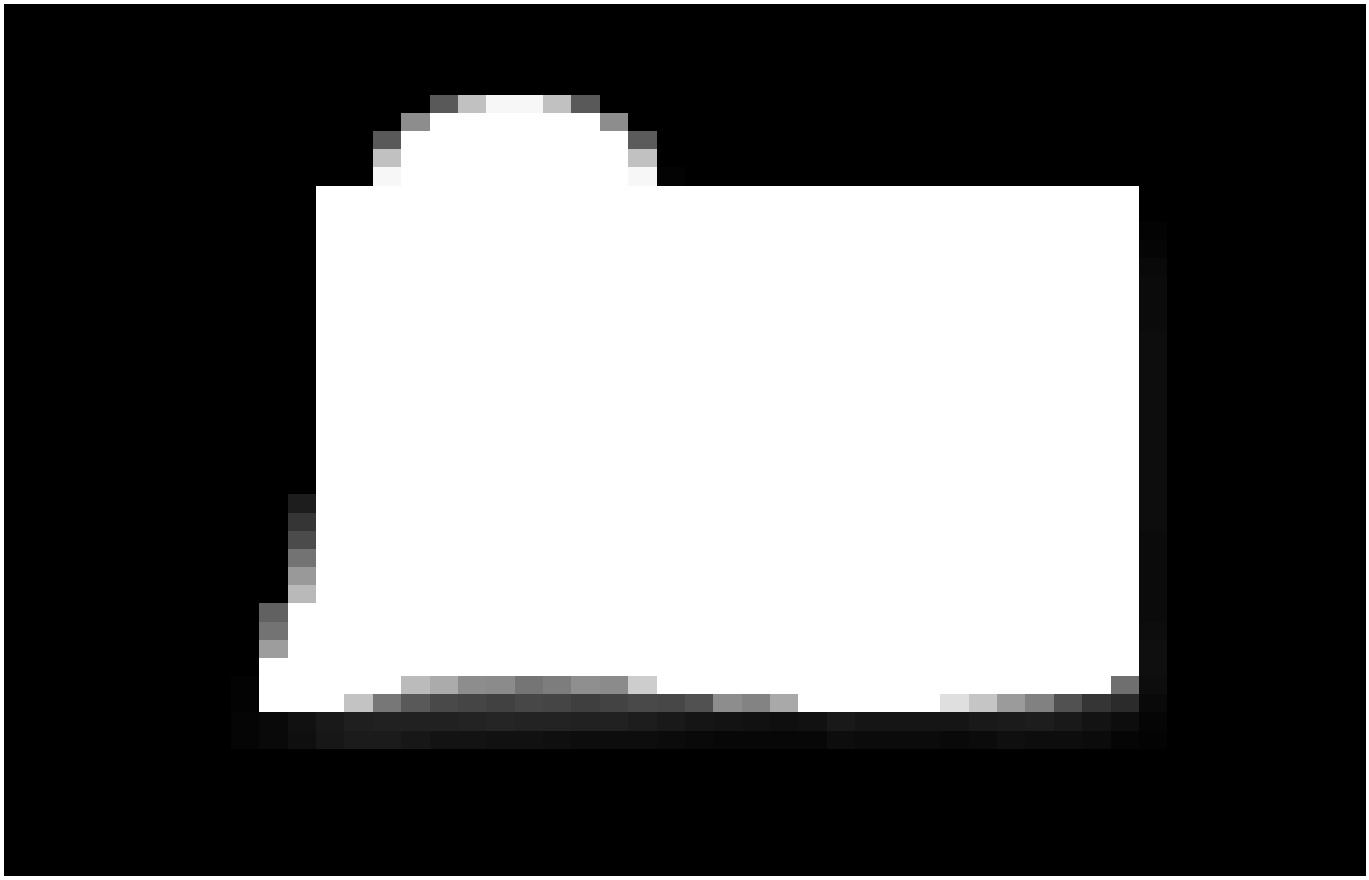
Przedstawiony w dalszej cz [ci proces konß guracji przebiega i gBównego napisanego w C++ jak i C#. Wykorzystane zrz konß guracji bibliotek DLL dla obu j zyków programowania, je drugoplanowe i nie maj wpBywu na proces konß guracji [rodowi

1.2.1 Utworzenie nowego projektu

W tym wiczeniu zajmiemy si doB czeniem biblioteki dll do programu n Konß guracj rozpoczynamy od utworzenia nowego projektu w j zyku C+



Utworzymy aplikacj konsolow , napisan w j zyku C++, przeznaczon d systemu operacyjnego Windows.



Aby utworzy projekt nale|y go nazwa . Dla tego projektu proponujemy
pozostawiamy domy[lne.

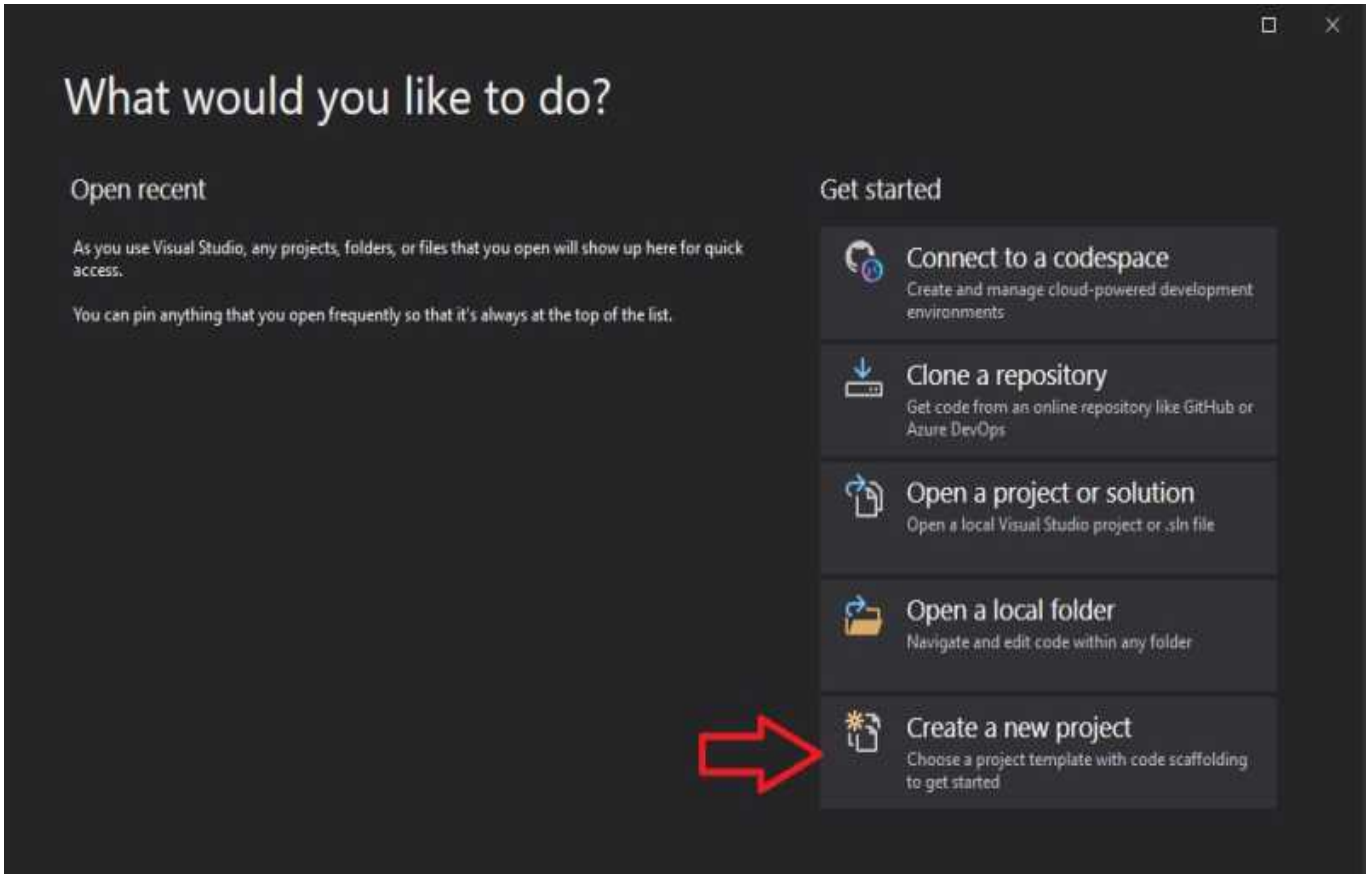


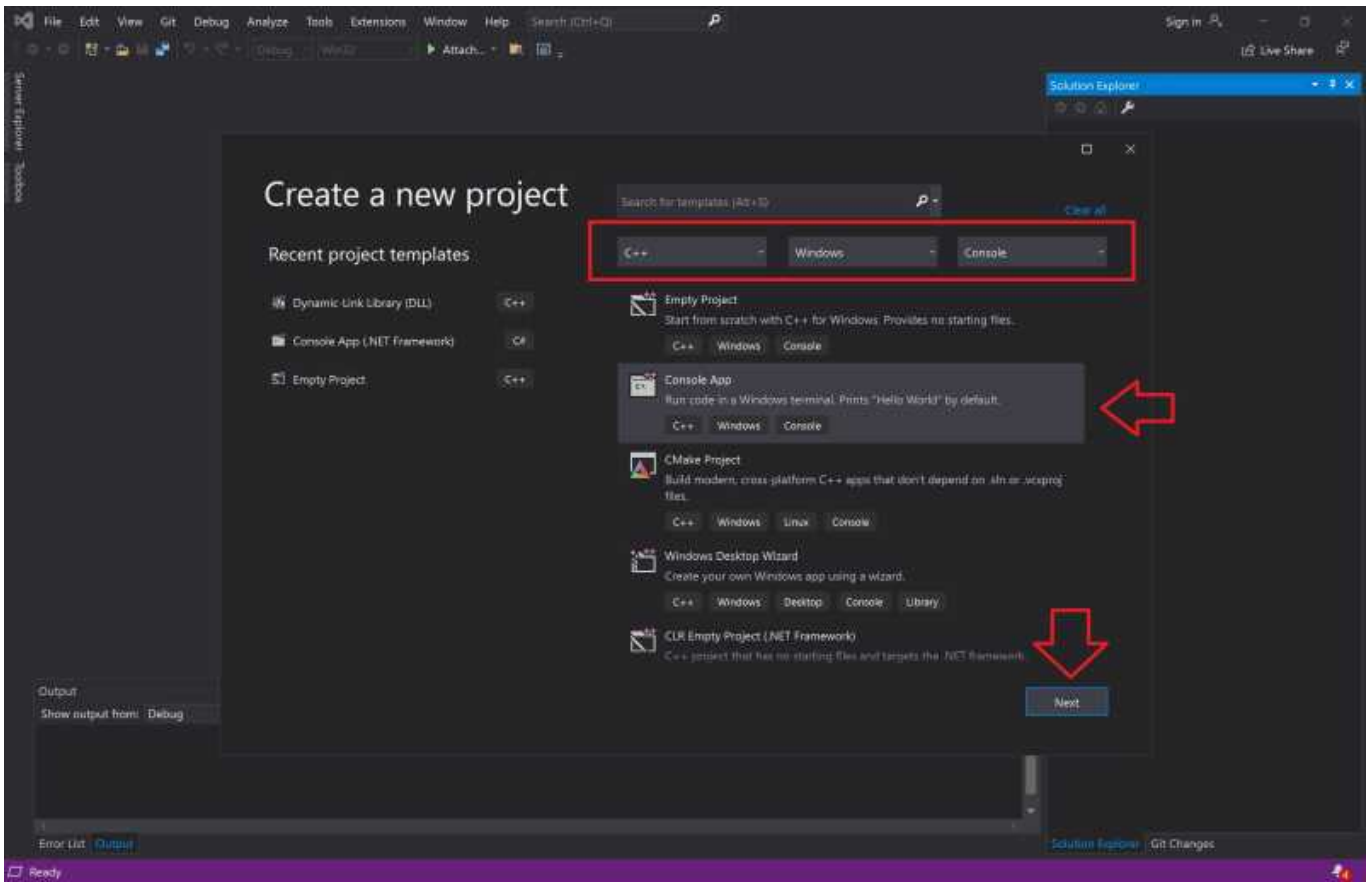


Na tym etapie zostanie utworzone rozwi zanie zawieraj programu C++. Rozwi zanie jest cz sto zwane solucj (z ang. s

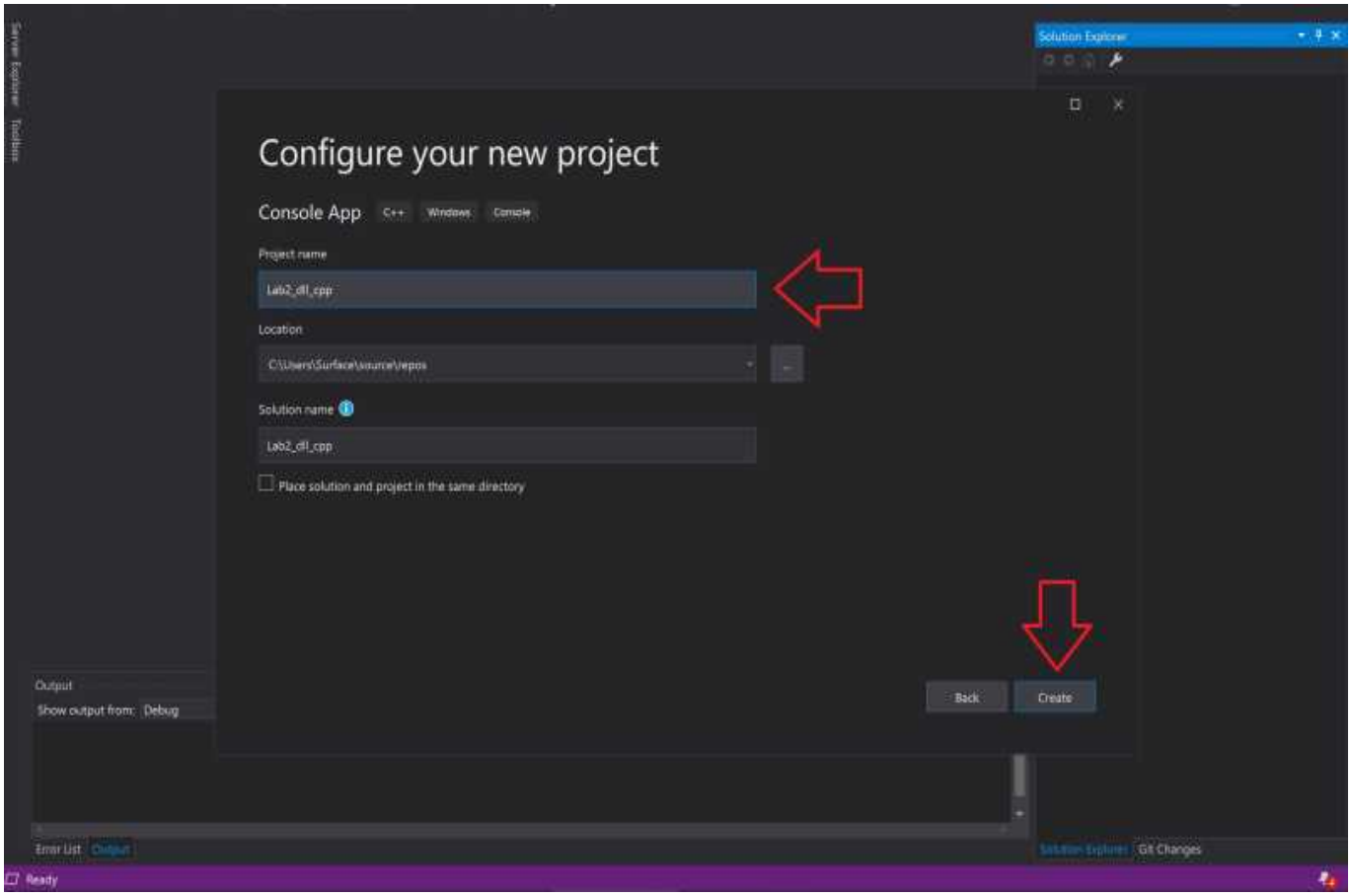
1.2.2 Dodanie do rozwi zania cz [ci asemblerowej

Aby w naszym rozwi zaniu doda bibliotek napisan w asemblerze, dod projekt. Klikaj c prawym klawiszem myszy na rozwi zaniu wybieramy do project).

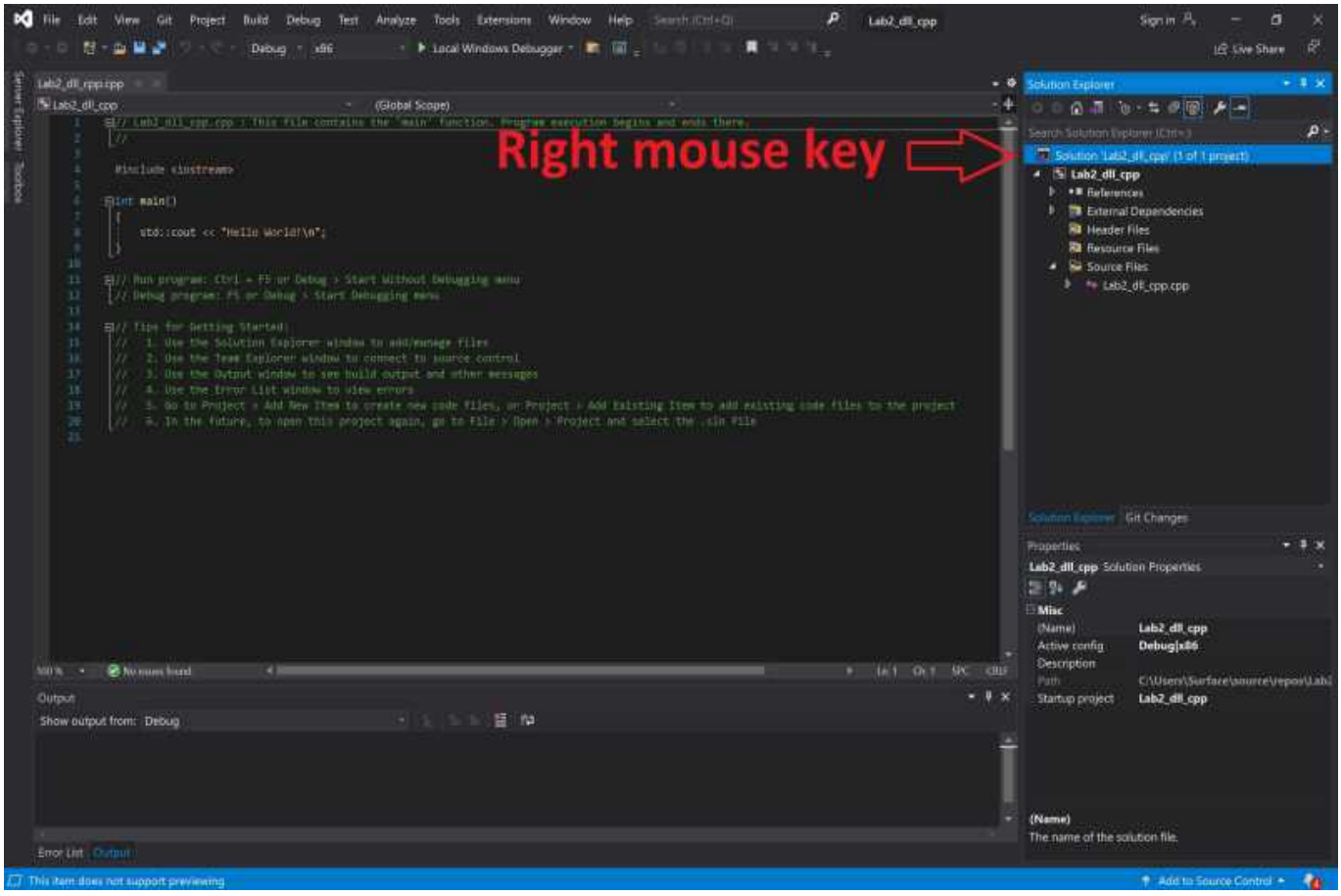




Nasz nowy projekt asemblerowy tworzymy jako projekt biblioteki (libra
pod kontrol systemu operacyjnego Windows. Z listy dost pnych opcji w
doB czan dynamicznie Dynamic-Link Library (DLL).



Nazywamy pojekt np. JaAsm i tworzymy go pozostawiaj c domy[ln [ci

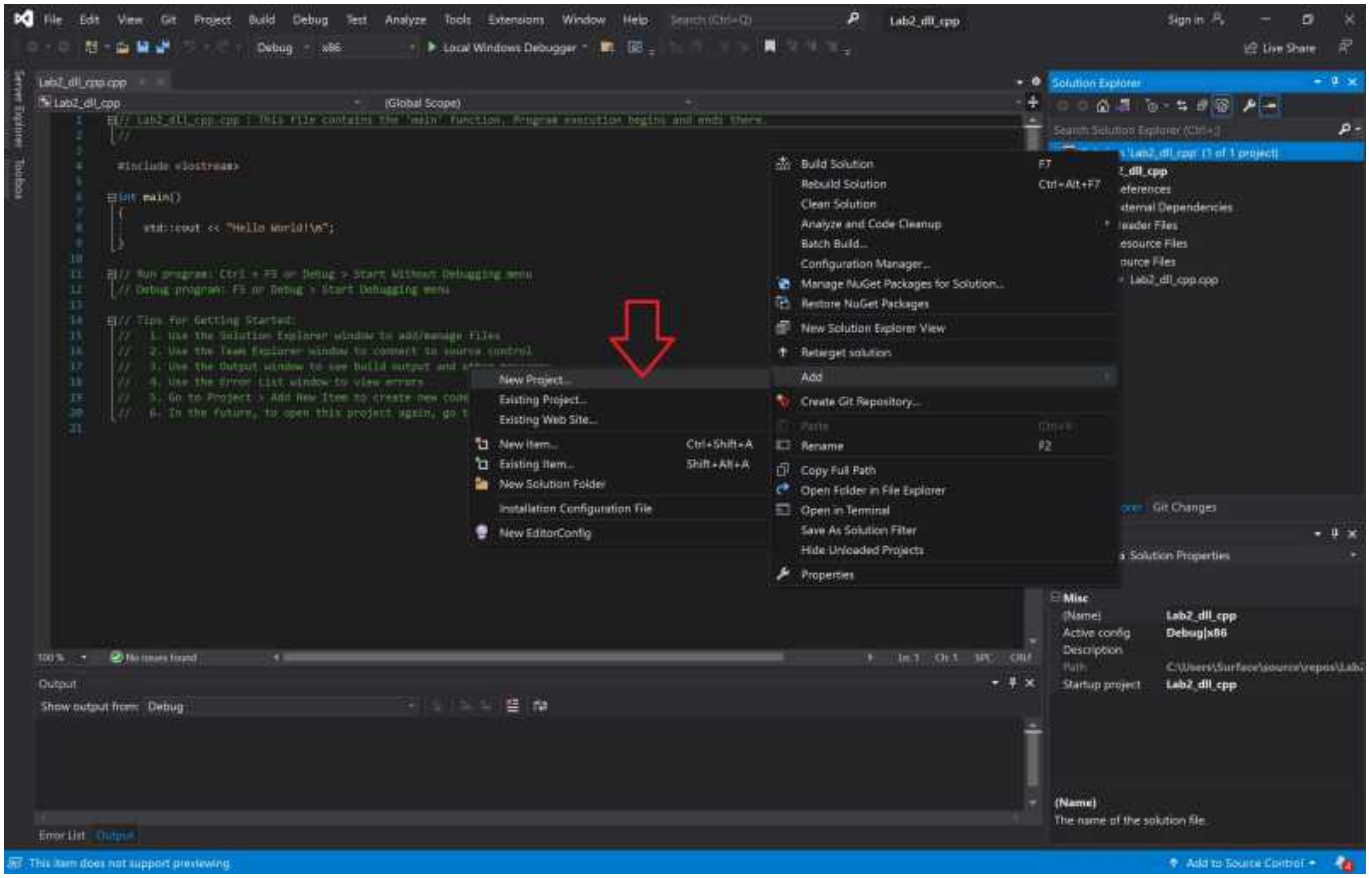


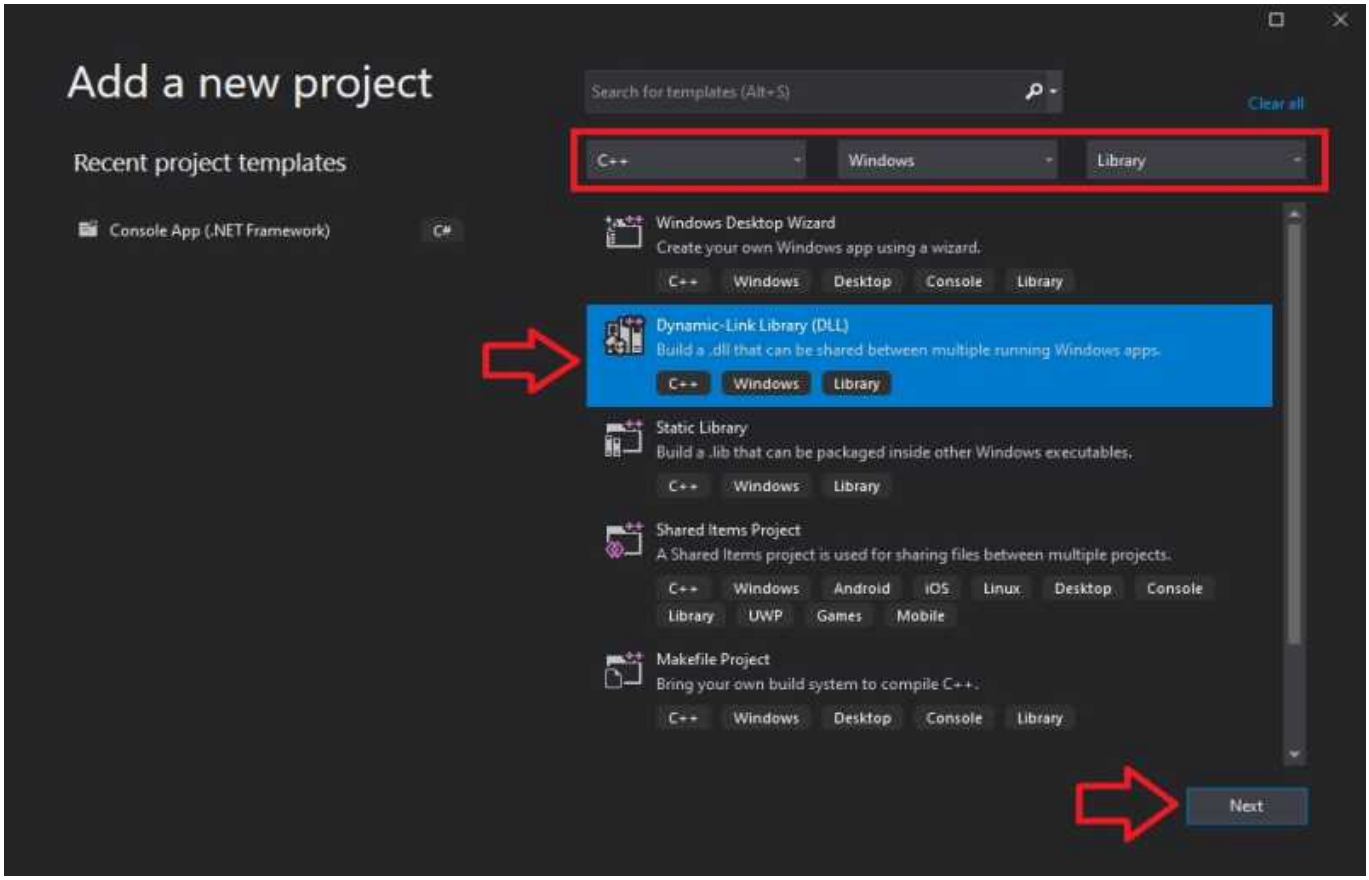


Obie cz [ci naszego rozwi zania b d umieszczone jako podfolo o nazwie zgodniej z nazw rozwi zania, w naszym przypadku tych podfolderów przyjmie nazw odpowiednich projektów, cz cz [ci napisanej w C++ i JaAsm dla cz [ci napisanej w asembl

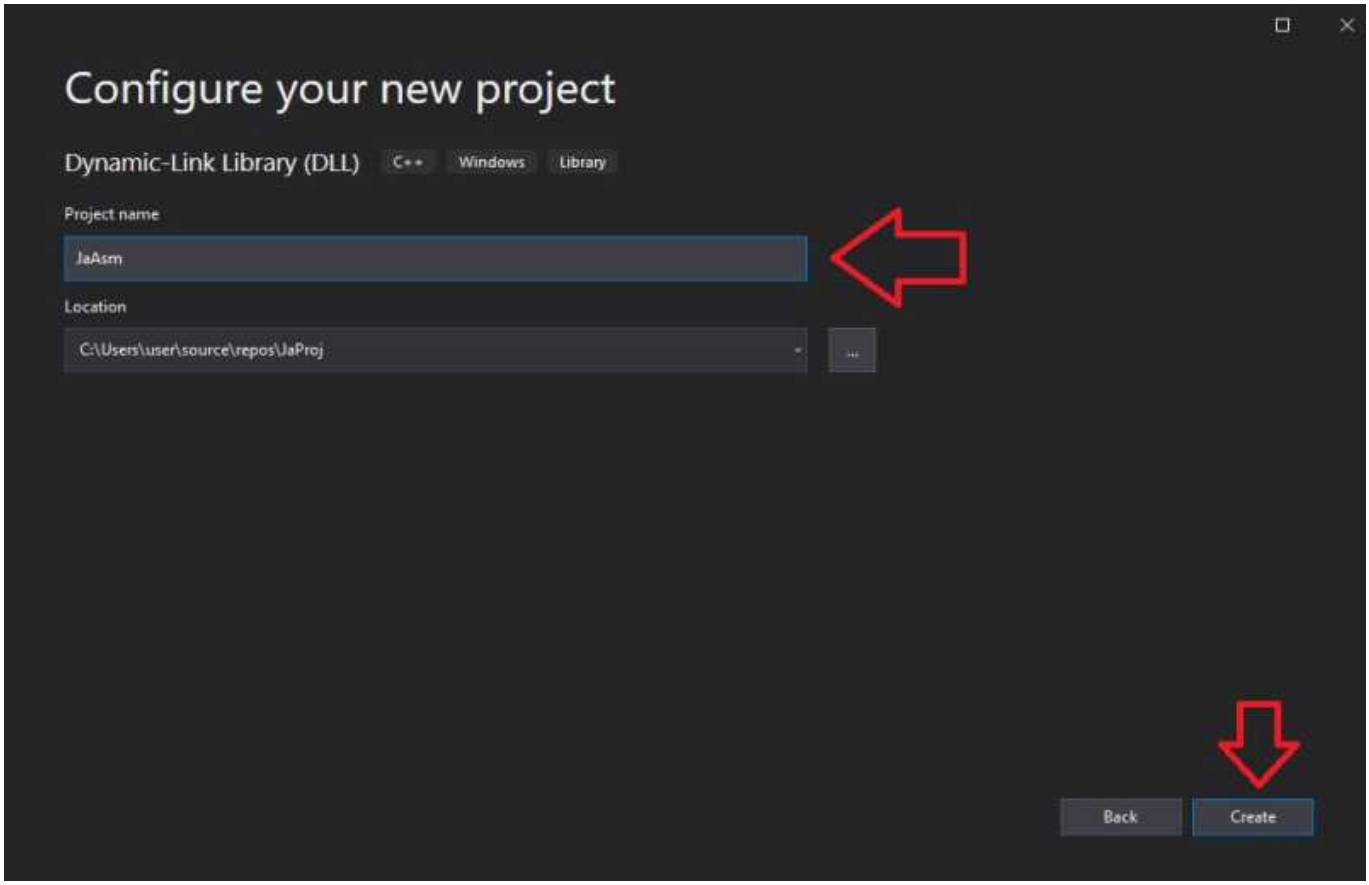
1.2.3Ustawienie zale|no[ci w rozwi zaniu

Nast pnym krokiem jest ustawienie zale|no[ci projektu w taki sposób, a konsolowej w C++ wymuszaBa automatycznie kompilacj biblioteki aser klikn prawym klawiszem myszy na projekt konsolowy C++, wybra mer (Build dependencies), a nast pnie Zale|no[ci projektu (Project depen





W nowo otwartym oknie zaznaczamy nasz bibliotek JaAsm i klikamy

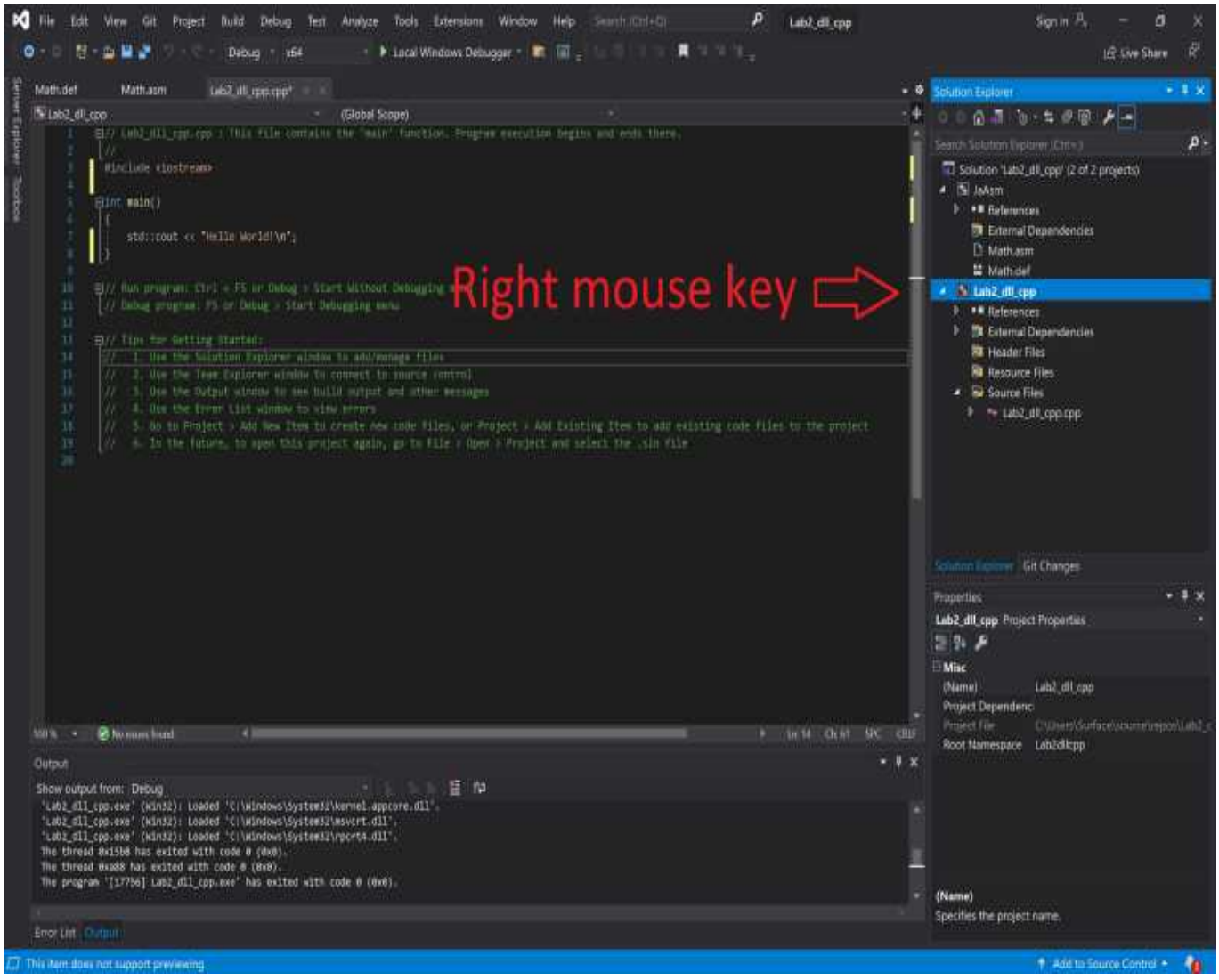




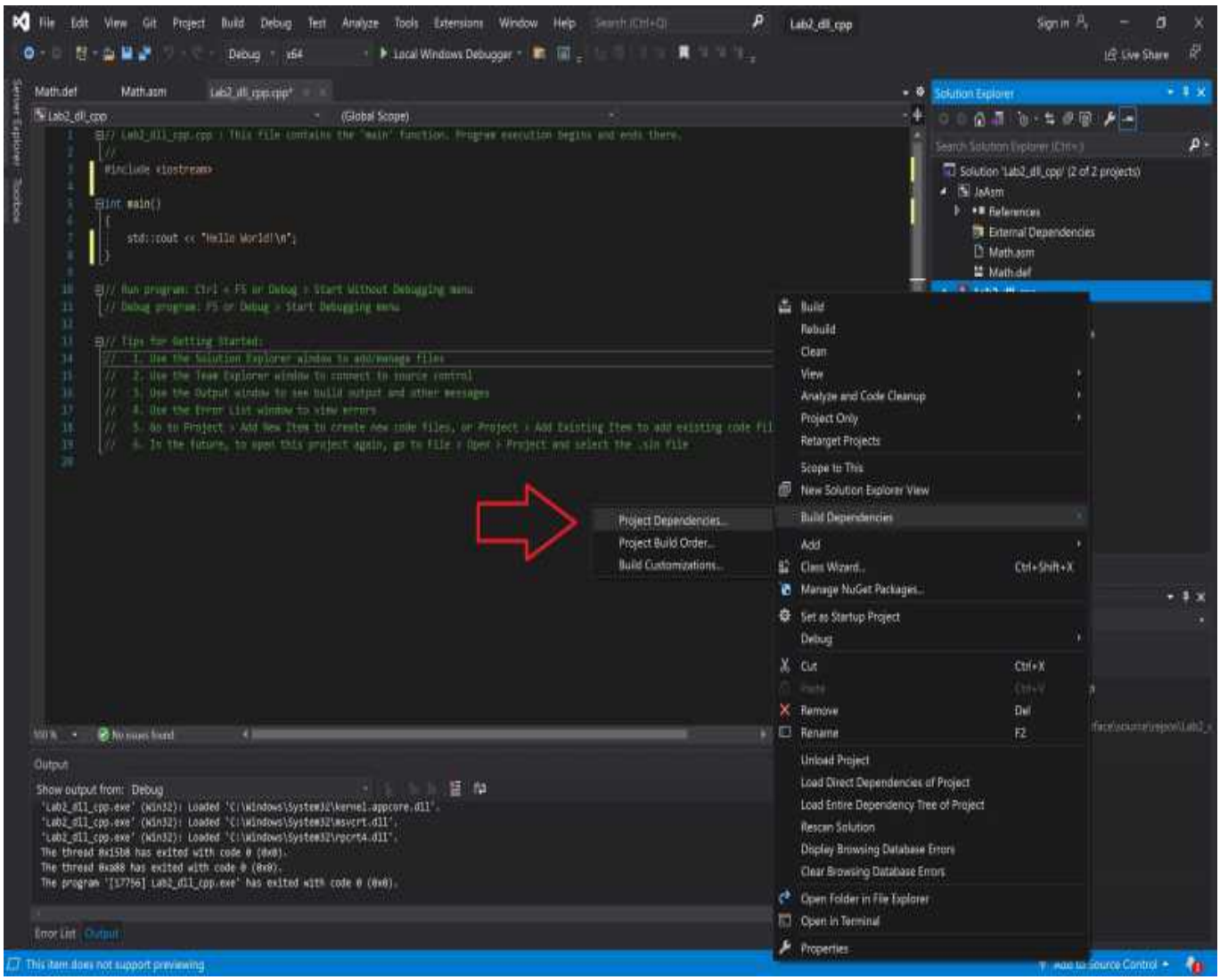
Po ustawieniu tej zale|no[ci, przy kompialcji projektu lub prog automatycznie zostanie tak|e przekompilowany plik asemblerow

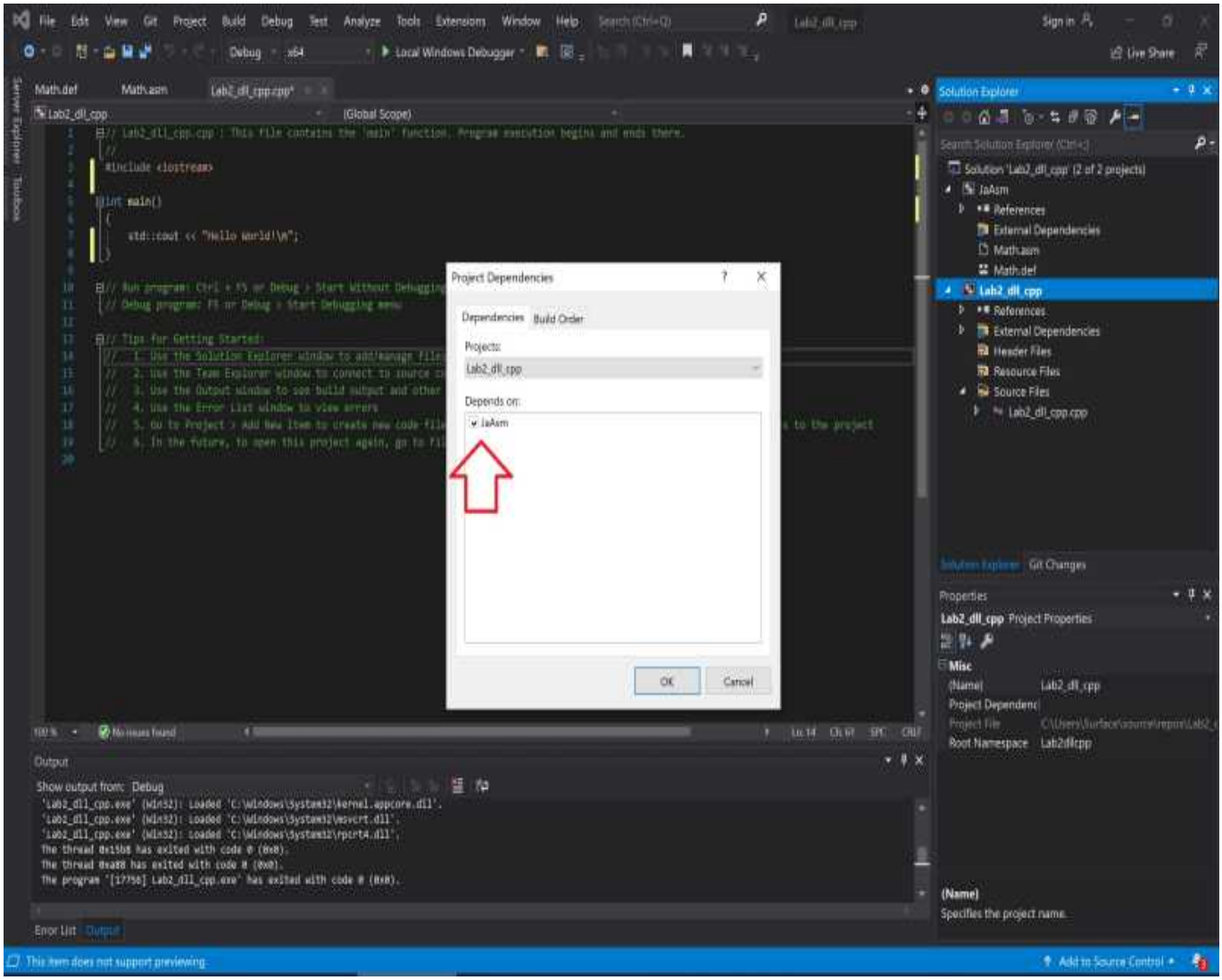
1.2.4Zamiana biblioteki z C++ na asemblerow

Najpierw nale|y usun automatycznie utworzone pliki projektu zgodneg wystarczy zaznaczy wskazane pliki nagBówkowe i zródBowe, a nast pn wybra UsuD b dz te| klikn Delete.

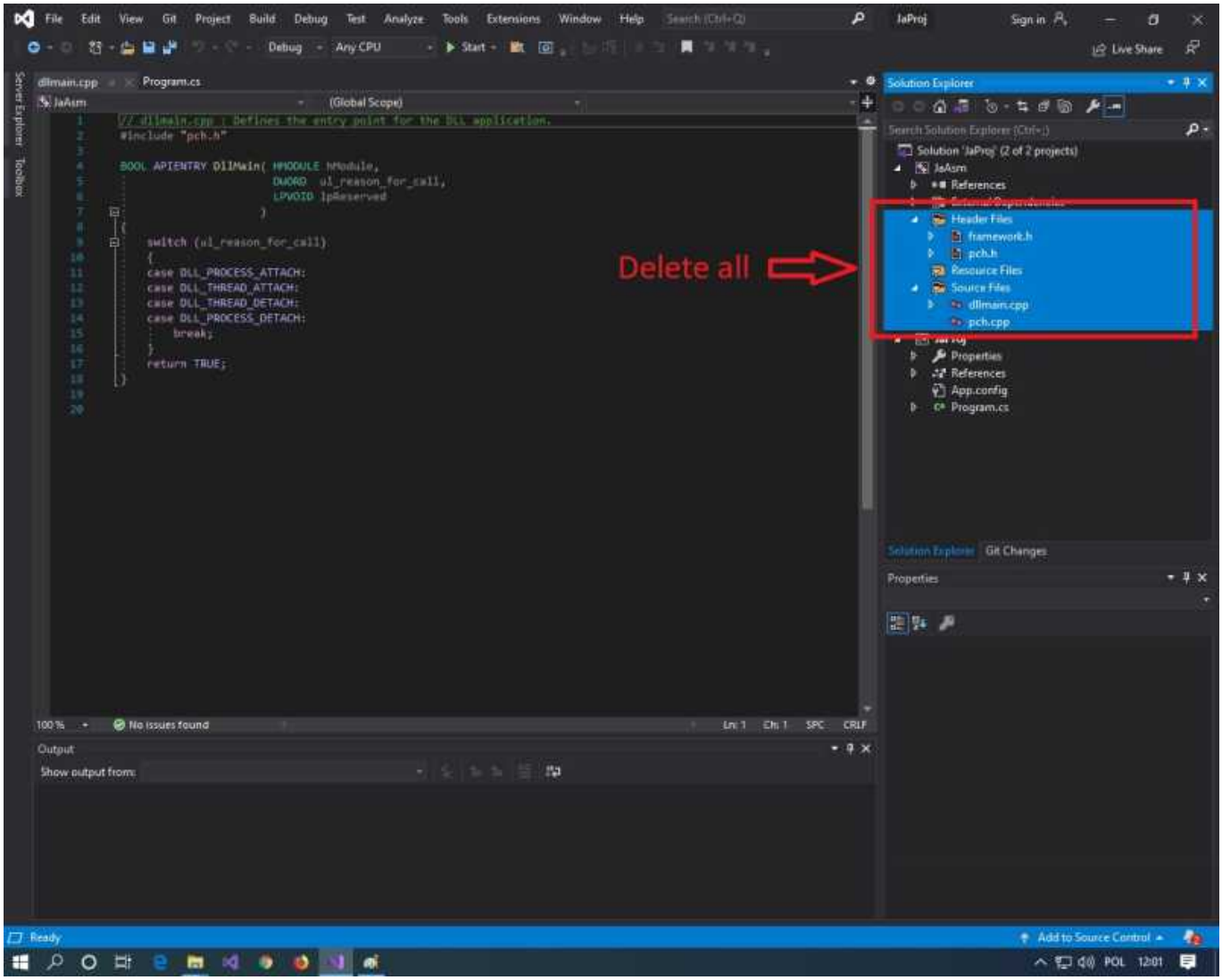


Dla biblioteki konieczne jest dostosowanie kompilacji jako plik asembl klikamy projekt biblioteki prawym klawiszem myszy i z listy wybieramy dependencies), a nast pnie Dostosowanie kompilacji (Build customiza

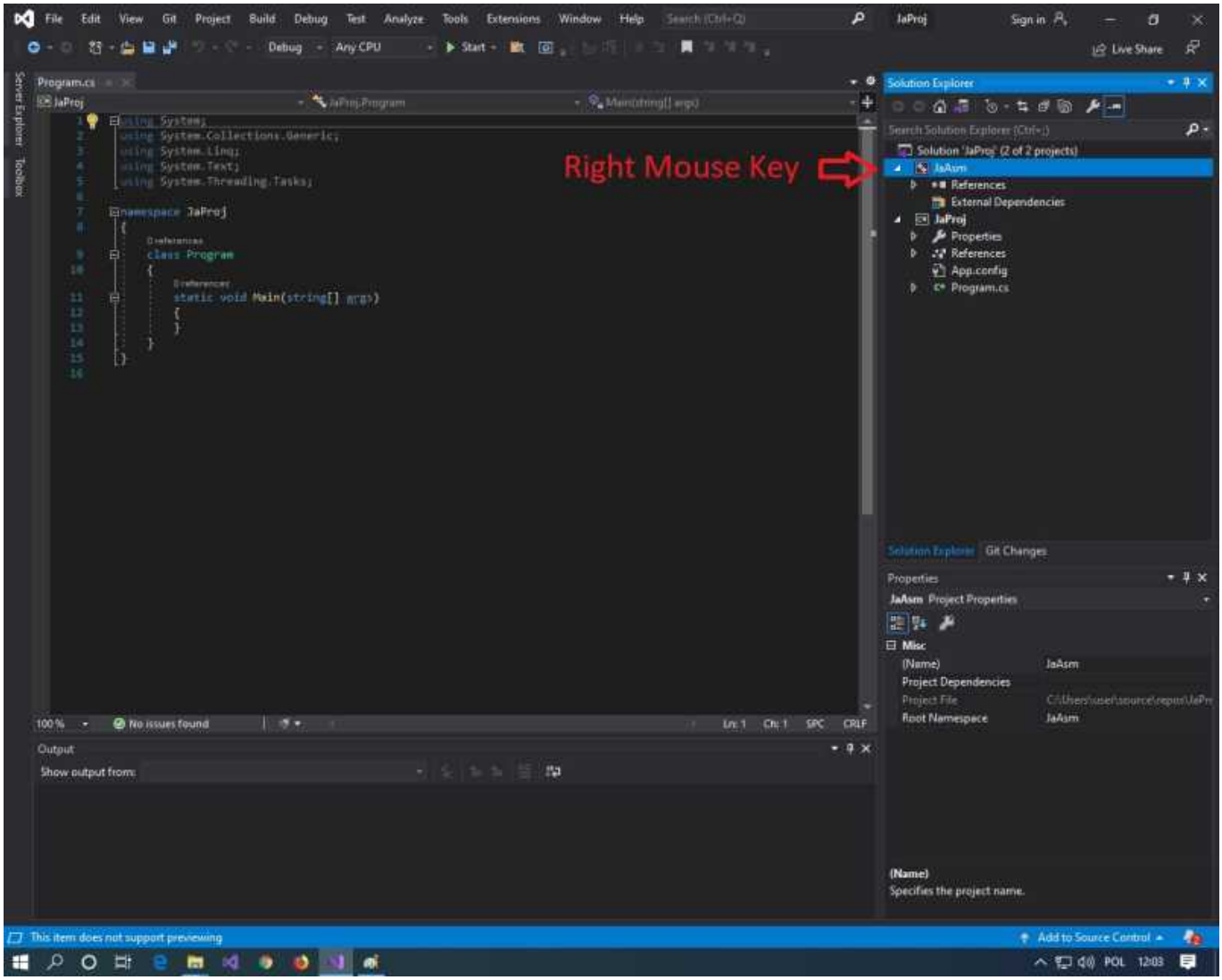

Project Dependencies...
Project Build Order...
Build Customizations...
Build
Rebuild
Clean
View
Analyze and Code Cleanup
Project Only
Retarget Projects
Scope to This
New Solution Explorer View
Build Dependencies
Add
Class Wizard... Ctrl+Shift+X
Manage NuGet Packages...
Set as Startup Project
Debug
Cut Ctrl+X
Remove Del
Rename F2
Unload Project
Load Direct Dependencies of Project
Load Entire Dependency Tree of Project
Rescan Solution
Display Browsing Database Errors
Clear Browsing Database Errors
Open Folder in File Explorer
Open in Terminal
Properties



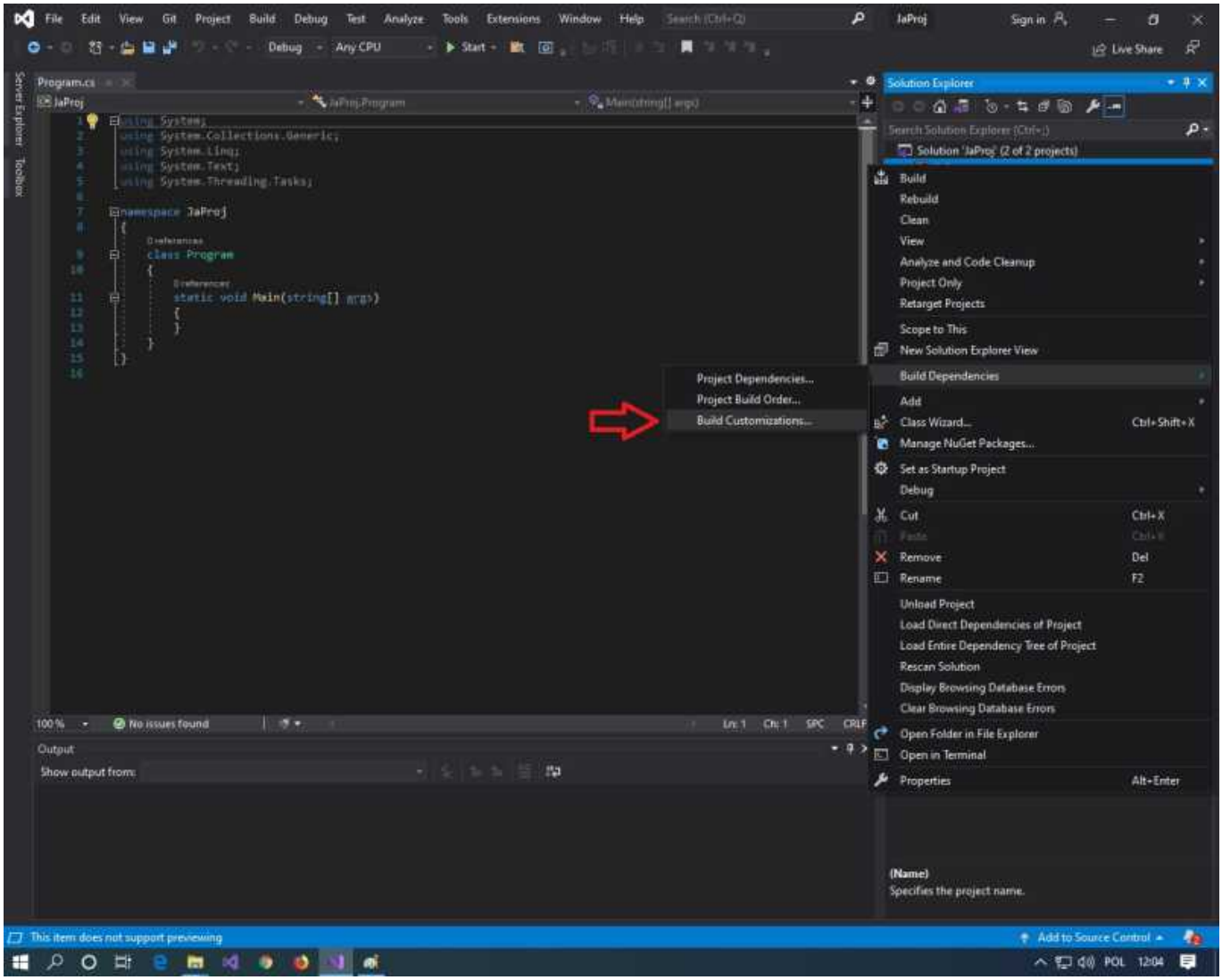
W nowo otwartym oknie zaznaczamy masm i potwierdzamy nasz wybór



Kolejnym krokiem jest dodanie do biblioteki plików źródBowego w asem
to zrobi ponownie klikamy prawym przyciskiem myszy na bibliotek Ja
Dodaj (Add) oraz Nowy element (New item).



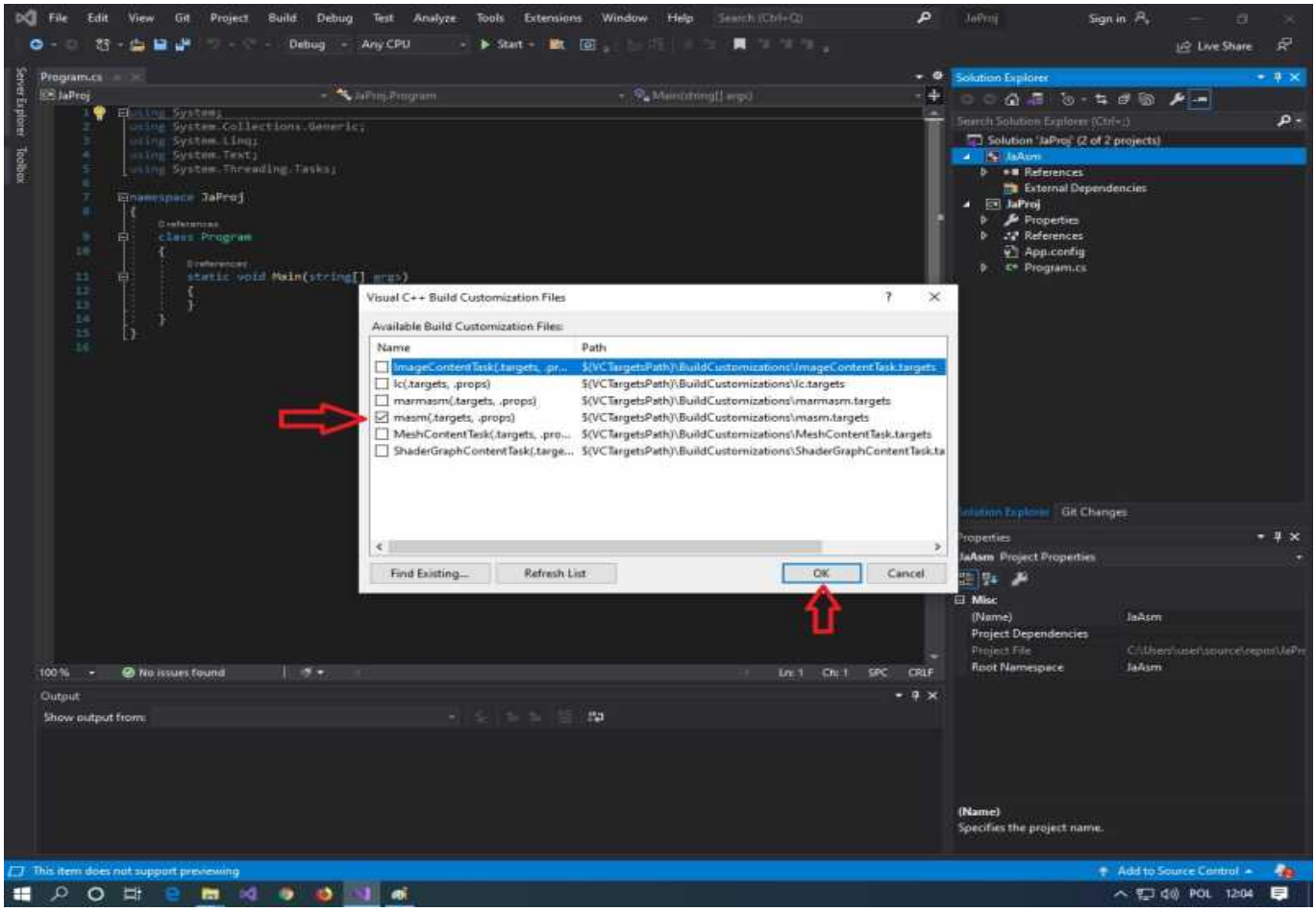
Czynno[t powtarzamy dwukrotnie wybieraj c za ka|dym razem Narz c
tekstowy, podaj c nazwy: JaDll.asm oraz JaDll.def .





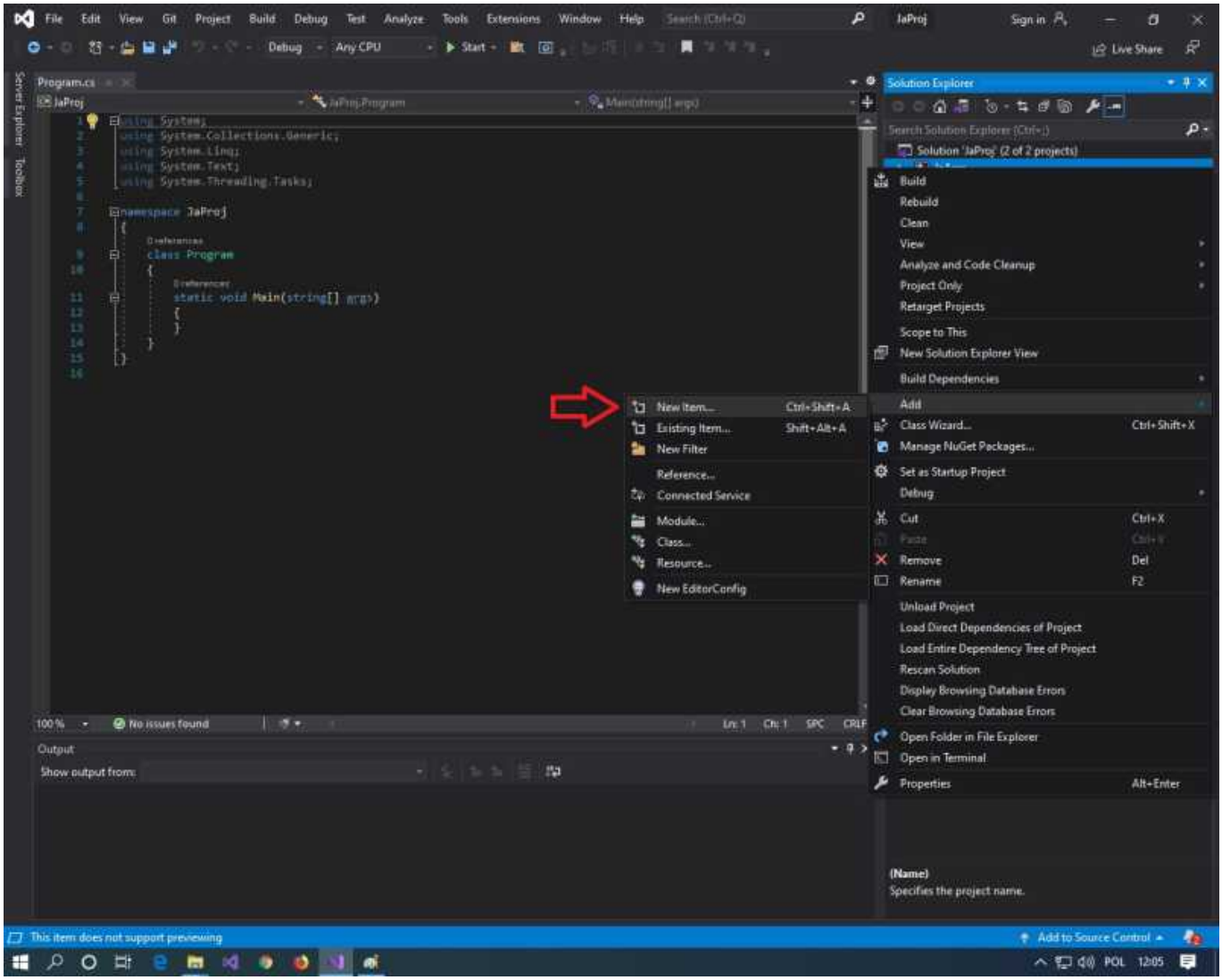
Przy wykonaniu czynno[ci zgodnie z podan uprzedni prawdopodobieDstwem krok przedstawiony na kolejnym rysunku

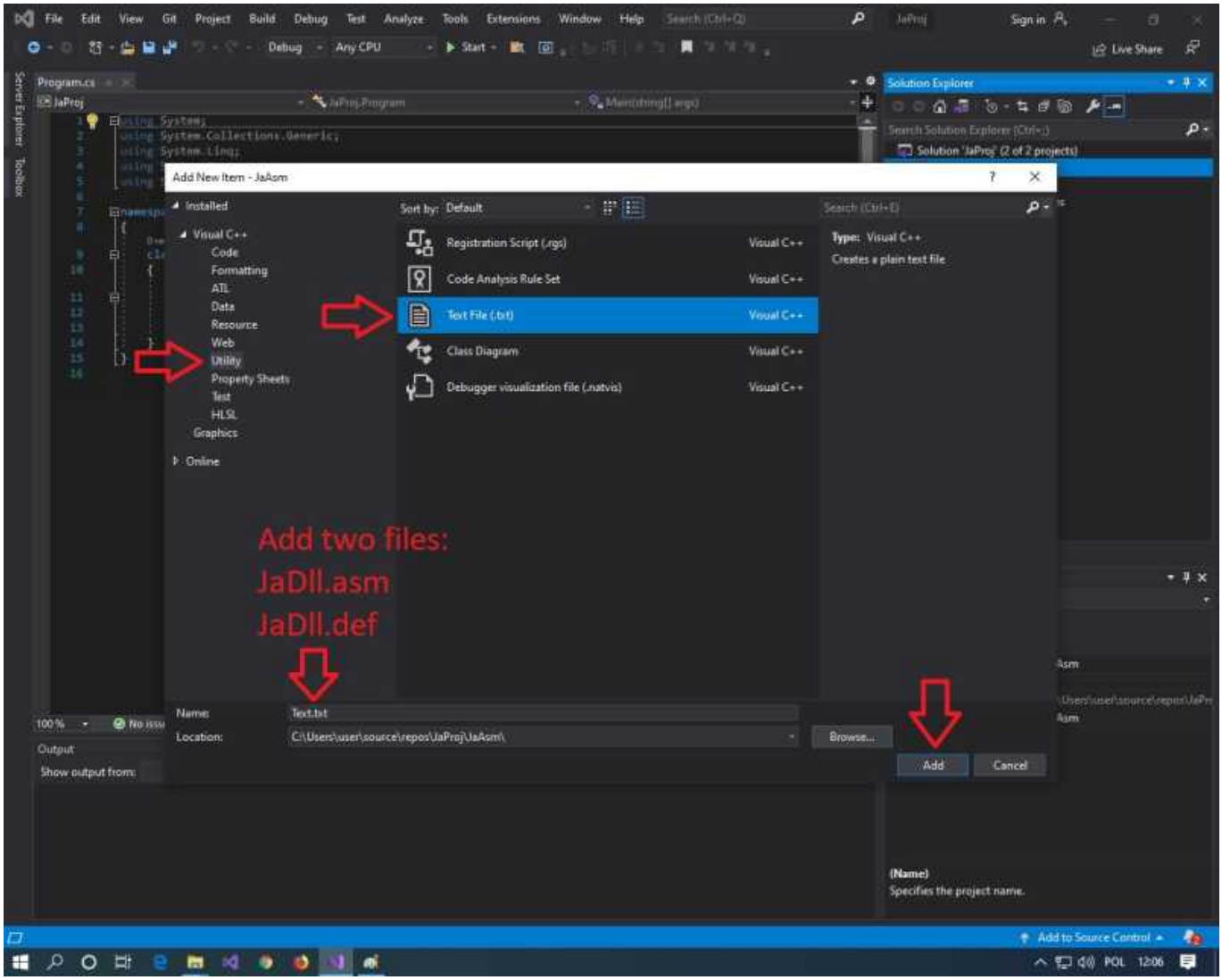
Aby kompilator wiedziaB, |e stworzony plik asemblerowy jest skojarzon typ. Krok ten jest niezb dny gdy wykorzystujemy plik zródBowy C++ i z rozszerzenie na *.asm. Klikaj c pracym klawiszem na pliku *.asm wybie typ pliku na Microsoft Macro Assembler.



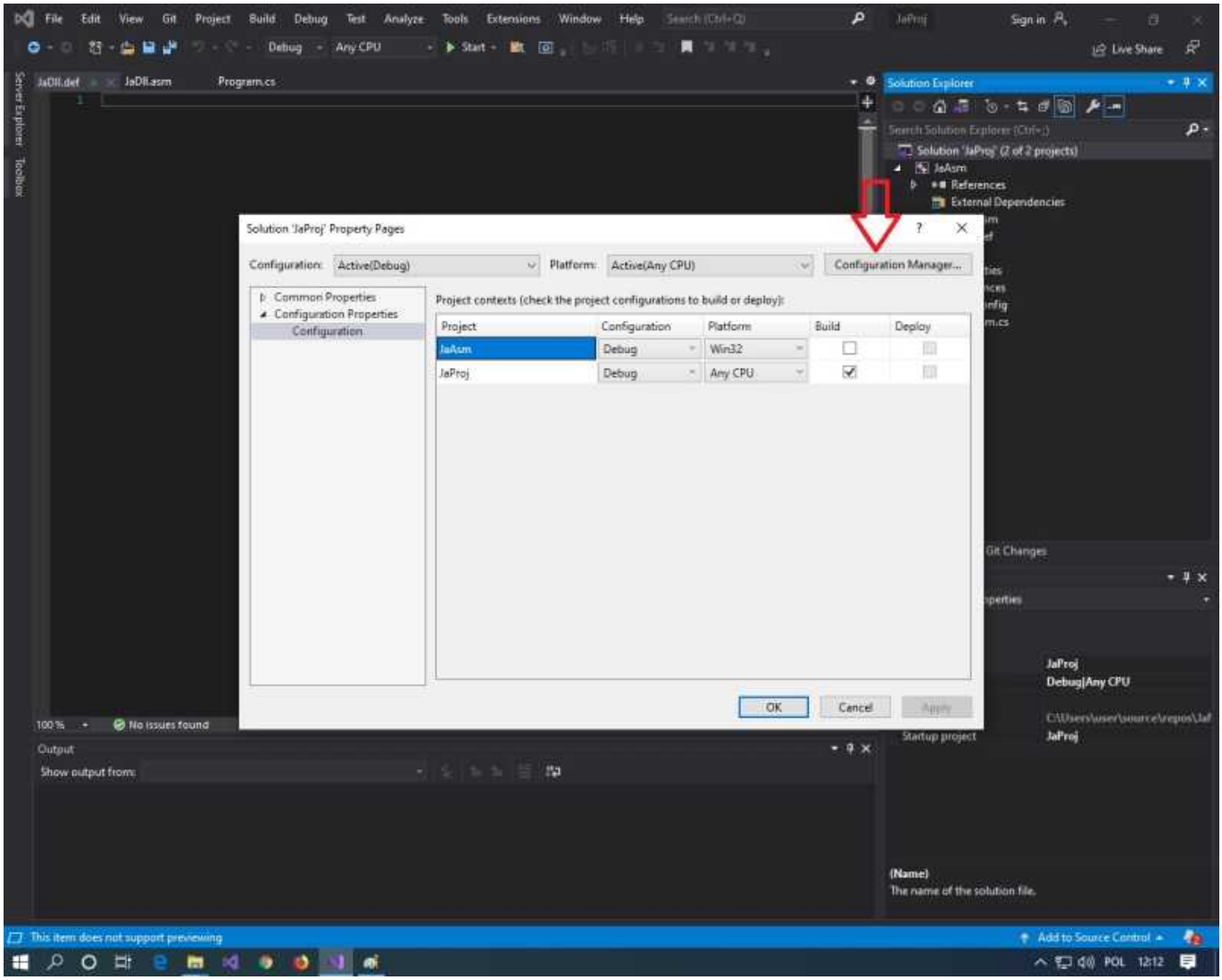
1.2.5Ustawienie trybu kompilacji

Kompilacja powinna przebiega w trybie 64-bitowym (X64). W tym celu opcjach rozwi zania. Klikamy prawym przyciskiem myszy na nasze rozw WBa[ciwo[ci (Properties).

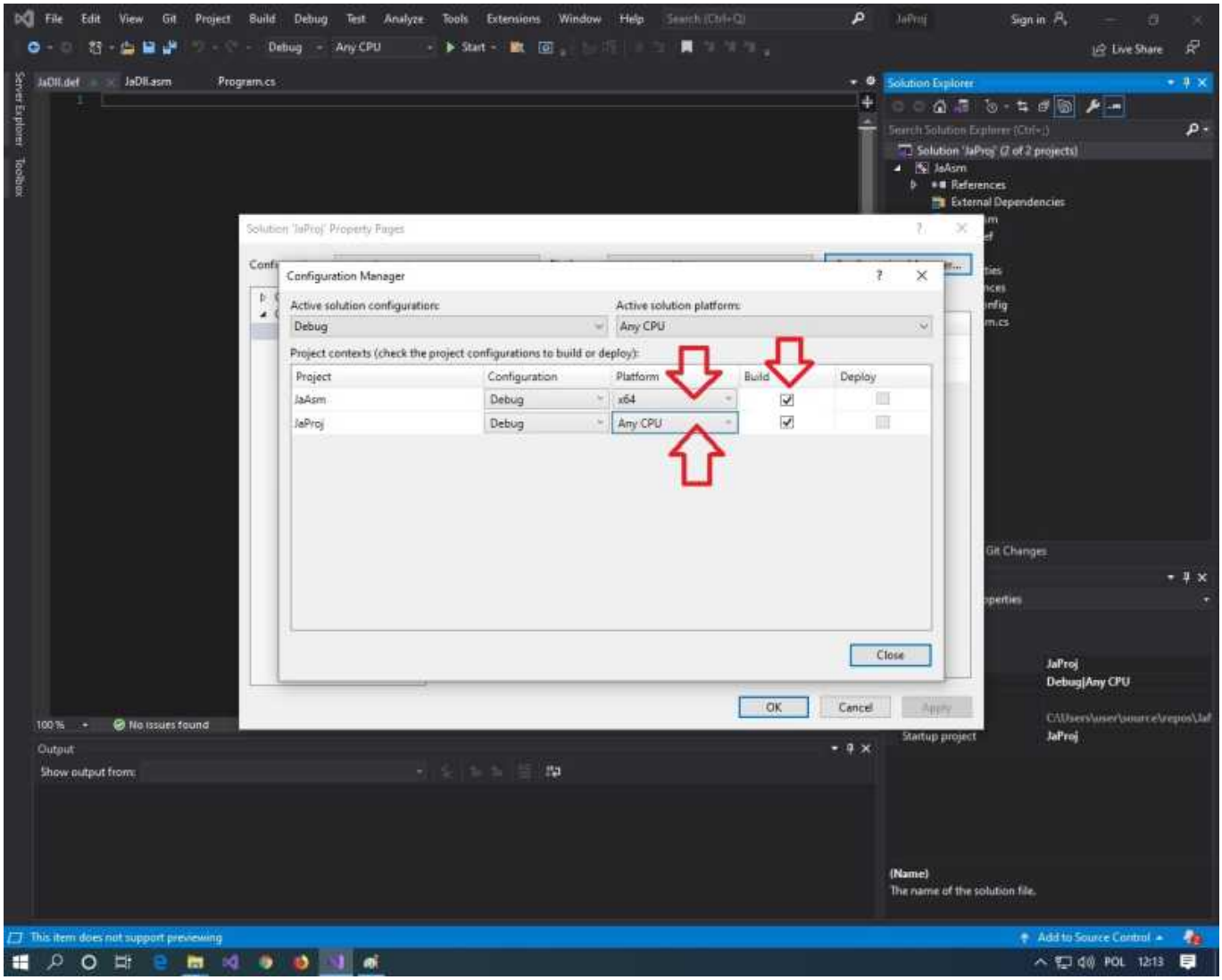

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Debug Any CPU Start
JaProj
Sign in
Live Share
Program.cs
JaProj
JaProj.Program
Main(string[] args)
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace JaProj
{
0 references
class Program
{
0 references
static void Main(string[] args)
{
}
}
}
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'JaProj' (2 of 2 projects)
Build
Rebuild
Clean
View
Analyze and Code Cleanup
Project Only
Retarget Projects
Scope to This
New Solution Explorer View
Build Dependencies
Add
Class Wizard... Ctrl+Shift+X
Manage NuGet Packages...
Set as Startup Project
Debug
Cut Ctrl+X
Paste Ctrl+V
Remove Del
Rename F2
Unload Project
Load Direct Dependencies of Project
Load Entire Dependency Tree of Project
Rescan Solution
Display Browsing Database Errors
Clear Browsing Database Errors
Open Folder in File Explorer
Open in Terminal
Properties Alt+Enter
New Item... Ctrl+Shift+A
Existing Item... Shift+Alt+A
New Filter
Reference...
Connected Service
Module...
Class...
Resource...
New EditorConfig
100 %
No issues found
Ln: 1 Ch: 1 SPC CRLF
Output
Show output from:
(Name)
Specifies the project name.
This item does not support previewing
Add to Source Control
POL 12:05

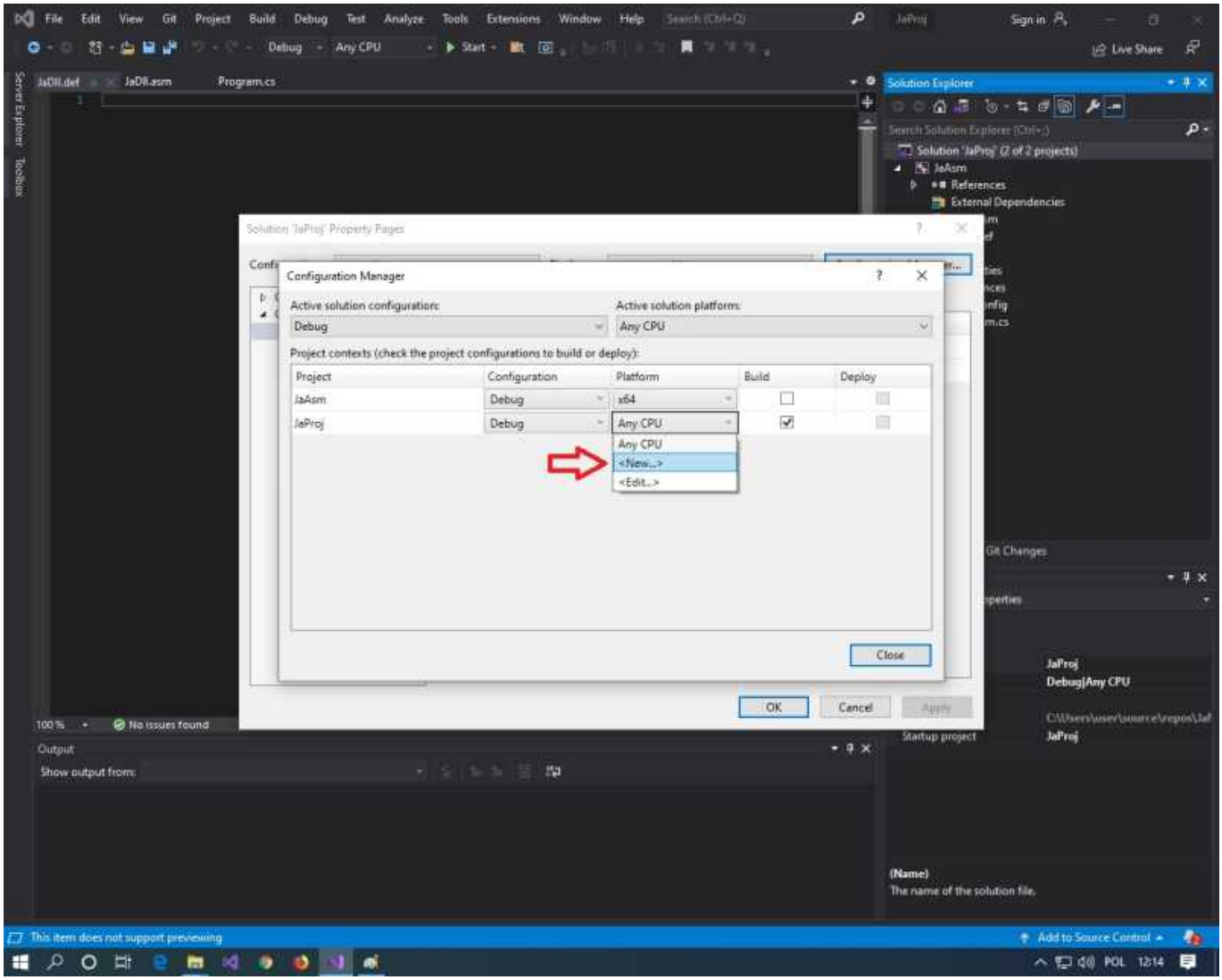


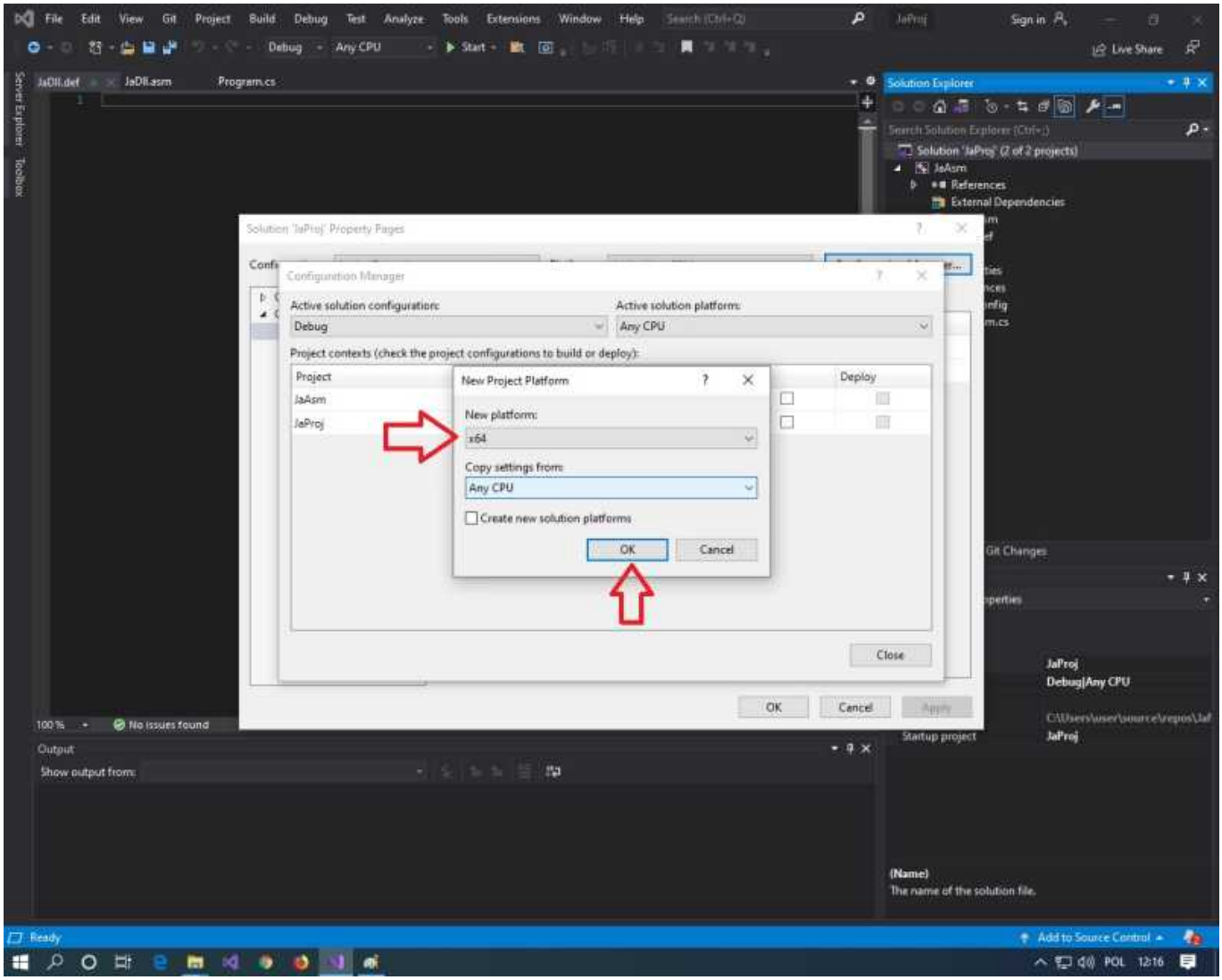
W nowo otwartym oknie przechodzimy na zakBadk WBa[ciwo[ci konß g
(Conß guration properties! Conß guration) na nast pnie klikamy w Men
(Conß guration manager).



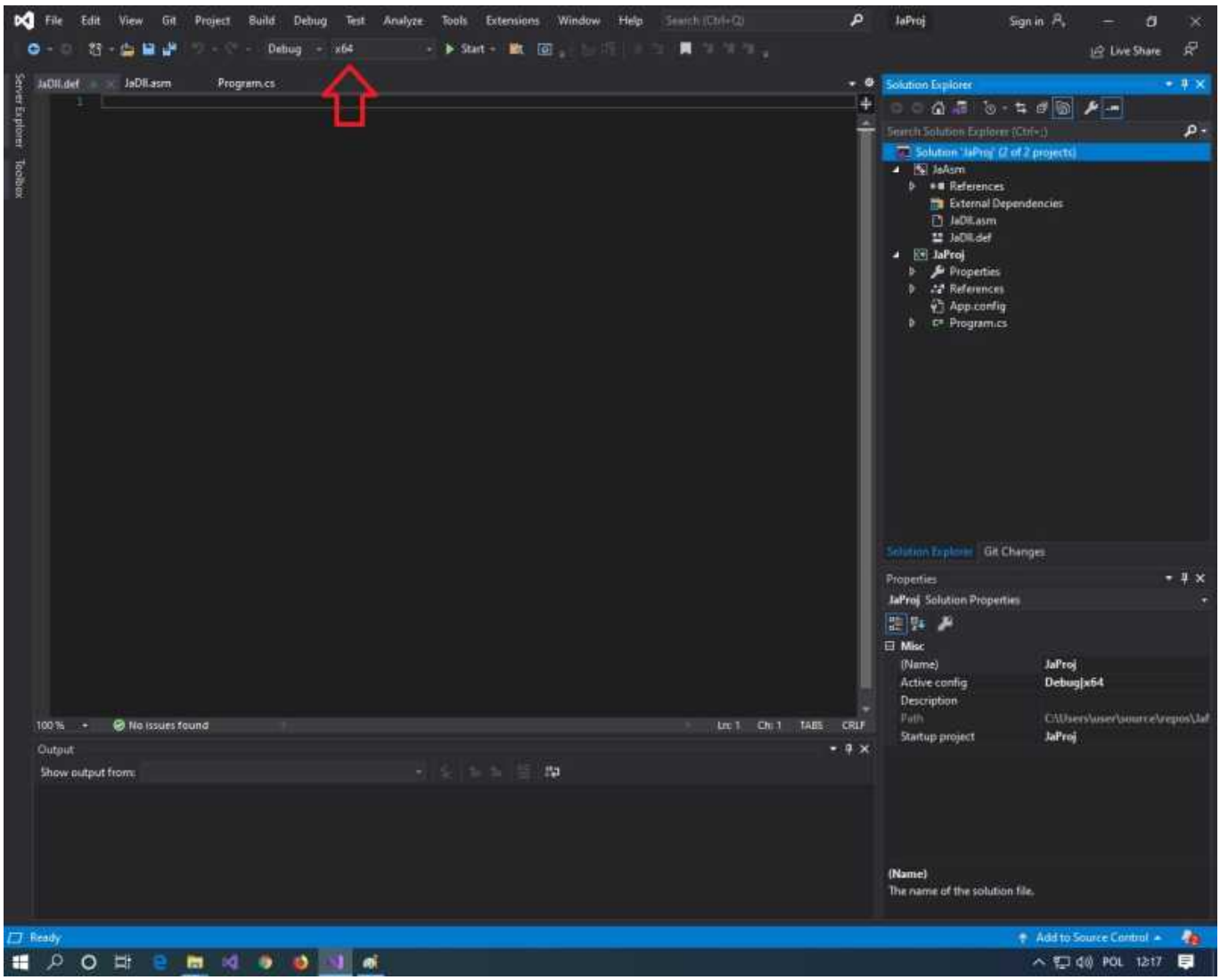
W nowym oknie wskazujemy wsz dzie platform x64 (gdyby dana opcja listy wybieramy pozycj Nowy (New), a nast pnie tworzymy platform ustawieD z Any CPU). Nale|y tak|e przy obu cz [ciach solucji wB czy


Configuration Manager
Active solution configuration:
Debug
Active solution platform:
Any CPU
Project contexts (check the project configurations to build or deploy):
Project
Configuration
Platform
Build
Deploy
JaAsm
Debug
x64
JaProj
Debug
Any CPU
Close
OK
Cancel
Apply


Configuration Manager
Active solution configuration:
Debug
Active solution platform:
Any CPU
Project contexts (check the project configurations to build or deploy):
Project
Configuration
Platform
Build
Deploy
JaAsm
Debug
x64
JaProj
Debug
Any CPU
Any CPU
<New...>
<Edit...>
Close
OK
Cancel
Apply
Solution Explorer
Solution 'JaProj' (2 of 2 projects)
JaAsm
References
External Dependencies
Git Changes
JaProj
Debug|Any CPU
JaProj
(Name)
The name of the solution file.
Output
Show output from:
No issues found
This item does not support previewing
Add to Source Control

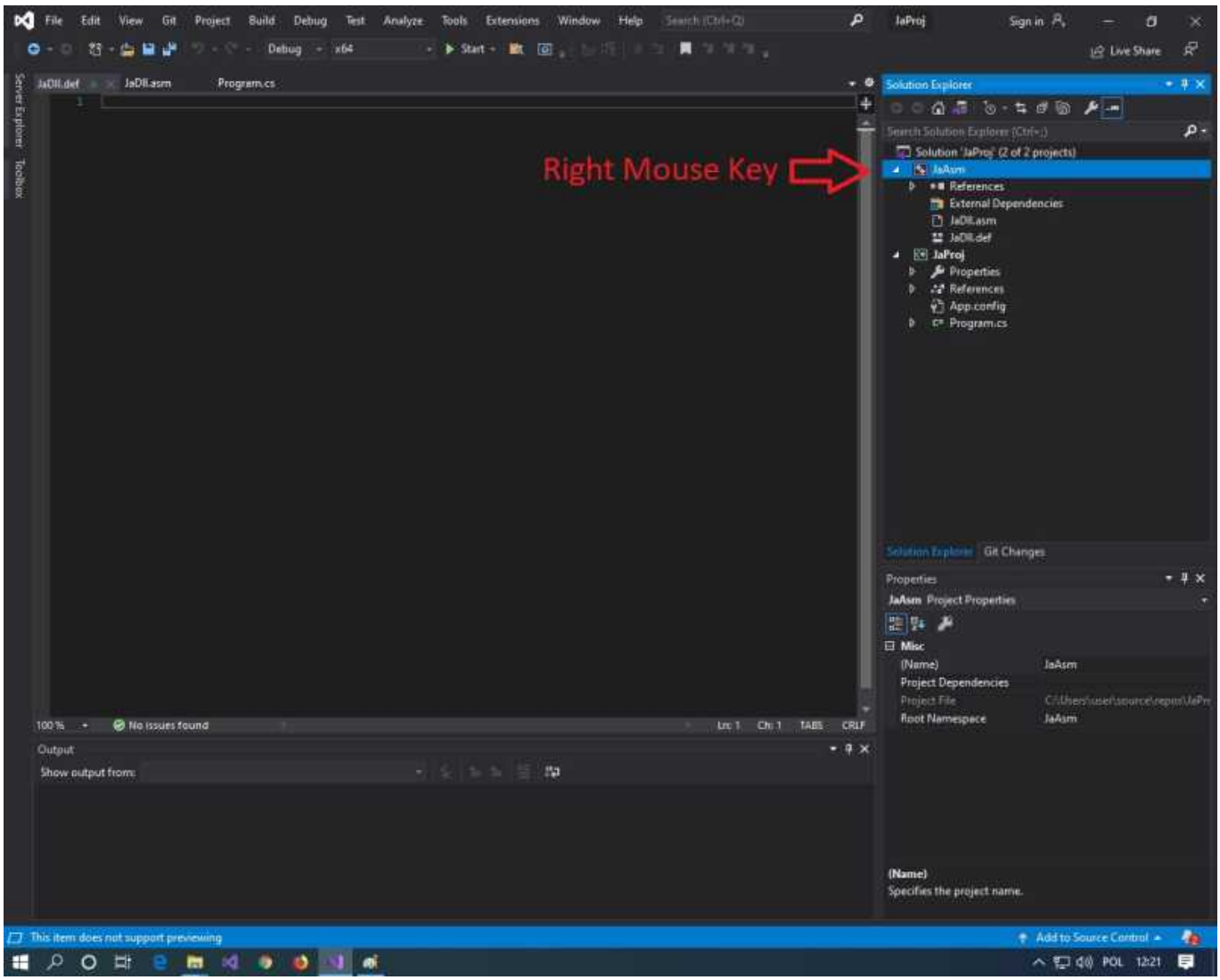


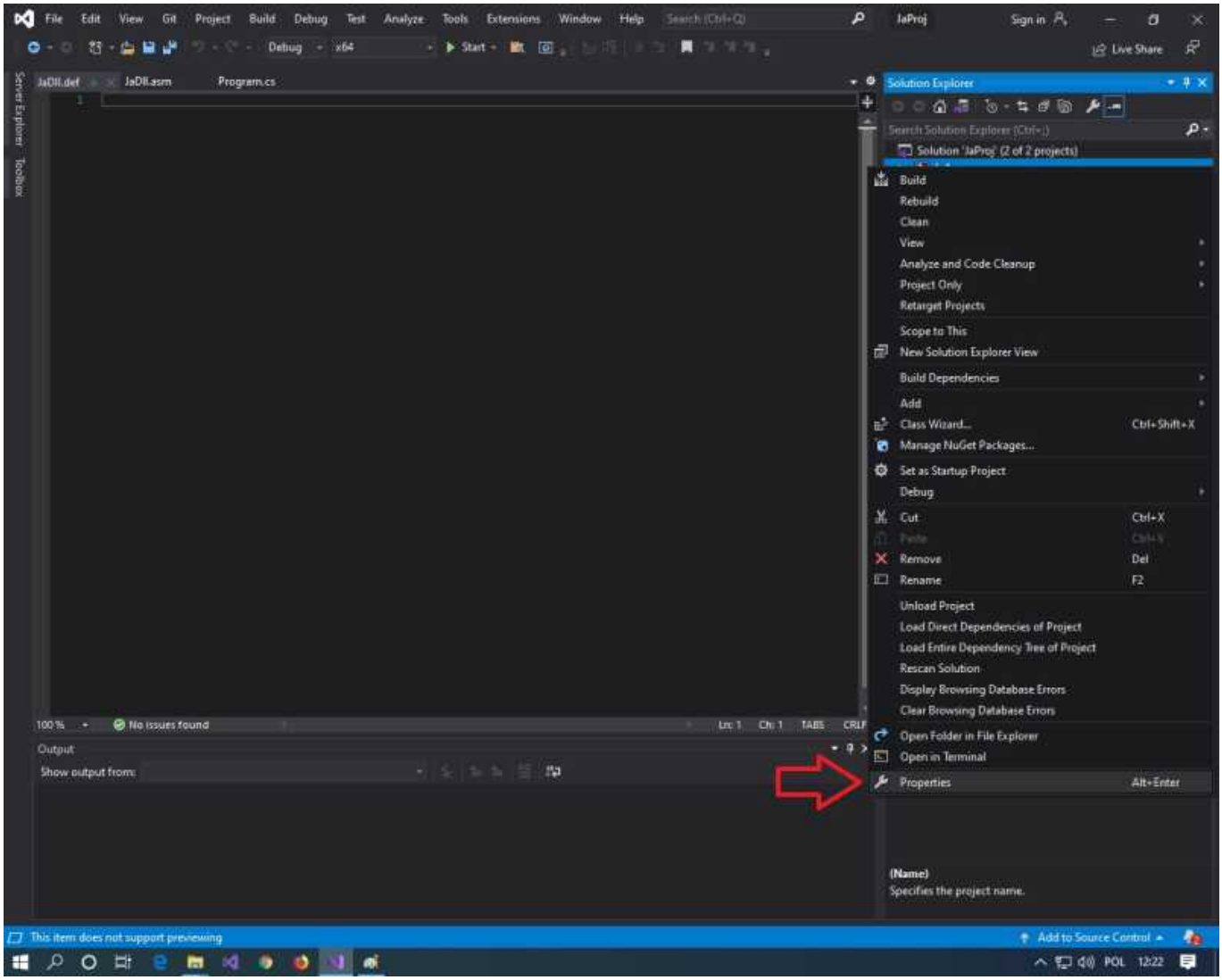
Po wszystkim upewniamy si raz jeszcze, |e wsz dzie wskazana platfor
W gBównym pasku menu Visual Studio powinna automatycznie ustawi s
warto to sprawdzi i w razie potrzeby zmodyß kowa .



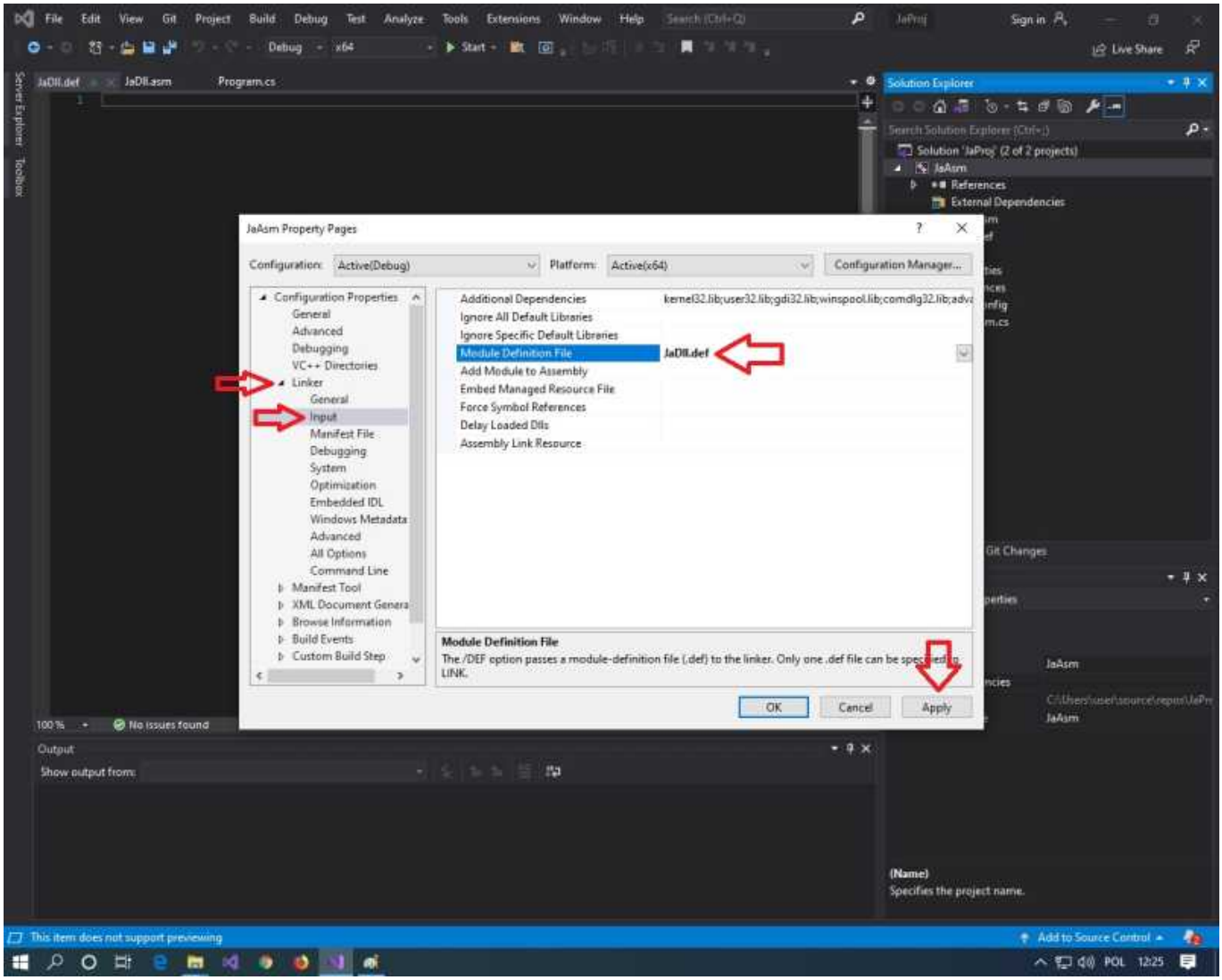
1.2.6KoDcowa konß guracja biblioteki asemblerowej

Przyst pujemy do konß guracji biblioteki. W tym celu klikamy prawym p
i wybieramy WBa[ciwo[ci (Properties).

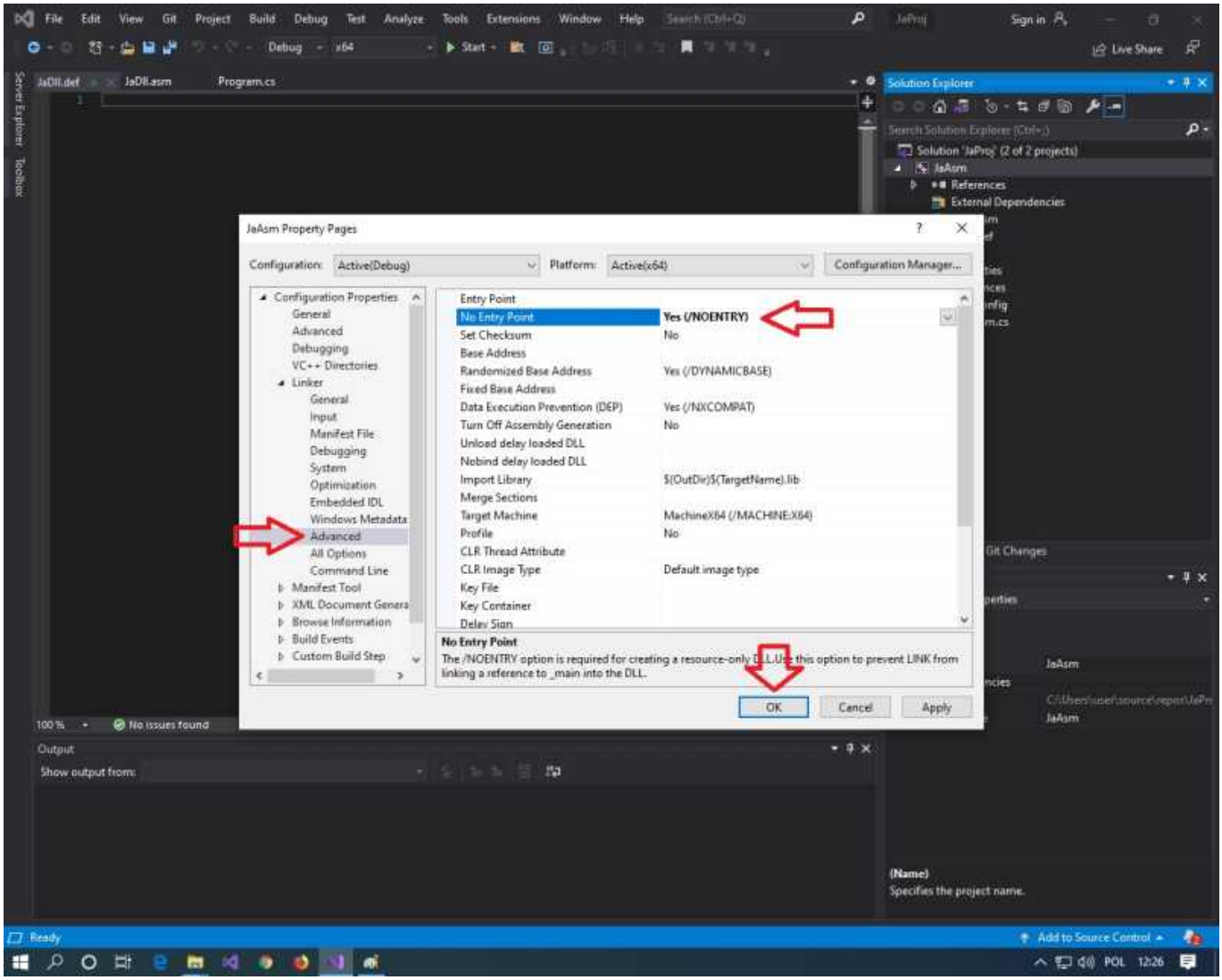

Right Mouse Key



W otwartym nowo oknie przechodzimy do Konsolidator ! Dane wej[cio polu Plik deß nicji moduBu (Module deß nitionß le) wpisujemy nazw n przypadku JaDll.def . Wpis potwierdzamy.



KoDcowym krokiem konß guracji jest wyB czenie punktu wej[cia do bibli
Konsolidator (Linker) przechodzimy do opcji Zaawansowane (Advan
wej[cia (No entry point) wybieramy opcj Tak (/NOENTRY). Zapisujer
zamykamy okno.





Biblioteka dll, w przeciwieDstwie do programu gBównego, startowego, który wykorzystuje system operacyjny do u Udost pnia jedynie zestaw funkcji dla innych programów.

1.3 Wykorzystanie biblioteki dll napisanej w as

1.3.1 Biblioteka asemblerowa dll

Biblioteka dll zawiera funkcje, które mo|na u|y w innych programach. biblioteki b dzie udost pniaBa jedn funkcj , zwracaj c staB warto[. odczytamy w pliku C++. PrzykBadowa zawarto[pliku asemblerowego (nast puj co.

```
.code
MyProc proc
mov RAX, 2021
ret
```

```
MyProc endp
end
```

Z plikiem kodu programu nierozerwalnie zwi zany jest plik deß nicji (Ja nazw biblioteki DLL jako agrument dyrektywy LIBRARY, oraz list nazw moduBów poprzedzonych dyrektyw EXPORTS. Aby nasza funkcja MyPro C++ musimy j zadeklarowa jako eksportowan . Plik JaDll.def w naszy nast puj co.

```
LIBRARY JaAsm
EXPORTS MyProc
```

Cz sto chcemy wykorzysta przekazane do funkcji parametry. Przypomn konwencji wywoBaD pierwsze cztery parametry s umieszczane bezpo[r SzczegóBy s opisane w instrukcji do pierwszego wiczenia, tak, dwa parametry typu caBko umieszczone w rejestrach RCX i RDX. Funkcja, któa zwraca sum dwóch wygl daBaby w nast puj cy sposób.

```
...
MyProc proc
mov RAX, RCX
add RAX, RDX
ret
MyProc endp
...
```

Argumenty typu zmiennoprzecinkowego nale|y przekaza przez rejestry zwróci przez XMM0. Oczywi[cie nale|y u|y instrukcji rozszerzeD MMX/

```
...
FloatProc PROC
addsd xmm0, xmm1
ret
FloatProc ENDP
...
```

1.3.2OdwoBanie do biblioteki dll z programu C++

Aby program napisany w C++ mógB wykorzysta funkcj z biblioteki nal

1. Przygotowa typ funkcji zgodny z jej późniejszym wywoBaniem,
2. ZaBadowa bibliotek funkcj LoadLibrary uzyskuj c jednocze[nie je
3. Odczyta aktualny adres interesuj cej nas funkcji wywoBuj c GetPr
4. U|y funkcji odwoBuj c si do jej adresu.

PrzykBadowy kod mógBby wygl da tak:

```
#include <windows.h>
#include <iostream>
```

```
typedef int(_stdcall* MyProc)();
HINSTANCE dllHandle = NULL;

int main()
{
    dllHandle = LoadLibrary(TEXT("JaAsm.dll"));
    MyProc procedura = (MyProc)GetProcAddress(dllHandle, "MyProc")
    std::cout << procedura();
}
```

U|ycie funkcji z parametrami wymaga nieco zmodyß kowanych deß nicji t uwag na typ warto[ci zwracanej przez funkcj . Pami ta nale|y, |e w pr zmiennoprzecinkowych, rozmiar rejestrów XMM0, XMM1 to 64 bity. Odp zmiennoprzecinkowej podwójnej precyzji, czyli w j zyku C++ typowi do

```
...
typedef int(_stdcall* MyProc)(int, int);
typedef double(_stdcall* FloatProc)(double, double);
...
```

1.3.3 Zadanie do zrealizowania w ramach laboratorium

Nale|y napisa bibliotek DLL w asemblerze i program w C++ demonstru Biblioteka DLL udost pnia funkcje wykonuj ce i zwracaj ce wynik podst arytmetycznych (dodawanie, odejmowanie, mno|enie i dzielenie) na pa parametry ich wywoBania. Funkcje powinny mie wersj staBo- i zmienn funkcji).
Program w C++ wywoBywa wszystkie funkcje przekazuj c przykBadowe wy[wietli wyniki na ekranie. Nie jest wymagany interfejs u|ytkownika by umieszczone w kodzie programu.

From:
https://dokuwiki.zmitac.aei.polsl.pl/ - RAU6 Dokuwiki

Permanent link:
https://dokuwiki.zmitac.aei.polsl.pl/pl:edu:zmitac:n1z:ja:[illegible]16174

Last update: 2021/04/03 19:38