

Wykład 1

**Wstęp do programowania w
środowisku Visual C++**

Autor: Zofia Kruczkiewicz

Zagadnienia

1. Podstawowe pojęcia
2. Tworzenie aplikacji w Windows Forms
3. Zawartość projektu
4. Podstawowe cechy języka C++/CLI
5. Najprostsza aplikacja – kalkulator
6. Tworzenie projektu do tworzenia definicji klasy w pliku nagłówkowym i pliku dll
7. Tworzenie programu korzystającego z klasy zdefiniowanej w pliku nagłówkowym
8. Interfejs graficzny aplikacji – obsługa zdarzeń

1. Podstawowe pojęcia

Typy programów

- C++ ISO/ANSI – programy natywne C++
- C++/CLI – programy pracujące pod kontrolą CLR

Standardy

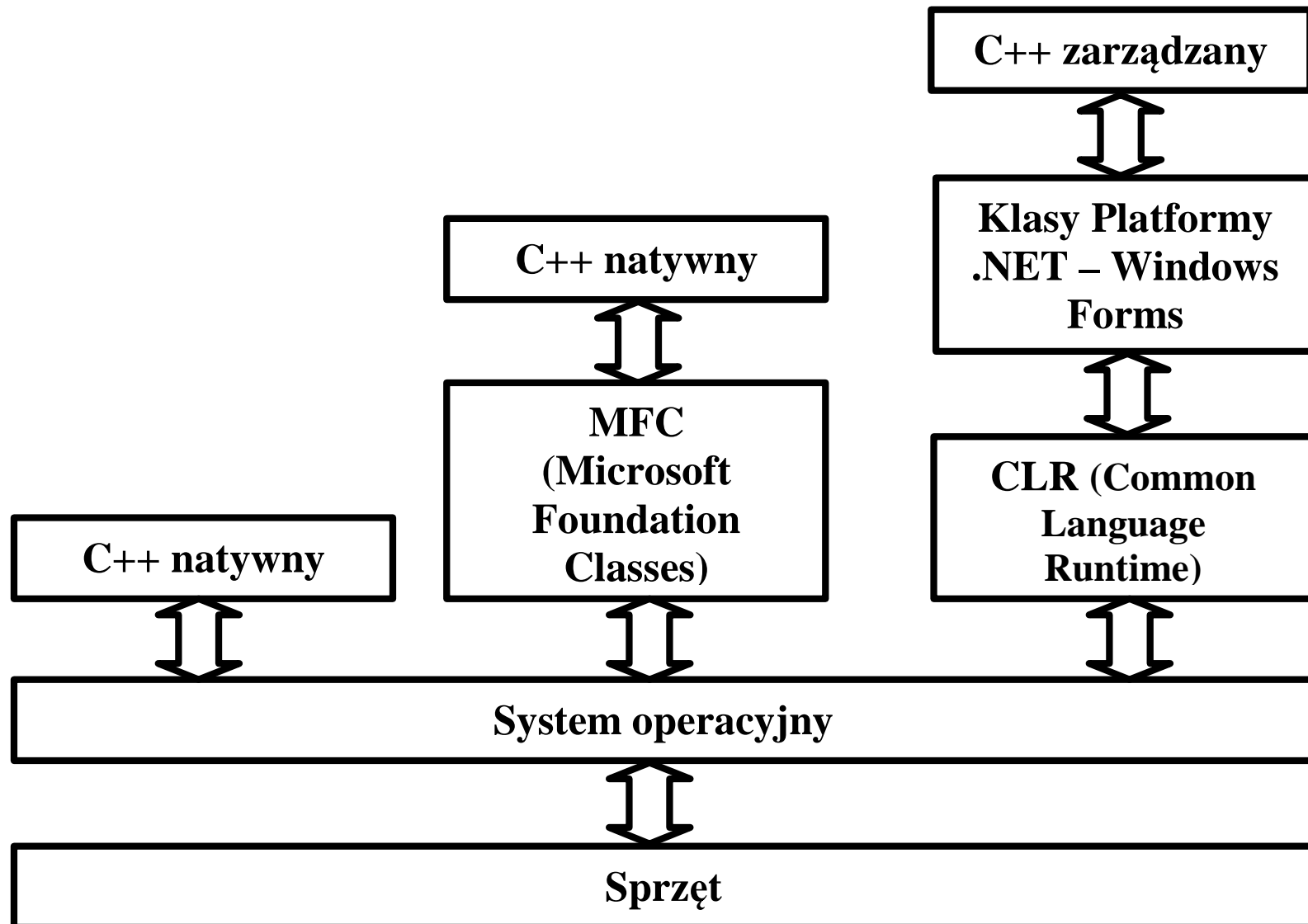
- CLR (*Common Language Runtime*) jest standardowym środowiskiem do wykonywania programów napisanych w językach Visual Basic, C#, C++ oraz bibliotek ADO.Net, ASP.NET. To podstawa całego systemu .NET Framework
- Specyfikacja CLR zawiera się w równorzędnych standardach
 - CLI (*Common Language Infrastructure*), opracowanym przez ECMA (European Computer Manufacture) jako standard ECMA-335
 - ISO (ISO/IEC 23271).

Właściwości

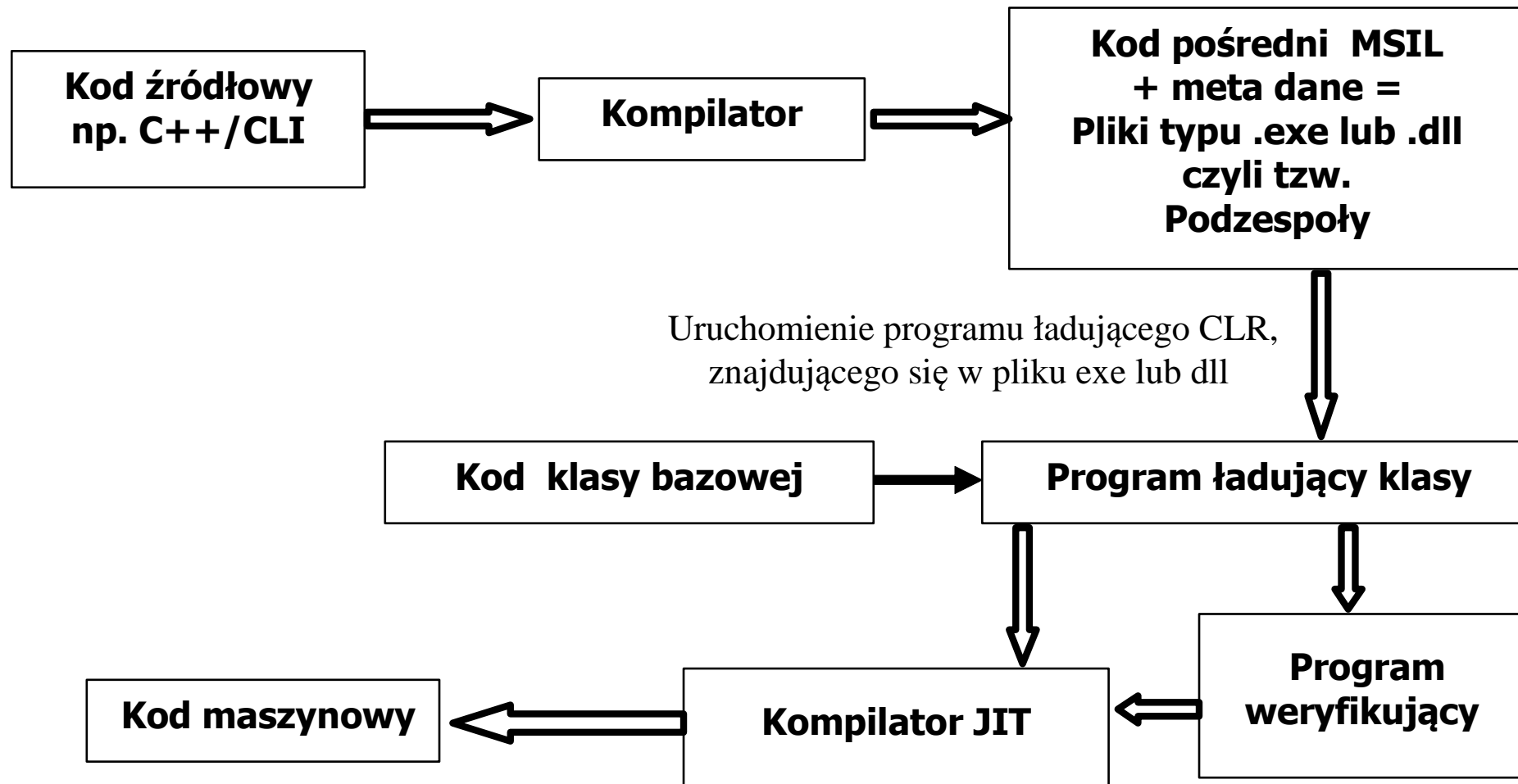
- Platforma .NET jest częścią systemu operacyjnego Windows
- Dzięki specyfikacji CLI kompilatory C++/CLI mogą działać w wielu systemach operacyjnych.

Tworzenie programów w C++

wg Ivor Horton „Od podstaw Visual C++ 2005”



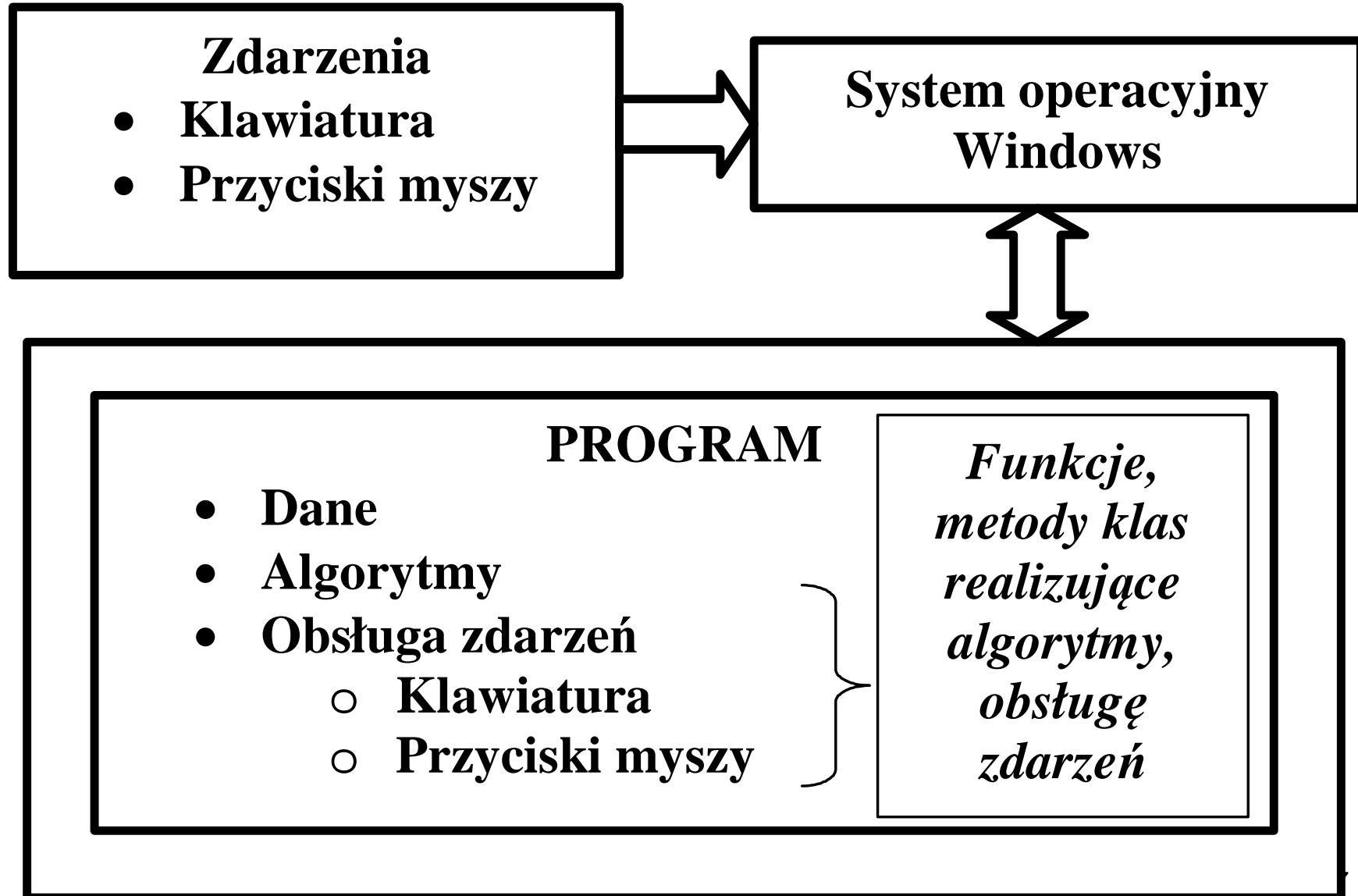
Przebieg tworzenia i działania programu w języku zarządzanym C++ (w środowisku CLR) – (1)



Przebieg tworzenia i działania programu w języku zarządzanym C++ (w środowisku CLR) – (2)

- Program napisany z języku wysokiego poziomu (C++, C#, J#, Visual Basic) jest kompilowany do języka pośredniego dla maszyny wirtualnej wyspecyfikowanej przez CLI – jest to język MSIL (*Microsoft Intermediate Language*). **Kompilator tworzy kod pośredni MSIL oraz tzw. metadane, zapewniające informacje o klasach**
- **Role metadanych:**
 - Udostępnianie definicji typów danych
 - Lokalizowanie i ładowanie klas
 - Rozmieszczenia obiektów w pamięci
 - Egzekwowanie zasad bezpieczeństwa
 - Odczytywane informacji o sposobie uczestniczenia danej klasy w transakcjach.
- **Kod pośredni w postaci pliku .exe lub dll tworzy tzw. podzespoły (assemblies)**, w których rezydują klasy C++/CLI, oraz zasoby, które współpracując tworzą podstawowe logiczne funkcjonalności programu.
- W momencie uruchomienia kodu pośredniego startuje biblioteka mscorlib.dll zapewniająca hosting wszystkim podzespołom - startuje program ładujący środowisko CLR w celu załadowania kompilatora JIT (Just-in-time)
- **Kompilator typu JIT mapuje kod w języku pośrednim i metadane na kod maszynowy "w locie"**. Dzięki temu kod pośredni CLI może być uruchomiony w dowolnym środowisku posiadającym implementację CLI (CLR jest implementacją stworzoną przez firmę Microsoft)
- Kompilator JIT jest uruchamiany w momencie wywoływania po raz pierwszy metody i tłumaczy ją na kod maszynowy. Dokonuje się wtedy weryfikacji kodu: czy typy danych są bezpieczne i operacje wykonywane są legalnie (nie każdy program podlega weryfikacji). Kiedy metoda jest wywołana ponownie, jest natychmiast wykonywana w kodzie maszynowym. **Oznacza to, że weryfikowane i kompilowane są tylko te metody, które są wywoływane.**
- Program jest kompilowany w trakcie pierwszego wykonywania lub podczas instalacji oprogramowania (można go uruchomić po raz kolejny bez potrzeby kompilacji).
- **Podczas działania programu po kompilacji do kodu maszynowego uruchamiane są następujące usługi zarządzające wykonaniem skompilowanego programu:**
 - Usuwanie nieużytków w pamięci,
 - Zarządzanie bezpieczeństwem - w CLR *bezpieczeństwo jest oparte na uprawnieniach kodu (Code Access Security — CAS) oraz na rolach (Role-Based Security — RBS)*.
 - Połączenie z kodem niezarządzanym
 - Debugowanie,
 - Wspieranie procesu ustalania wersji programu..
 - Przygotowanie programu do uruchomienia

Działanie programu



Trzy sposoby tworzenia programu (1)

- Korzystanie z **API Windows** jako podstawowego interfejsu systemu operacyjnego Windows do komunikacji między systemem operacyjnym i aplikacją
- Korzystanie z biblioteki **Microsoft Foundation Classes** (MFC), która zawiera klasy C++ obejmujące API Windows
- Korzystanie z **Windows Forms** – sposób tworzenia aplikacji oparty na formularzach, która jest uruchamiana w środowisku CLR

Charakterystyka trzech podejść (2)

- **Windows API** - Należy napisać cały kod dla wszystkich elementów GUI (pracochłonny)
- **MFC** – korzystanie z gotowych kontrolek GUI, korzystając z metody wizualnego programowania. Należy jednak napisać sporo kodu do obsługi interakcji z użytkownikiem. Umożliwia to dużą kontrolę nad tworzonym GUI, większą niż w Windows Forms. Program działa natywnie na PC, więc nieznacznie ma lepszą wydajność
- **Windows Forms** – tworzenie GUI budując projekt metodą „przeciągnij i upuść” bez konieczności pisania dużej ilości kodu. Jest to najszybszy i najprostszy sposób budowania aplikacji, która działa w środowisku CLR, które pogarsza nieznacznie wydajność.

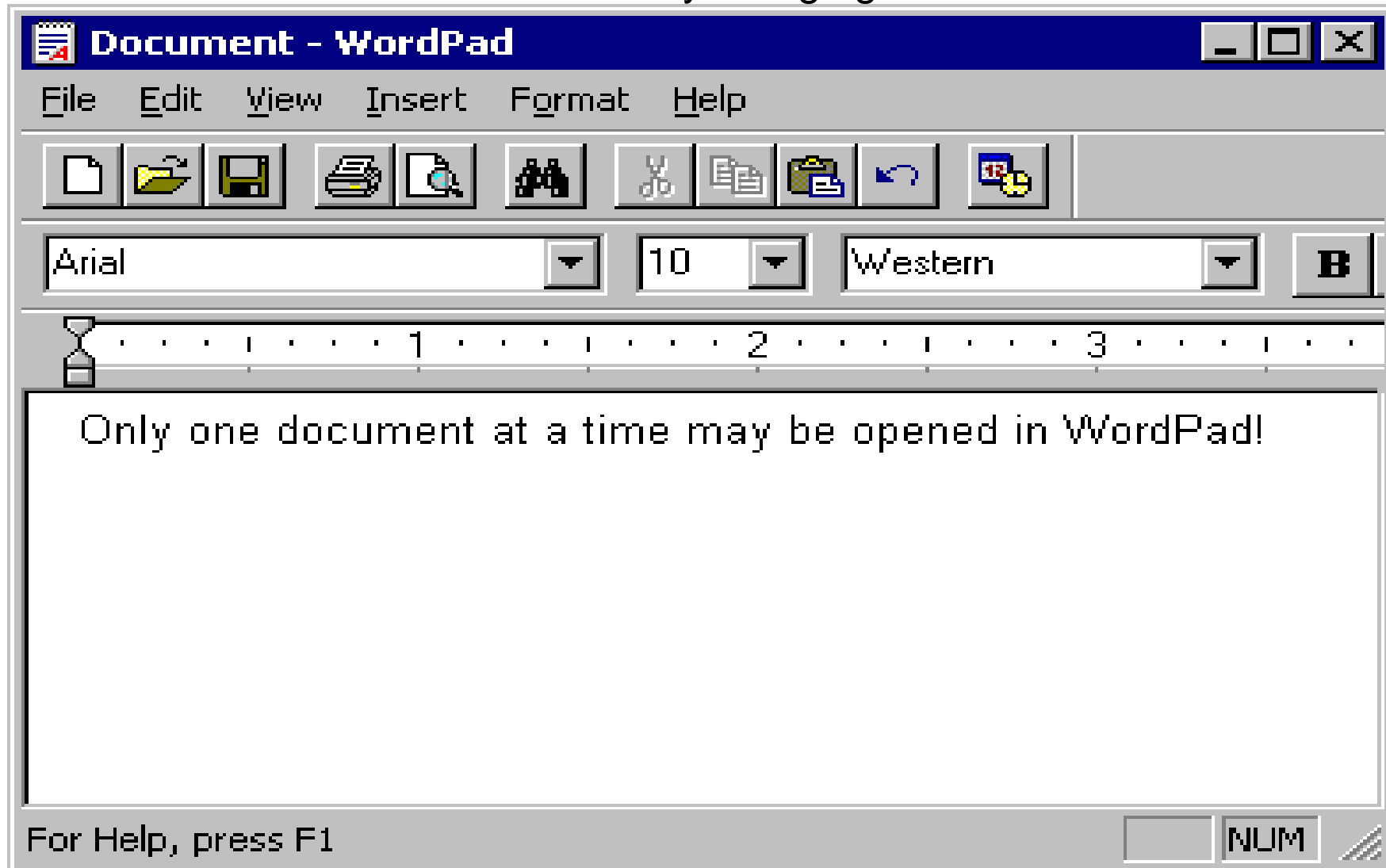
Style interfejsów graficznych programów Windows (1)

Trzy style

1. Interfejs typu jednodokumentowy *SDI*
(the *single-document interface*)
2. Interfejs typu wielodokumentowy *MDI*
(the *multiple-document interface*)
3. Interfejs typu „*Explorer-style*”

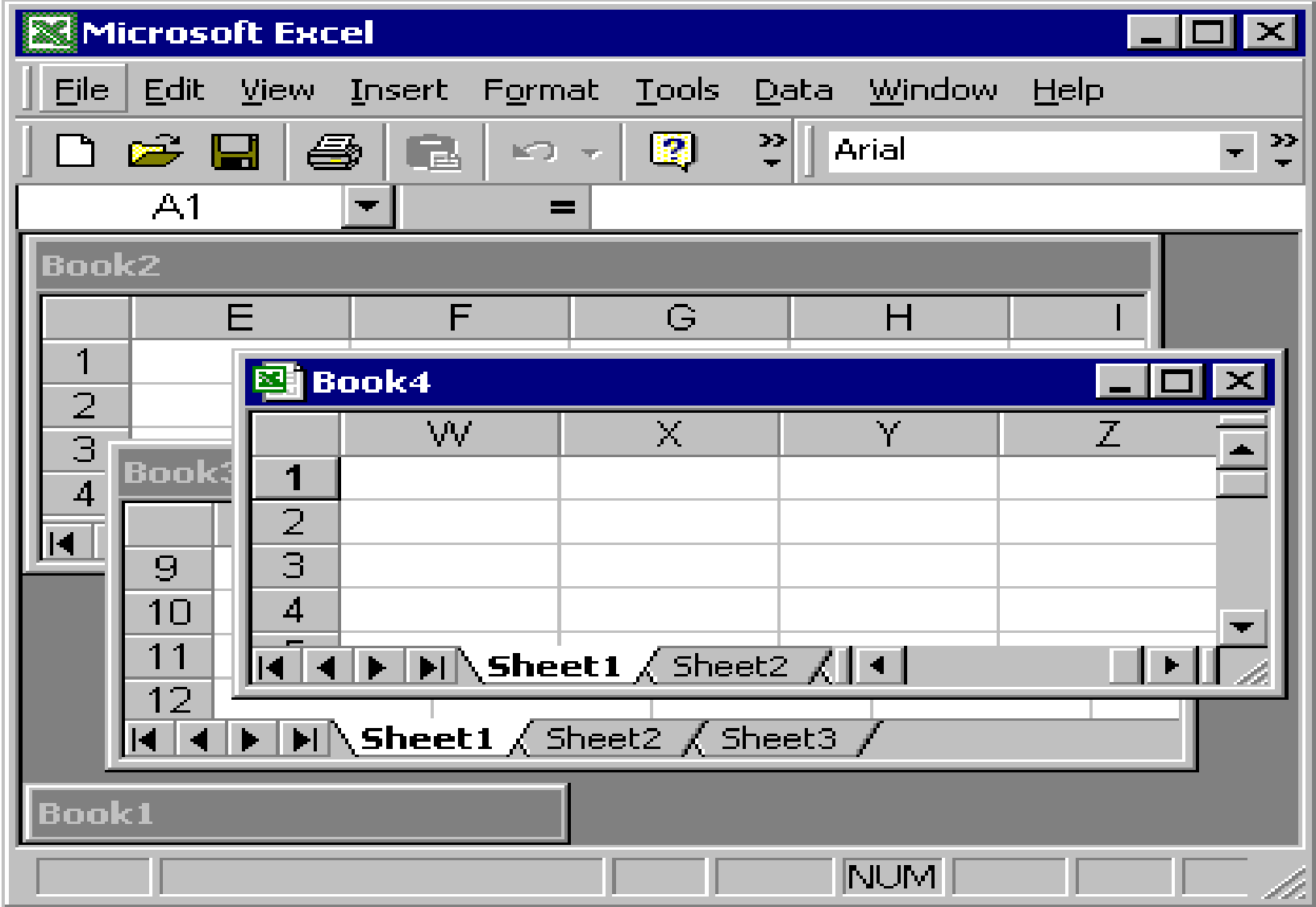
Word Pad– widok jednodokumentowego interfejsu programu SDI (2)

Można otworzyć tylko jeden dokument - jeśli jest otwarty jeden dokument, nie można otworzyć drugiego okna



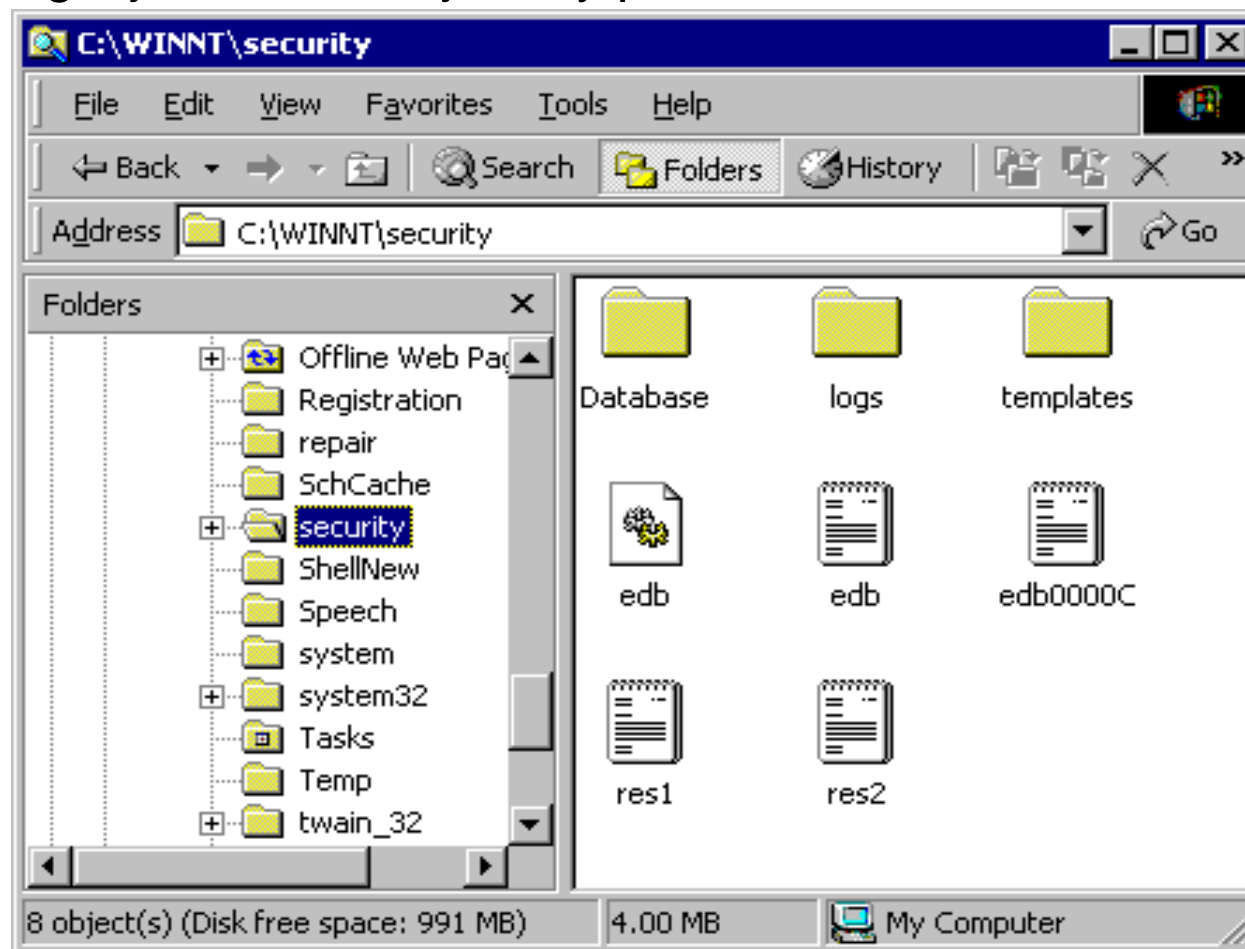
Microsoft Excel - widok wielodokumentowego interfejsu programu MDI (3)

W programie można jednocześnie wiele okien

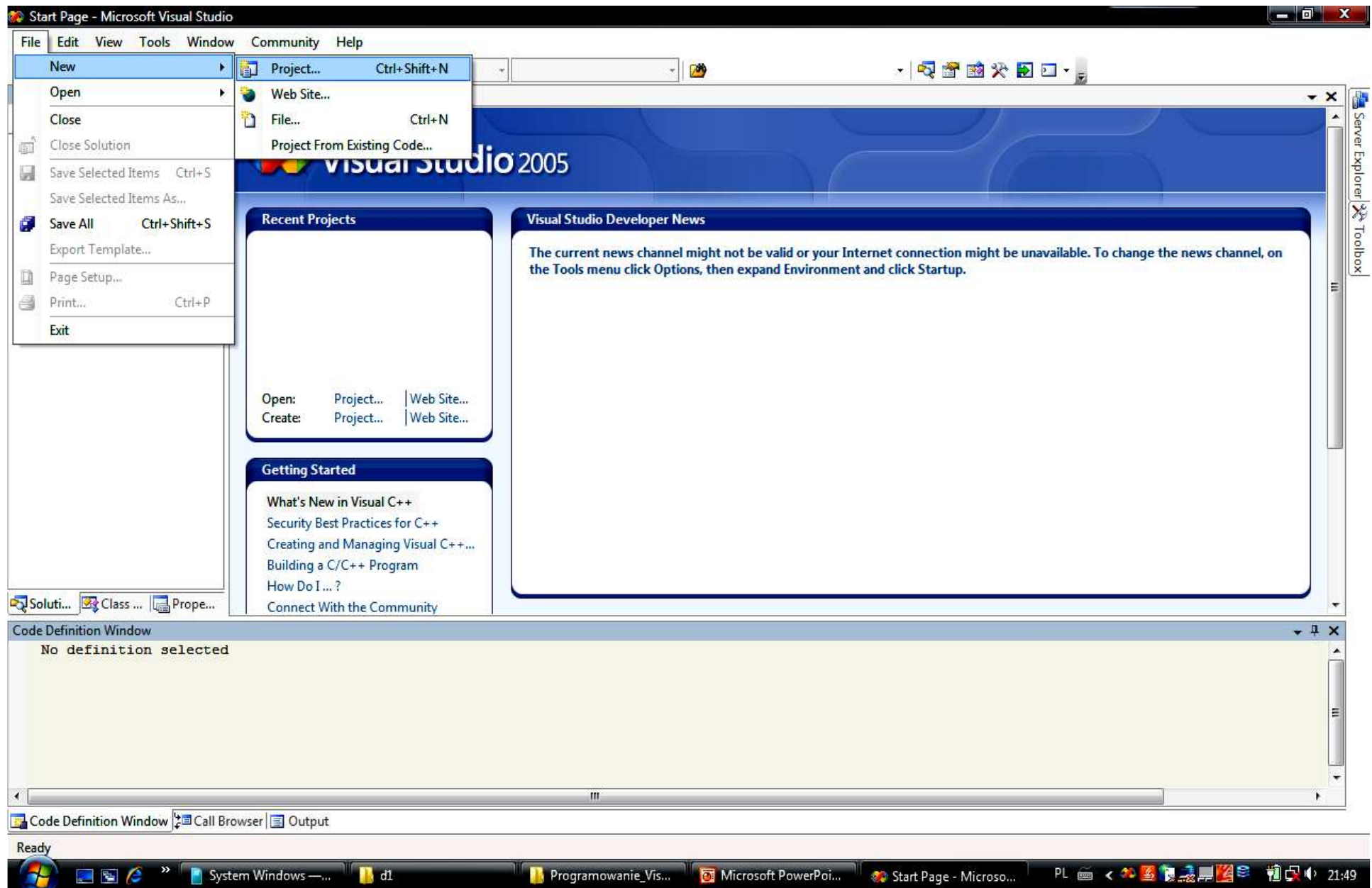


Windows Explorer - widok interfejsu programu typu „explorer-style” (4)

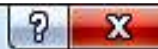
Pojedyncze okno zawierające dwa panele - w lewym panelu zawierające zazwyczaj drzewo lub hierarchiczny widok, w prawym panelu wyświetla zawartość obszaru zaznaczonego w lewym panelu. Służy do wyboru lub nawigacji wśród dużej liczby plików, dokumentów, obrazów



2. Tworzenie aplikacji w Windows Forms



New Project



Project types:

- Visual C++
 - ATL
 - CLR**
 - General
 - MFC
 - Smart Device
 - Win32
- Other Languages
- Other Project Types

Templates:



Visual Studio installed templates

- ASP.NET Web Service
- CLR Console Application
- SQL Server Project
- Windows Forms Control Library
- Class Library
- CLR Empty Project
- Windows Forms Application
- Windows Service

My Templates

- Search Online Templates...

A project for creating an XML Web Service

Name: <Enter_name>

Location: C:\Users\kruczkiewicz\Documents\Visual Studio 2005\Projects

Browse...

Solution Name: <Enter_name>

Create directory for solution

OK

Cancel

New Project



Project types:

- [-] Visual C++
 - ... ATL
 - ... CLR
 - ... General
 - ... MFC
 - ... Smart Device
 - ... Win32
- [+] Other Languages
- [+] Other Project Types

Templates:



- Visual Studio installed templates
- ASP.NET Web Service
 - CLR Console Application
 - SQL Server Project
 - Windows Forms Control Library
 - Class Library
 - CLR Empty Project
 - Windows Forms Application**
 - Windows Service
- My Templates
- Search Online Templates...

A project for creating an application with a Windows user interface

Name: <Enter_name>

Location: E:\Dydaktyka\d1\Programowanie_VisualCPLUS\Wyklad1

Browse...

Solution: Create new Solution

Create directory for solution

Solution Name: <Enter_name>

OK

Cancel

New Project



Project types:

- [-] Visual C++
 - ... ATL
 - ... CLR
 - ... General
 - ... MFC
 - ... Smart Device
 - ... Win32
- [+] Other Languages
- [+] Other Project Types

Templates:



- Visual Studio installed templates
- ASP.NET Web Service
 - CLR Console Application
 - SQL Server Project
 - Windows Forms Control Library
 - Class Library
 - CLR Empty Project
 - Windows Forms Application
 - Windows Service
- My Templates
- Search Online Templates...

A project for creating an application with a Windows user interface

Name: WForms1

Location: E:\Dydaktyka\d1\Programowanie_VisualCPLUS\ProjektyWForms

Browse...

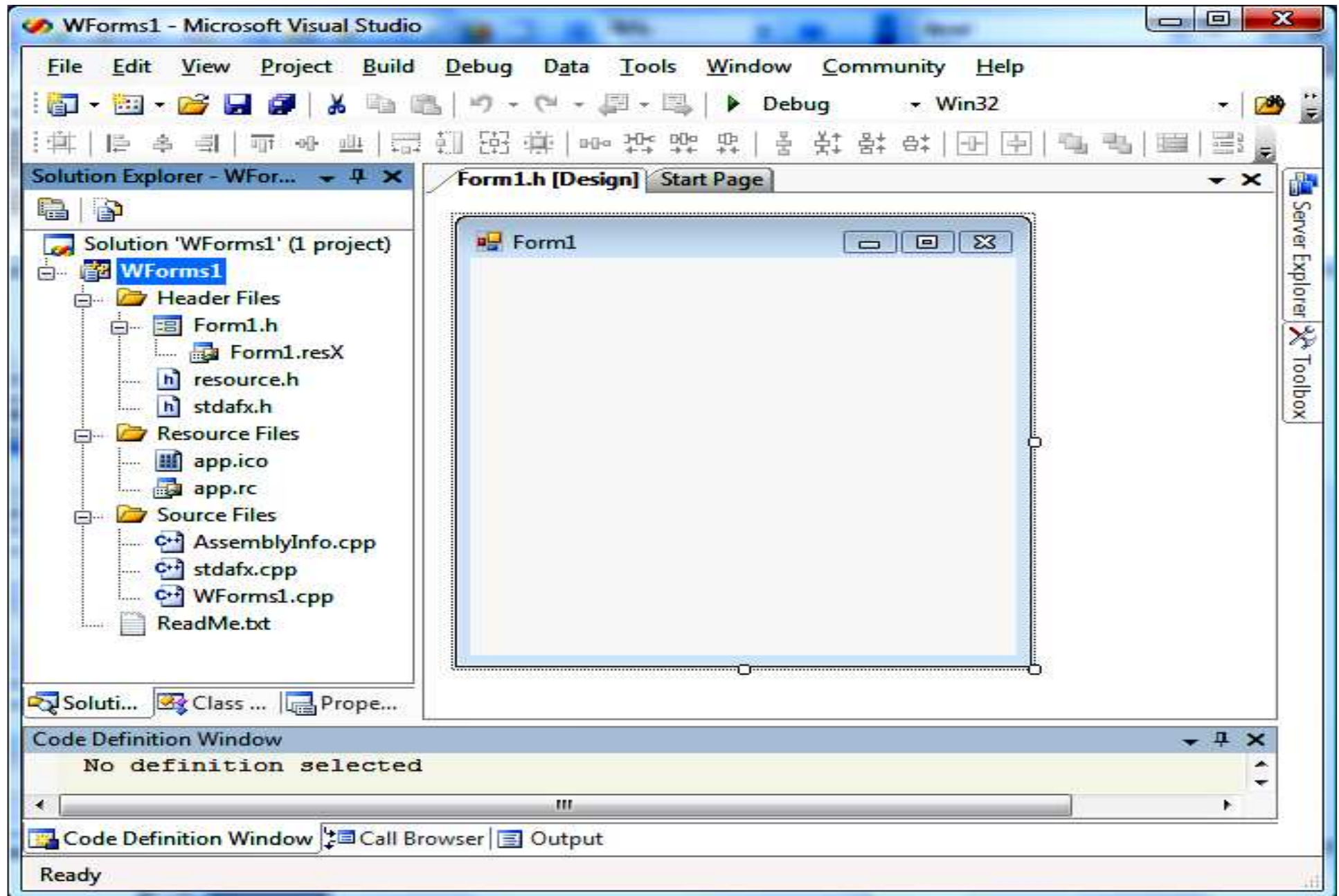
Solution Name: WForms1

Create directory for solution

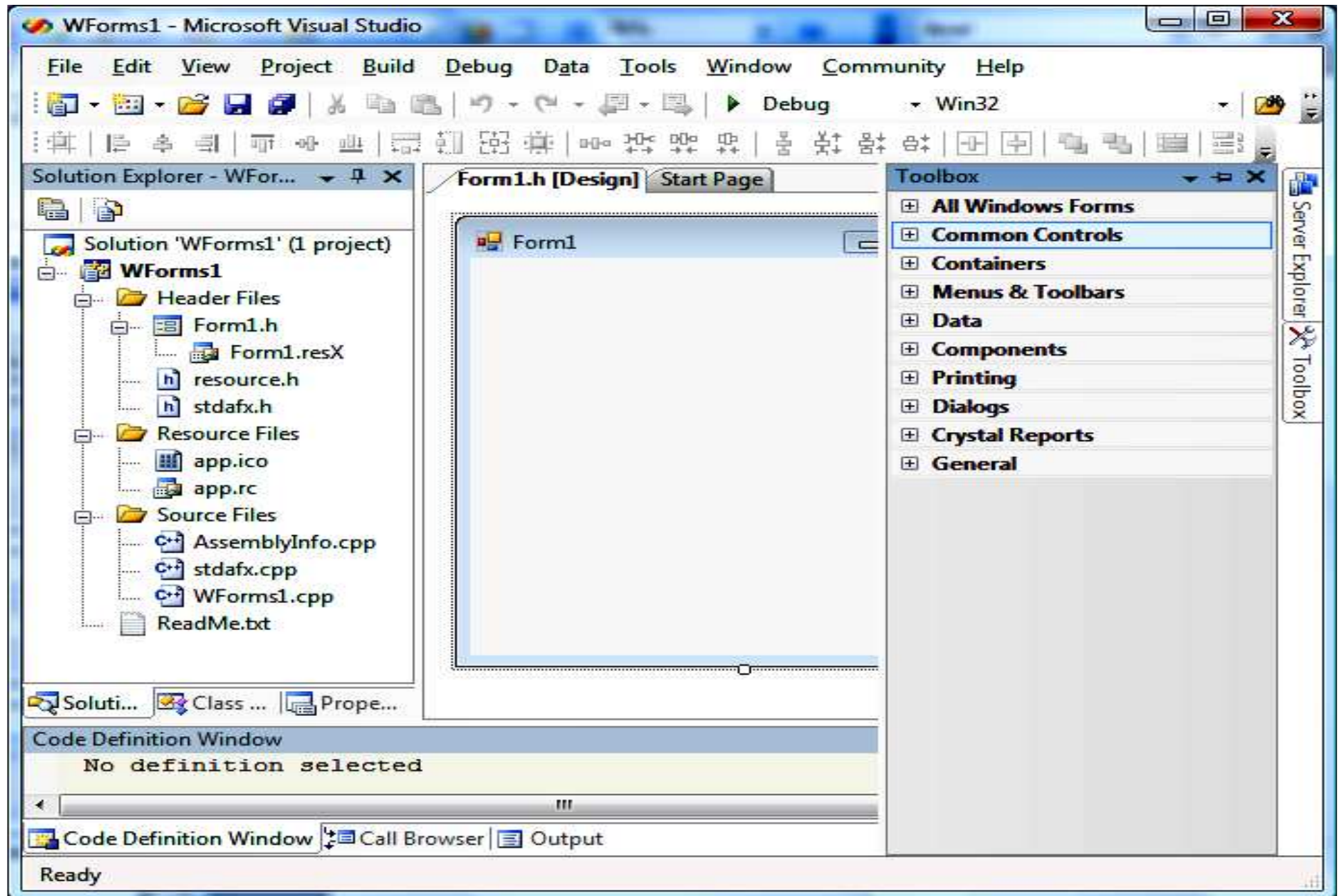
OK

Cancel

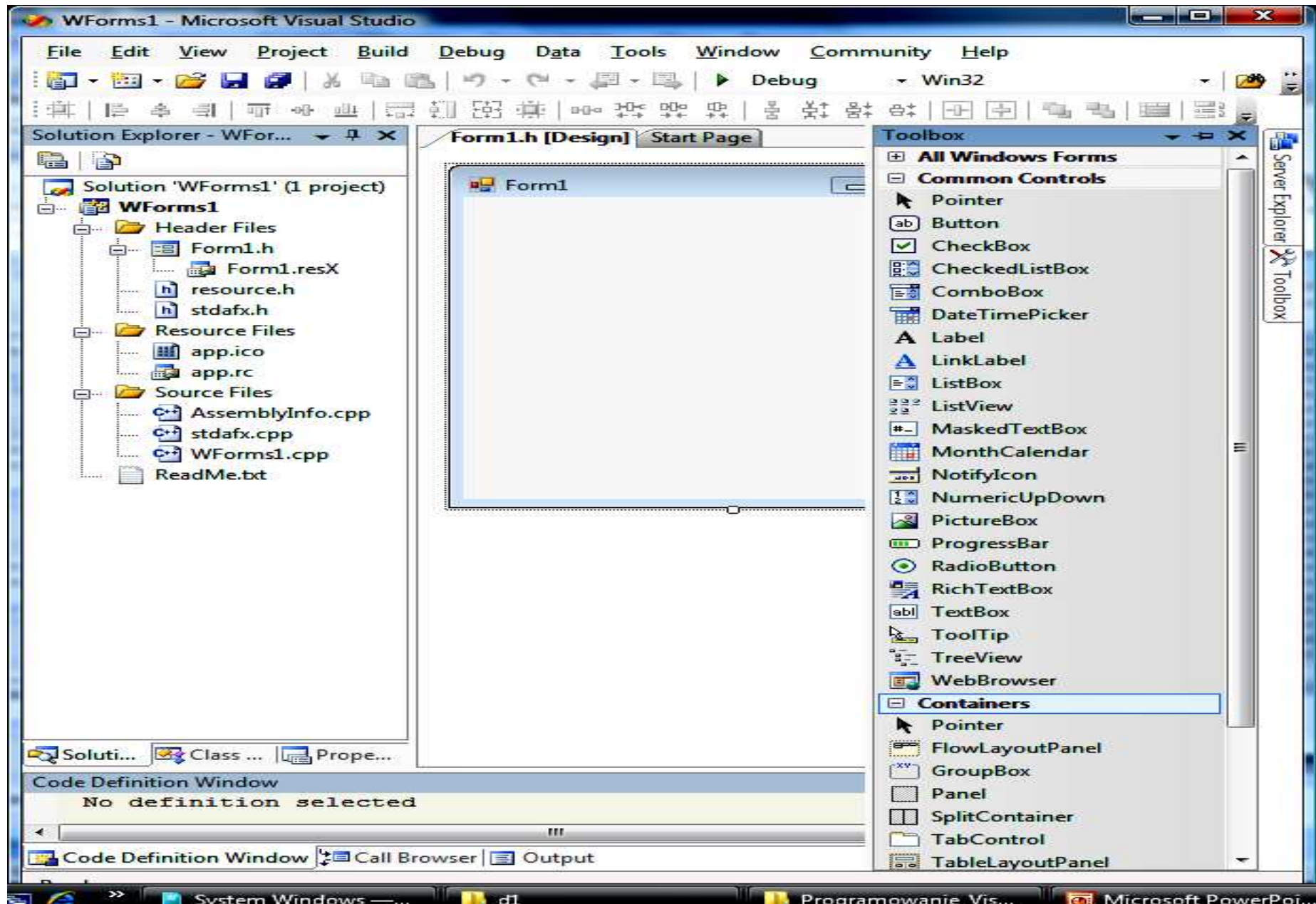
Widok projektu CLR Windows Forms

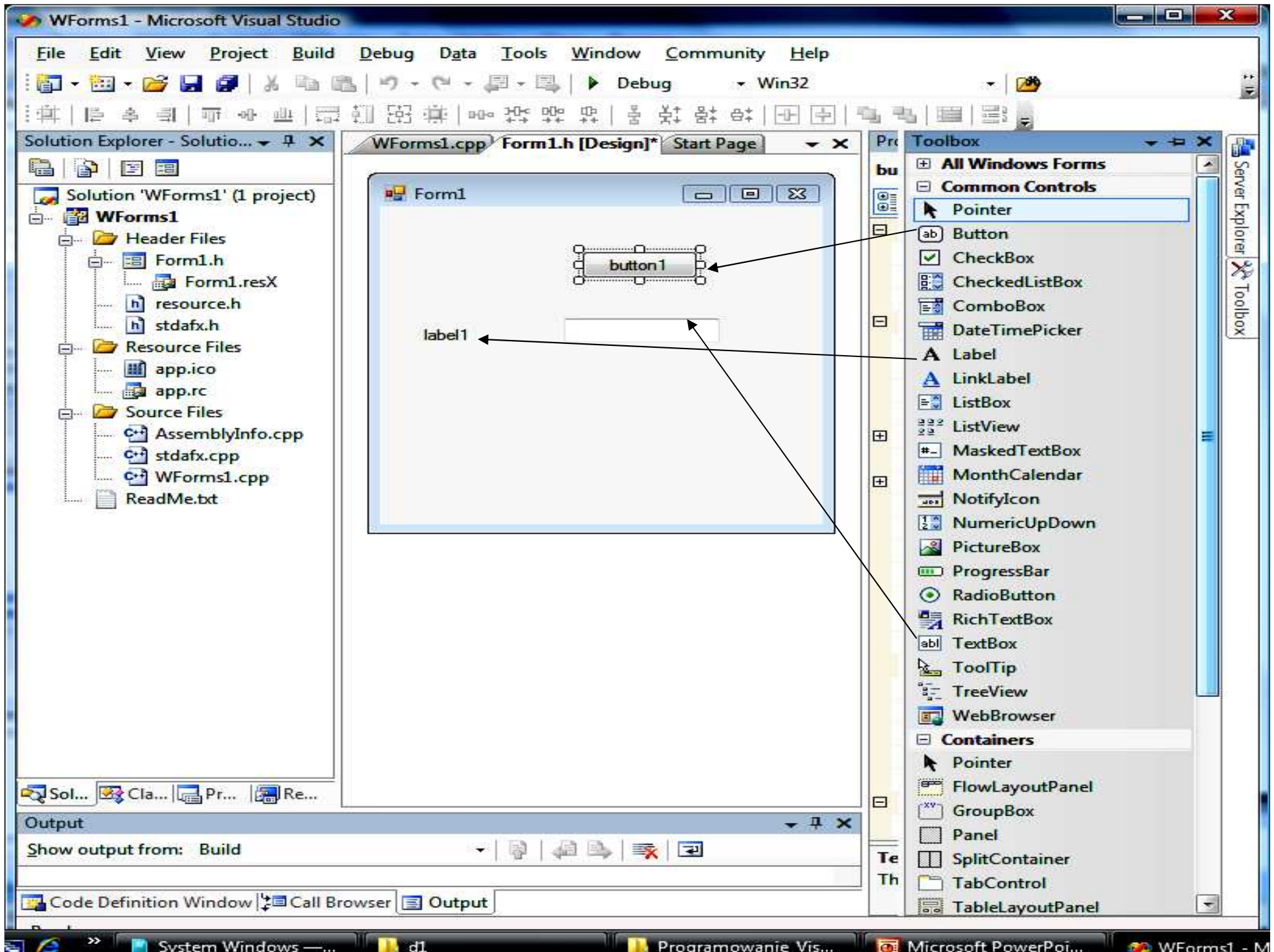


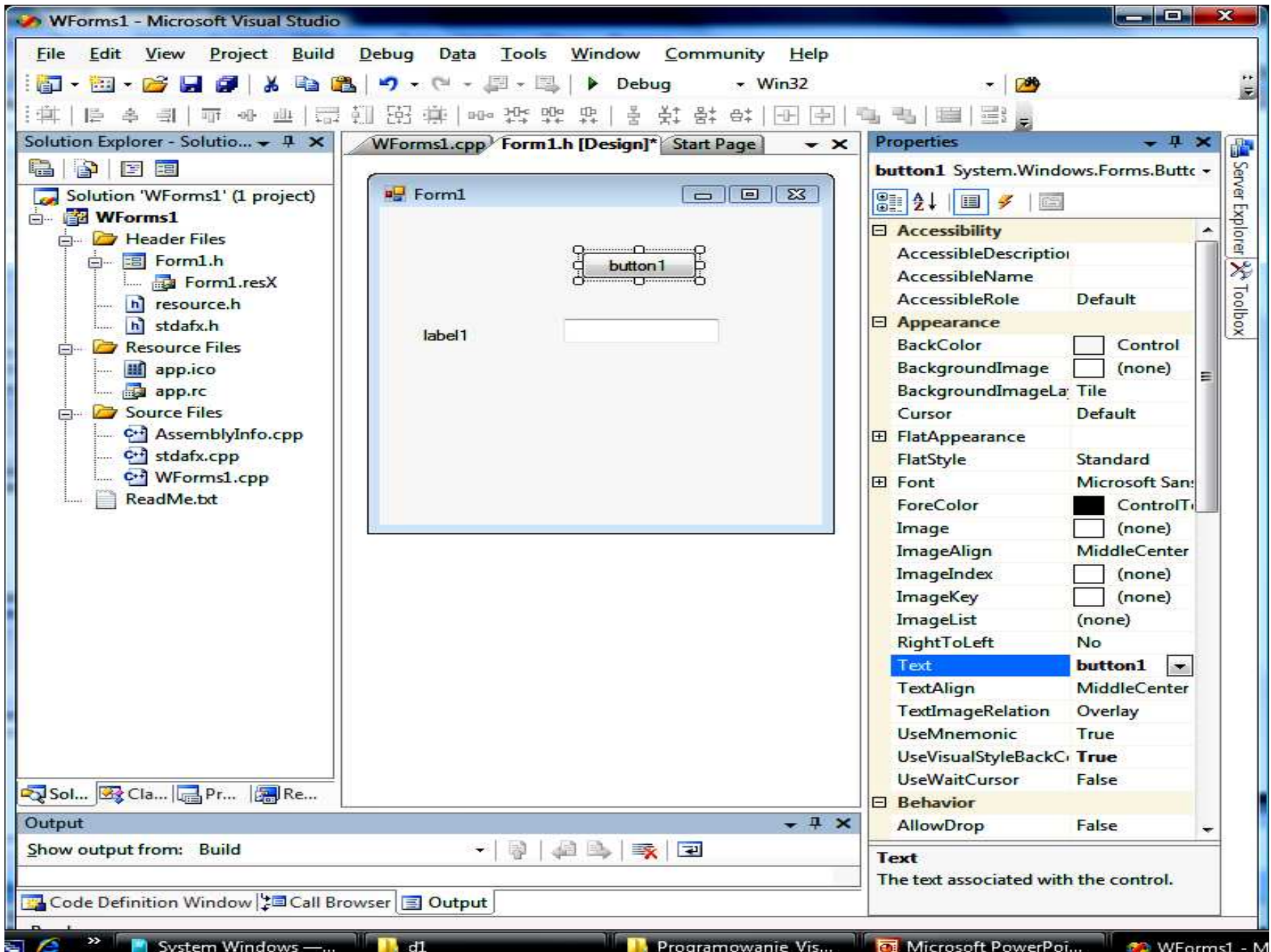
Paleta kontrol Windows Forms (1)

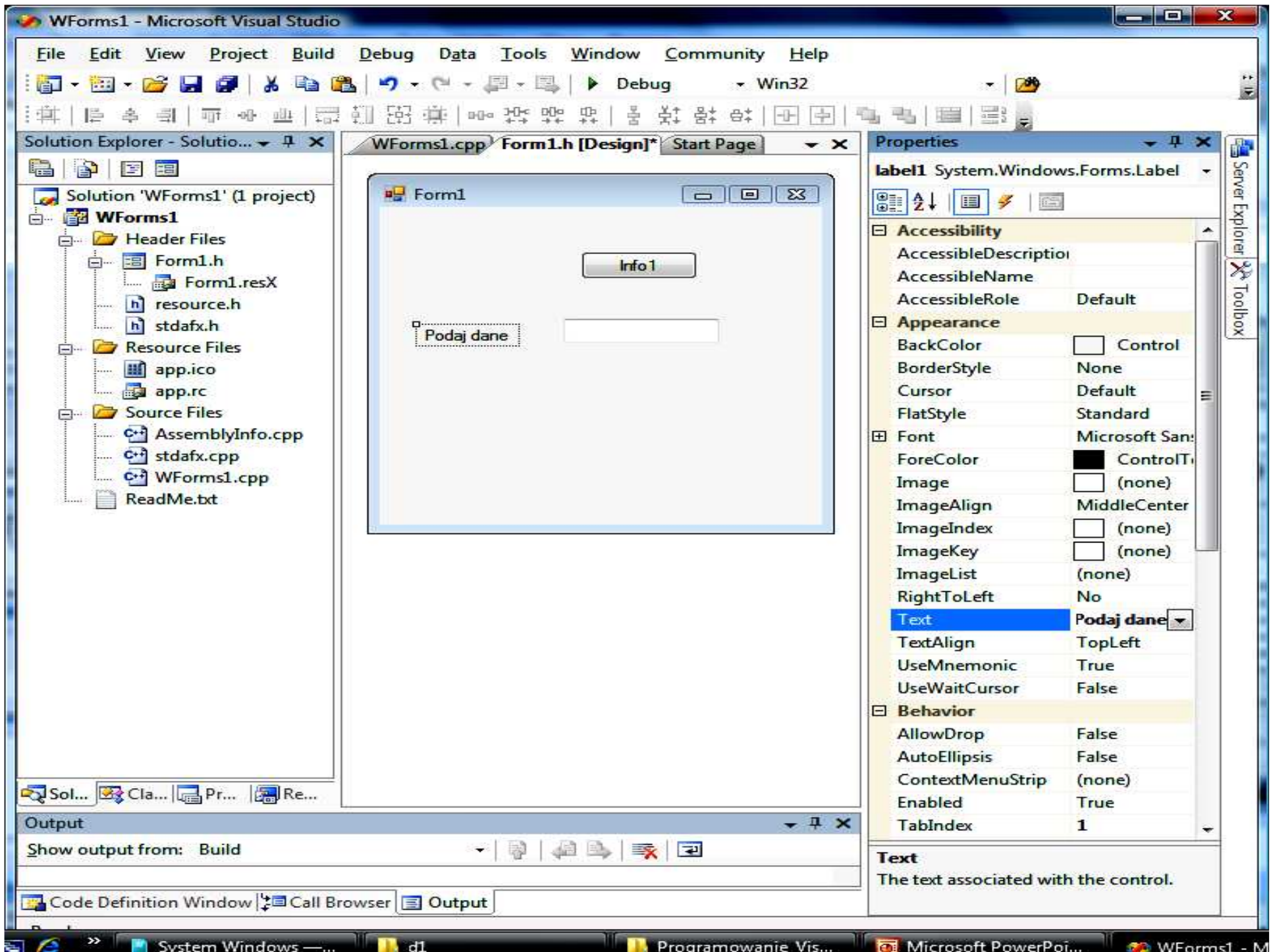


Paleta kontrol Windows Forms (2)









WForms1 - Microsoft Visual Studio

File Edit View Project Build Debug Data Tools Window Community Help

Solution Explorer - Solution Explorer

Solution 'WForms1' (1 project)

- WForms1
 - Header Files
 - Form1.h
 - Form1.resX
 - resource.h
 - stdafx.h
 - Resource Files
 - app.ico
 - app.rc
 - Source Files
 - AssemblyInfo.cpp
 - stdafx.cpp
 - WForms1.cpp
 - ReadMe.txt

WForms1.cpp Form1.h [Design]*

Pierwszy program

Info1

Podaj dane

Properties

Form1 System.Windows.Forms.Form

- Accessibility
 - AccessibleDescription
 - AccessibleName
 - AccessibleRole Default
- Appearance
 - BackColor Control
 - BackgroundImage (none)
 - BackgroundImageLayout Tile
 - Cursor Default
 - Font Microsoft Sans Serif; 8,
 - ForeColor ControlText
 - FormBorderStyle Sizable
 - RightToLeft No
 - RightToLeftLayout False
 - Text Pierwszy program**
 - UseWaitCursor False
- Behavior
 - AllowDrop False
 - AutoValidate EnablePreventFocusCl
 - ContextMenuStrip (none)
 - DoubleBuffered False
 - Enabled True
 - ImeMode NoControl
- Data
 - (ApplicationSettings)
 - (DataBindings)
 - Tag

Text

The text associated with the control.

Po kliknięciu prawym klawiszem myszy na kontrolkę można wybrać okno Properties

Output

Show output from: Build

Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped

Code Definition Window Call Browser Output

System Window... d1 Programowanie... Microsoft Powe... WForms1 - Mic... Form1

Solution Explorer - Solution...

- Solution 'WForms1' (1 project)
 - WForms1
 - Header Files
 - Form1.h
 - Form1.resX
 - resource.h
 - stdafx.h
 - Resource Files
 - app.ico
 - app.rc
 - Source Files
 - AssemblyInfo.cpp
 - stdafx.cpp
 - WForms1.cpp
 - ReadMe.txt

WForms1.cpp Form1.h [Design]* Start Page

Pierwszy program

Info1

Podaj dane

Properties

label1 System.Window

- Appearance
 - BackColor
 - BorderStyle
 - Cursor
 - FlatStyle
- Font
 - ForeColor
 - Image
 - ImageAlign
 - ImageIndex
 - ImageKey
 - ImageList
 - RightToLeft
 - Text
 - TextAlign
 - UseMnemonic
 - UseWaitCursor
- Behavior
 - AllowDrop
 - AutoEllipsis
 - ContextMenuStrip
 - Enabled
 - TabIndex

Toolbox

- MaskedTextBox
- MonthCalendar
- NotifyIcon
- NumericUpDown
- PictureBox
- ProgressBar
- RadioButton
- RichTextBox
- TextBox
- ToolTip
- TreeView
- WebBrowser
- Containers
 - Pointer
 - FlowLayoutPanel
 - GroupBox
 - Panel
 - SplitContainer
 - TabControl
 - TableLayoutPanel
- Menus & Toolbars
 - Pointer
 - ContextMenuStrip
 - MenuStrip

Output

Show output from: Build

Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped

Code Definition Window Call Browser Output

Solution Explorer - Solution Explorer

- Solution 'WForms1' (1 project)
 - WForms1
 - Header Files
 - Form1.h
 - Form1.resX
 - resource.h
 - stdafx.h
 - Resource Files
 - app.ico
 - app.rc
 - Source Files
 - AssemblyInfo.cpp
 - stdafx.cpp
 - WForms1.cpp
 - ReadMe.txt

WForms1.cpp Form1.h [Design]* Start Page

Pierwszy program

Type Here

Info1

Podaj dane

menuStrip1

- View Code
- Embed in ToolStripContainer
- Insert Standard Items**
- Edit Items...
- Cut
- Copy
- Paste
- Delete
- Properties

Properties

menuStrip1 System.Windows.Forms.MenuStrip

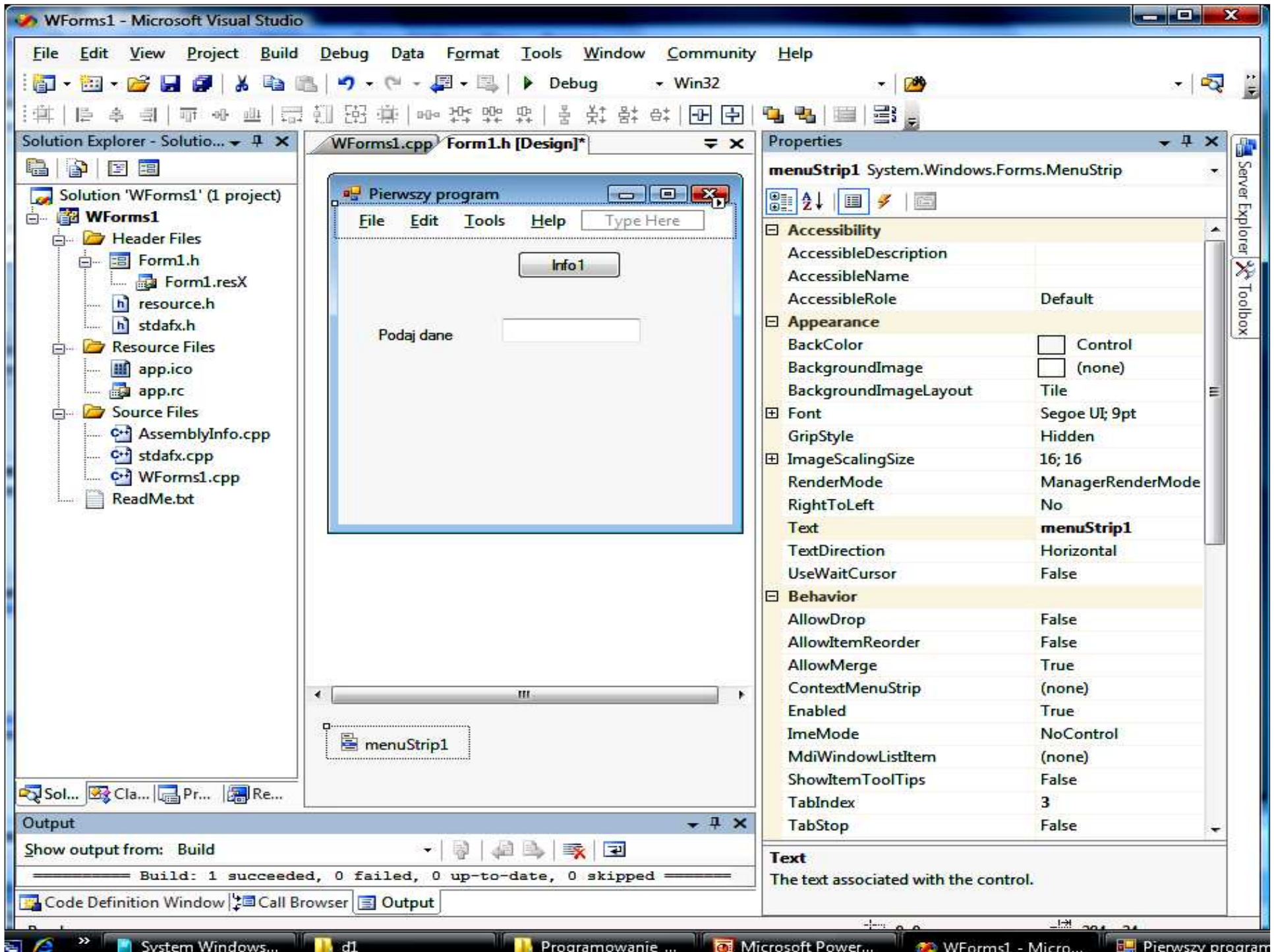
- Accessibility
 - AccessibleDescription
 - AccessibleName
 - AccessibleRole Default
- Appearance
 - BackColor
 - BackgroundImage
 - BackgroundImageLayout Tile
 - Font Segoe UI; 9pt
 - GripStyle Hidden
 - ImageScalingSize 16; 16
 - RenderMode ManagerRenderMode
 - RightToLeft No
 - Text **menuStrip1**
 - TextDirection Horizontal
 - UseWaitCursor False
- Behavior
 - AllowDrop False
 - AllowItemReorder False
 - AllowMerge True
 - ContextMenuStrip (none)
 - Enabled True
 - ImeMode NoControl
 - MdiWindowListItem (none)
 - ShowItemToolTips False
 - TabIndex 3
 - TabStop False

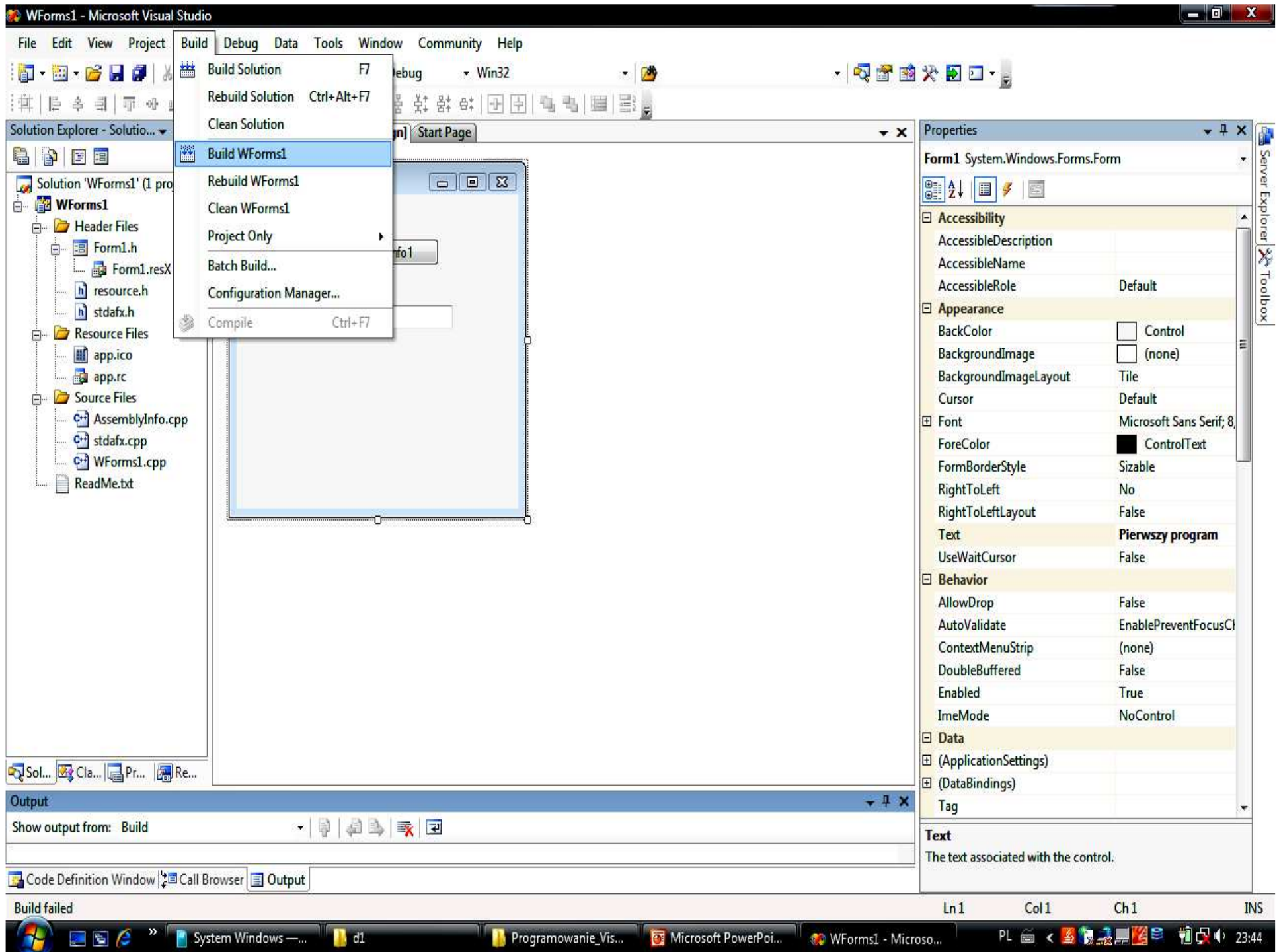
Text
The text associated with the control.

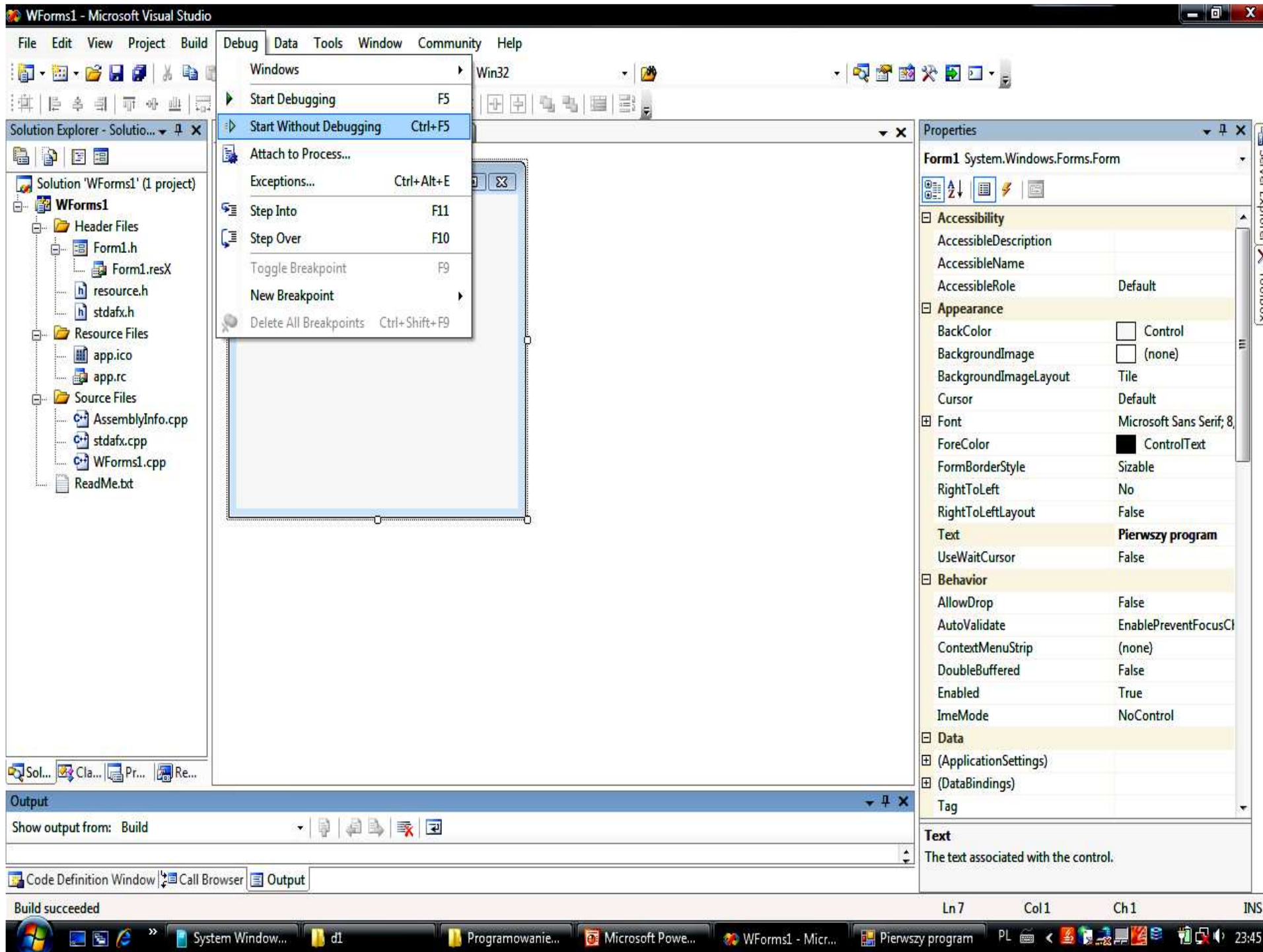
Output

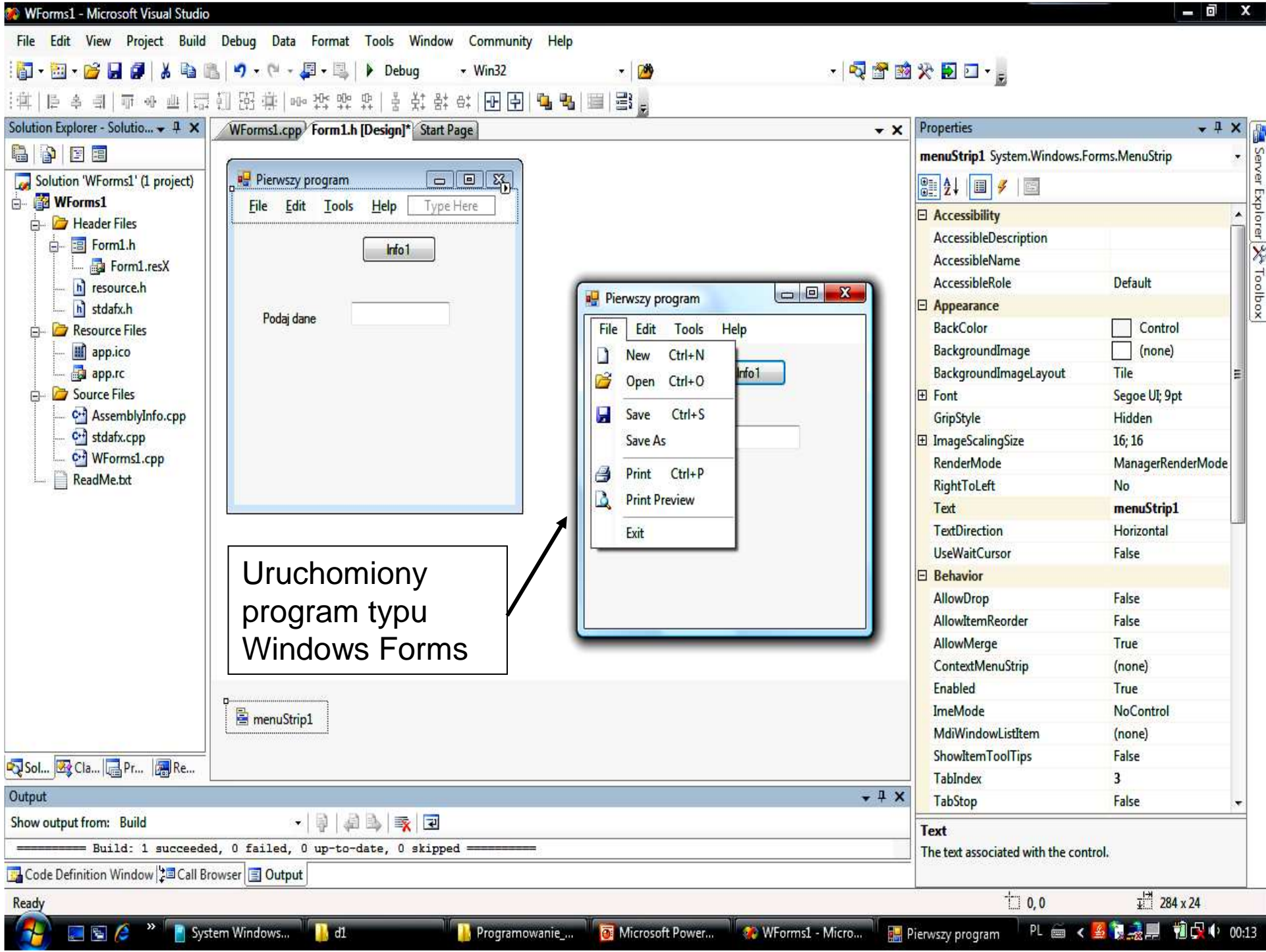
Show output from: Build

Code Definition Window Call Browser Output









Solution 'WForms1' (1 project)

- WForms1
 - Header Files
 - Form1.h
 - Form1.resX
 - resource.h
 - stdafx.h
 - Resource Files
 - app.ico
 - app.rc
 - Source Files
 - AssemblyInfo.cpp
 - stdafx.cpp
 - WForms1.cpp
 - ReadMe.txt

Pierwszy program

File Edit Tools Help Type Here

Info1

Podaj dane

Pierwszy program

File Edit Tools Help

- New Ctrl+N
- Open Ctrl+O
- Save Ctrl+S
- Save As
- Print Ctrl+P
- Print Preview
- Exit

Uruchomiony program typu Windows Forms

menuStrip1 System.Windows.Forms.MenuStrip

Accessibility	
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
Appearance	
BackColor	<input type="checkbox"/> Control
BackgroundImage	<input type="checkbox"/> (none)
BackgroundImageLayout	Tile
Font	
GripStyle	Hidden
ImageScalingSize	16; 16
RenderMode	ManagerRenderMode
RightToLeft	No
Text	
Text	menuStrip1
TextDirection	Horizontal
UseWaitCursor	False
Behavior	
AllowDrop	False
AllowItemReorder	False
AllowMerge	True
ContextMenuStrip	(none)
Enabled	True
ImeMode	NoControl
MdiWindowListItem	(none)
ShowItemToolTips	False
TabIndex	3
TabStop	False

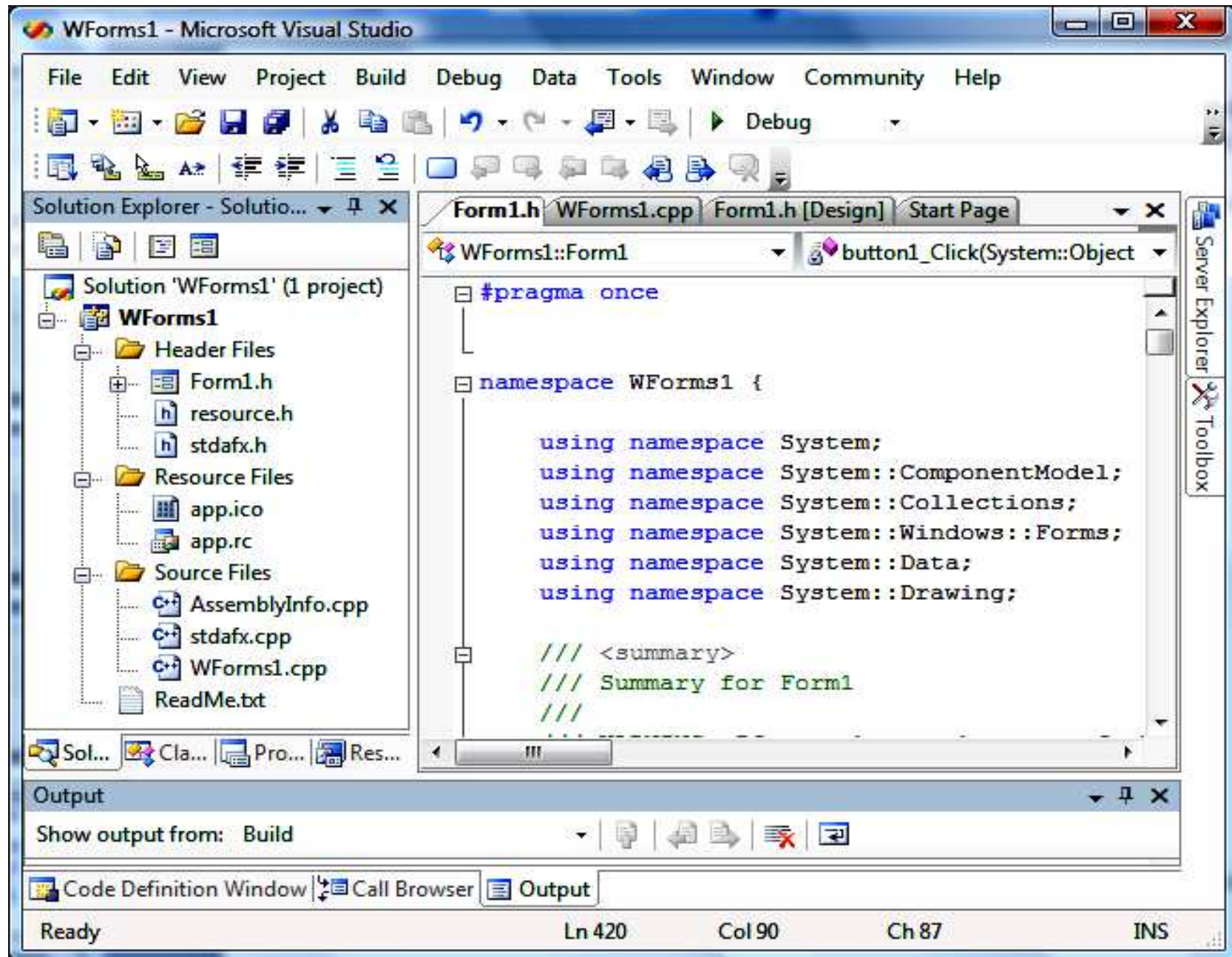
Show output from: Build

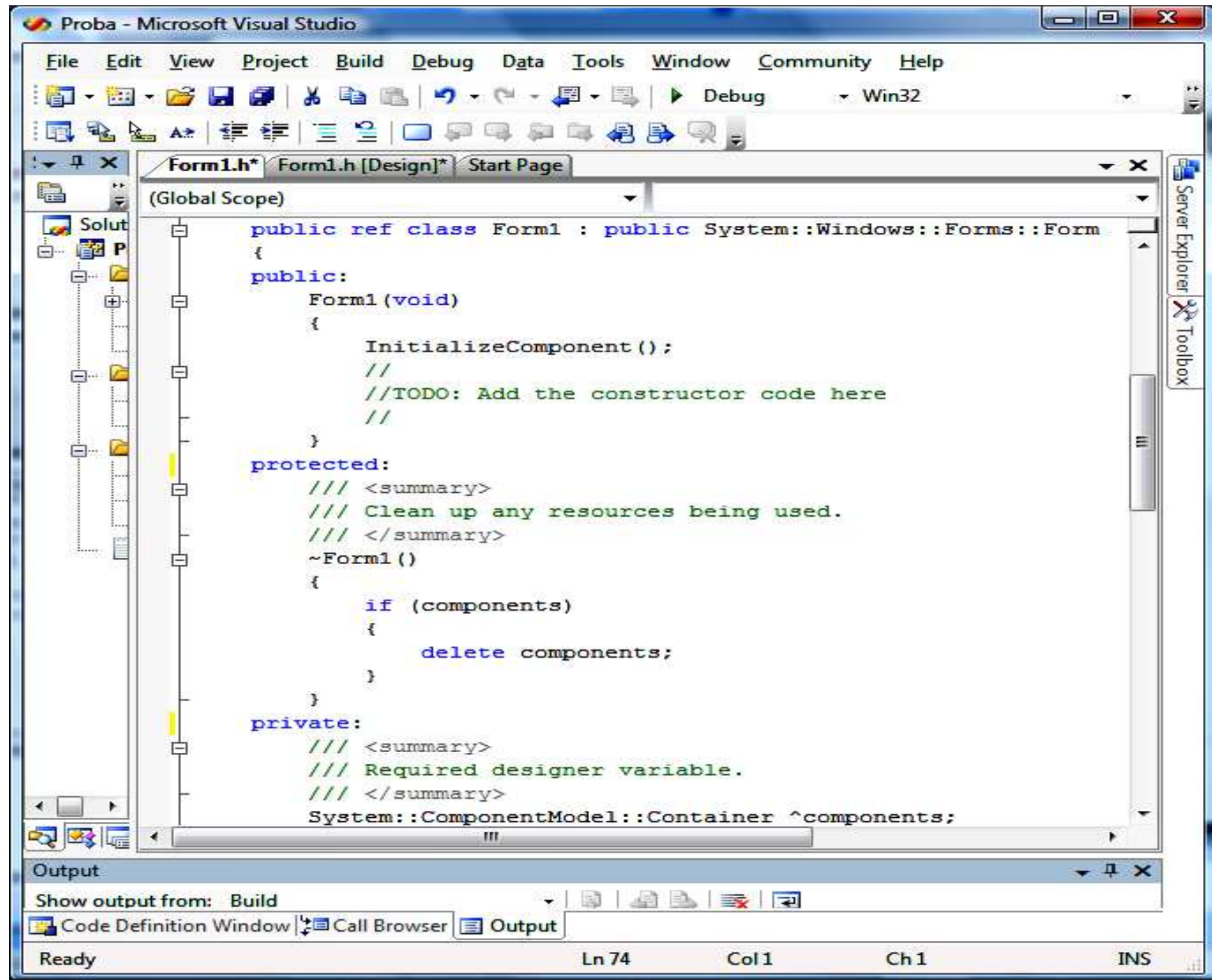
Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped

Podsumowanie: Czynności podczas tworzenia aplikacji

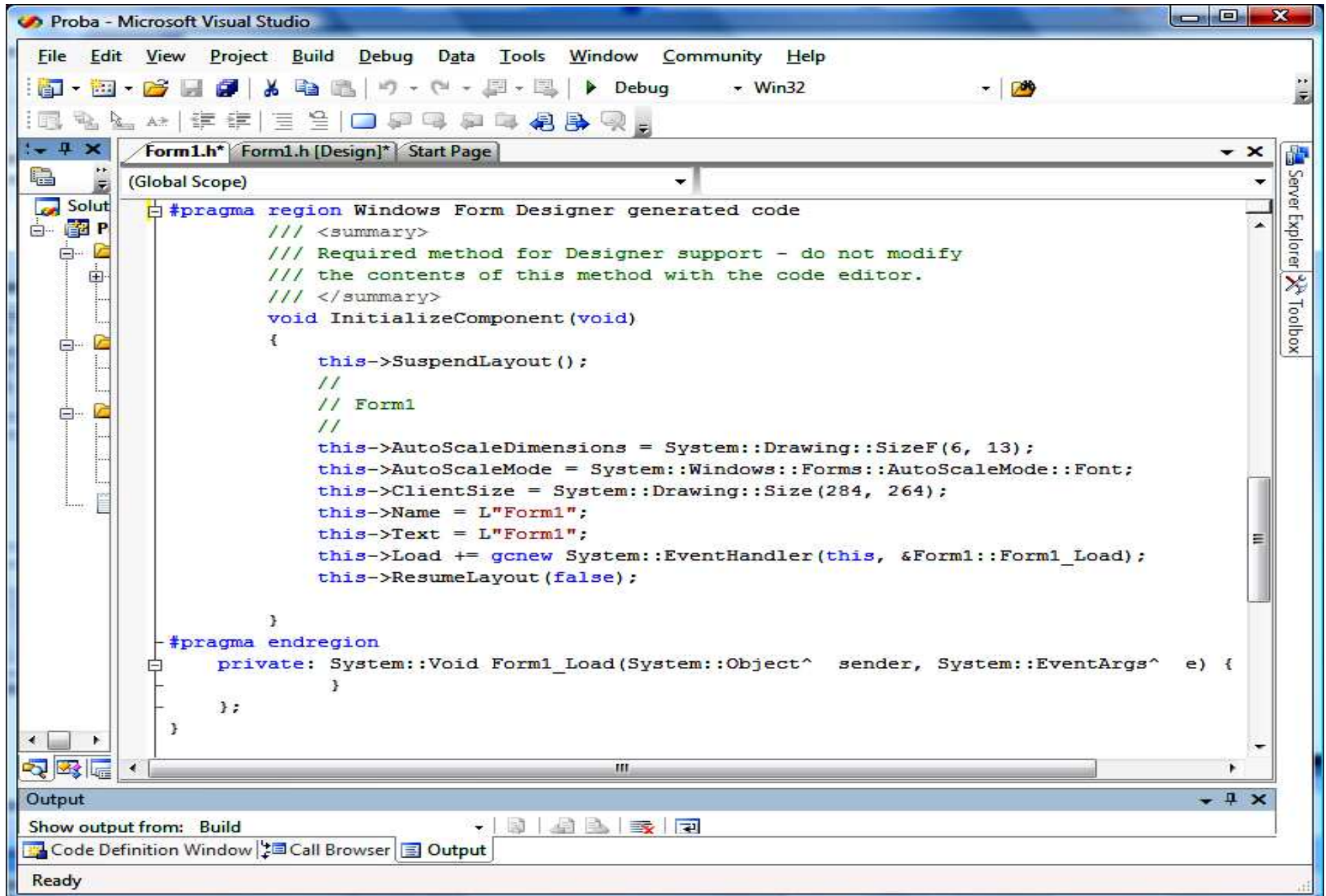
- W trybie *Form Design* stosowanie metody „przeciągnij i upuść” – przeciąganie komponentów z okna *Toolbox* (generowanie automatyczne kodu)
- W trybie *Form Design* modyfikowanie właściwości komponentów w oknie *Properties* (generowanie automatyczne kodu)
- Uzupełnianie ręczne kodu w plikach źródłowych aplikacji dotyczących obsługi zdarzeń komponentów oraz modyfikowanie klas utworzonych automatycznie w projekcie

3. Zawartość projektu

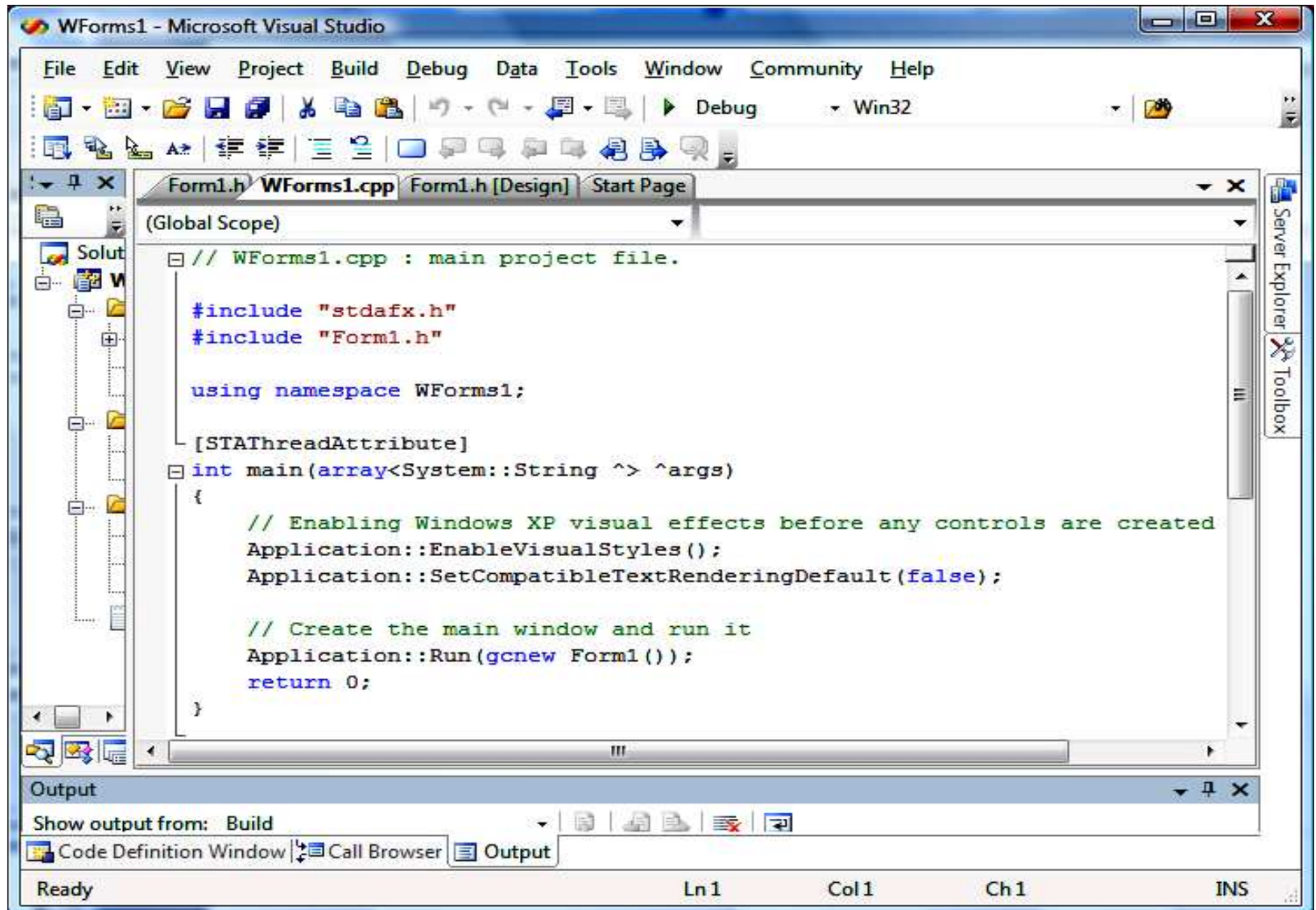




Początkowa zawartość pliku – przed wstawieniem komponentów



Główny plik



4. Podstawowe cechy języka C++/CLI

- **Klasy wartości lub referencji** nie może zawierać składowych będących tablicami oraz klas w natywnym C++
- Nie można używać funkcji zaprzyjaźnionych
- Nie wolno używać pól bitowych
- Nie ma składowych typu `const`
- W definicji konstruktora kopiującego
- Nie można przesłaniać domyślnego operatora przypisania w klasie wartości
- Brakuje domyślnego operatora przypisania dla klas referencyjnych
- Nie można przeciążać operatora **gcnew**
- Przeciążone operatory można realizować za pomocą metod statycznych i zwykłych
- Składowe typu **literal** – definiowanie stałych wewnątrz klasy
- Składowe typu **initonly** – można je inicjować tylko podczas tworzenia obiektu
- Oprócz zwykłych konstruktorów (metod o nazwie klasy bez zwracanego typu) bezparametrowych i z parametrami, które są wywoływane podczas tworzenia obiektów klas wartości i referencyjnych istnieją konstruktory typu **static** wywoływane niejawnie zawsze przed jawnym wywołaniem zwykłych konstruktorów/ Umożliwiają inicjowanie pól **initonly** podczas działania programu. Pola **initonly** inicjowane przez przypisanie są inicjowane przed uruchomieniem programu podczas kompilacji programu.

Typy klas w C++/CLI

Klasy wartości – zmienne tych klas, czyli obiekty posiadają własne dane

value struct (domyślnie składowe są publiczne),

value class (domyślnie składowe są prywatne)

Klasy referencji – zmienne tych klas są uchwytami do obiektów, którym należy przydzielać pamięć za pomocą operatora **gcnew**

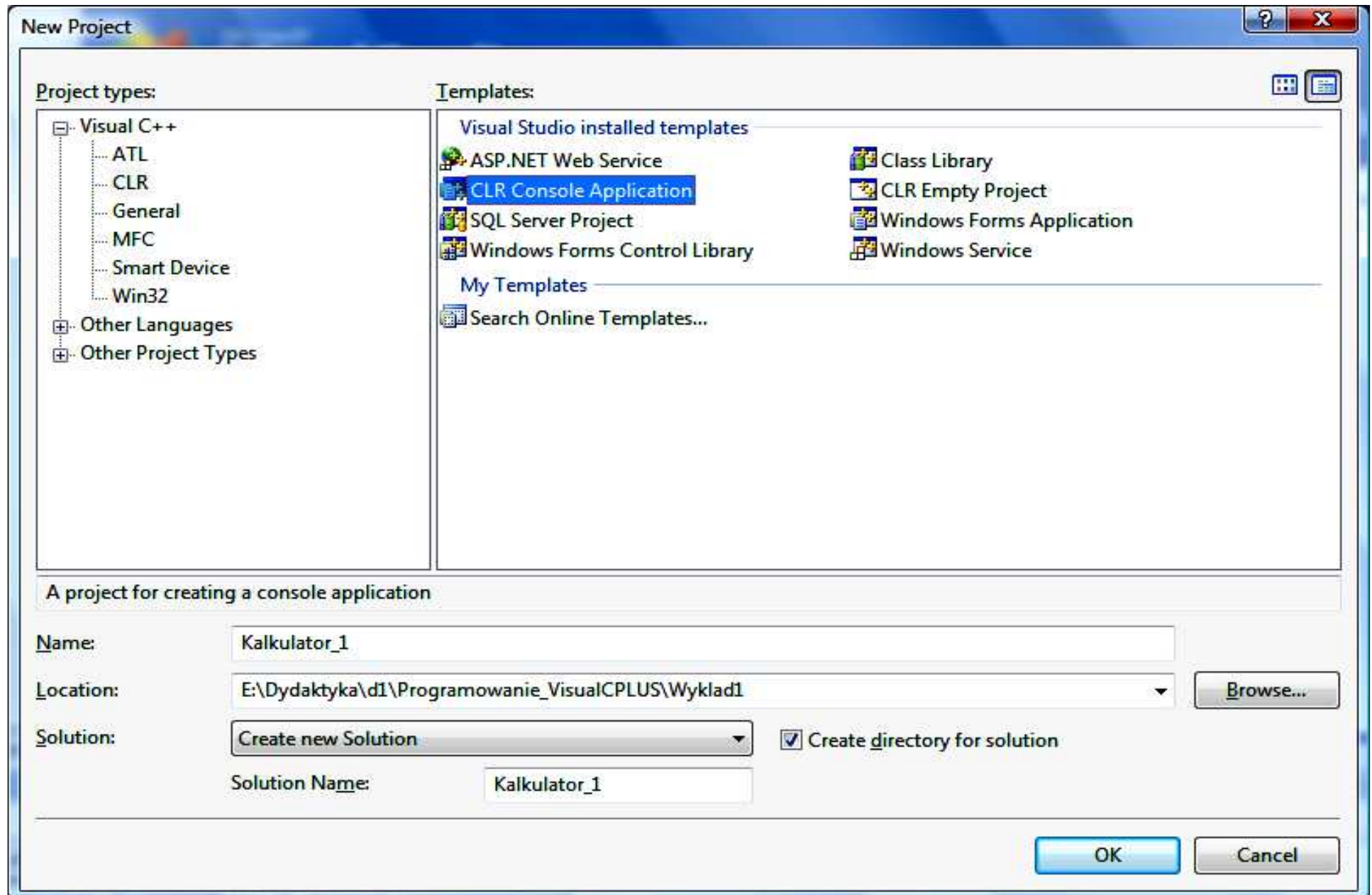
ref struct (domyślnie składowe są publiczne),

ref class (domyślnie składowe są prywatne)

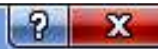
5. Najprostsza aplikacja - kalkulator

- Wykonanie klasy kalkulator w trybie konsolowym - wykonanie modułu
- Testowanie kalkulatora w trybie konsolowym – wykonanie projektu konsolowego
- Wykonanie interfejsu kalkulatora w trybie Window Forms

Tworzenie programu konsolowego CLR



New Project



Project types:

- [-] Visual C++
 - ... ATL
 - ... CLR
 - ... General
 - ... MFC
 - ... Smart Device
 - ... Win32
- [+] Other Languages
- [+] Other Project Types

Templates:



- Visual Studio installed templates
- ASP.NET Web Service
 - CLR Console Application
 - SQL Server Project
 - Windows Forms Control Library
 - Class Library
 - CLR Empty Project
 - Windows Forms Application
 - Windows Service
- My Templates
- Search Online Templates...

A project for creating an application with a Windows user interface

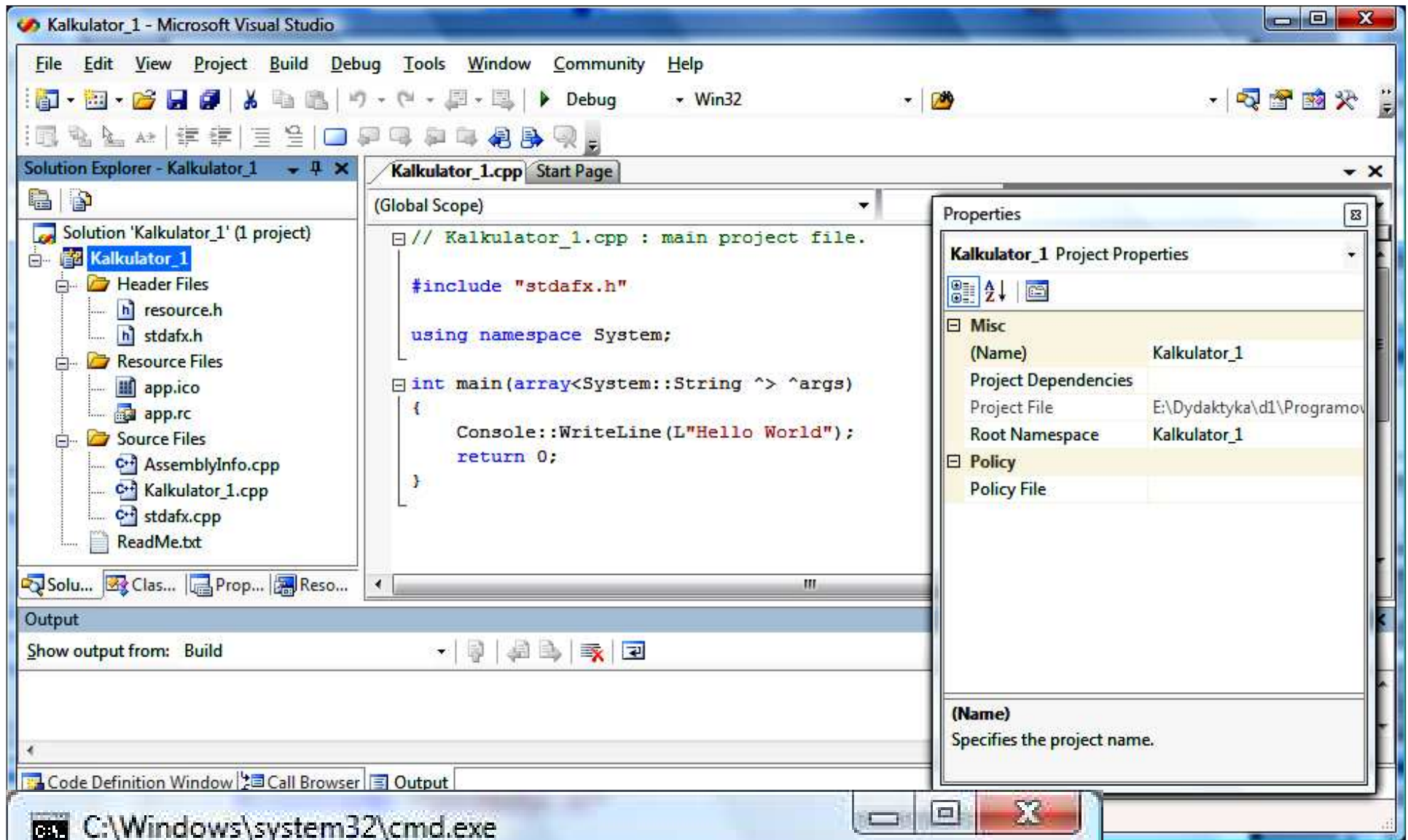
Name:

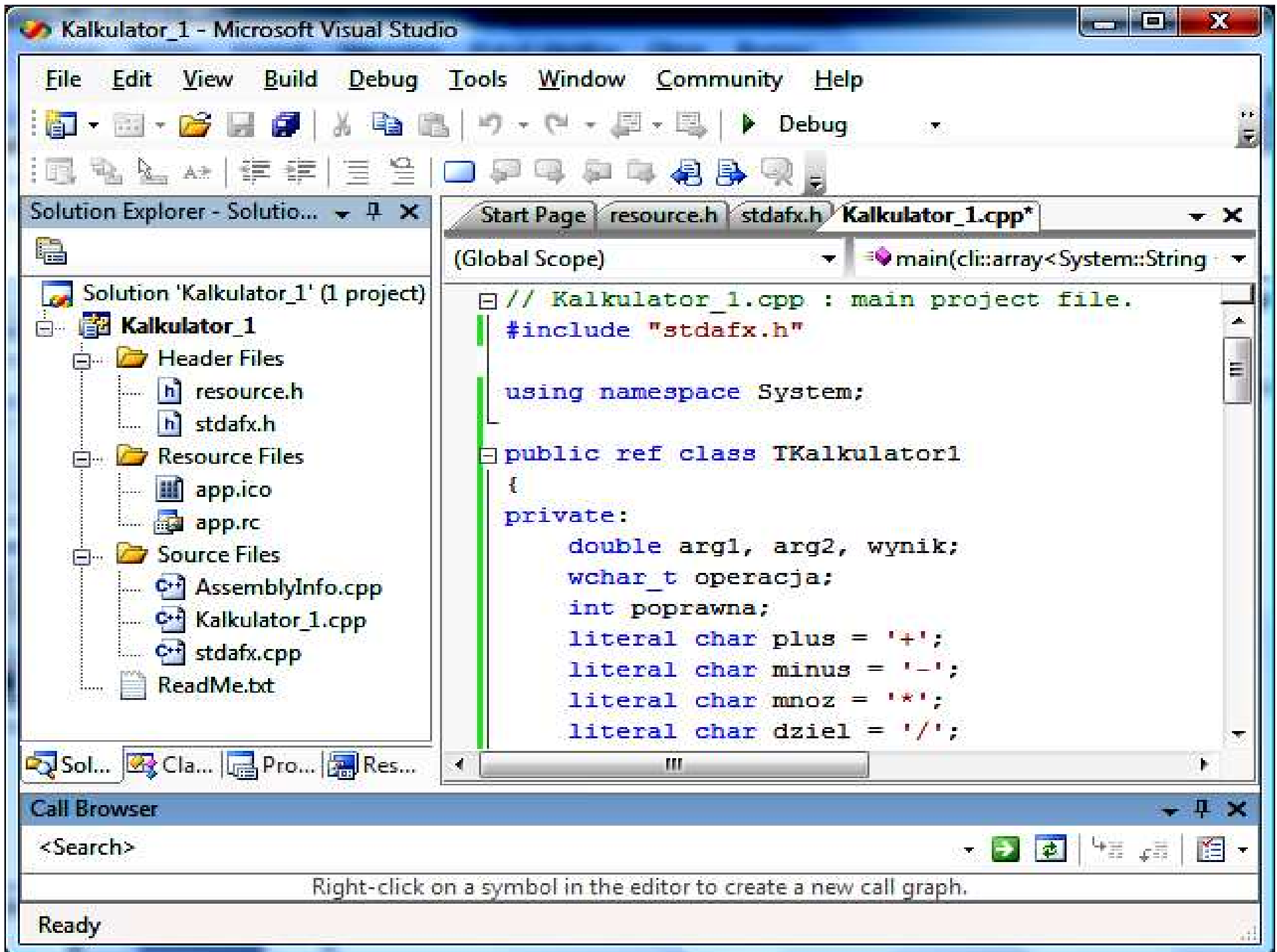
Location:

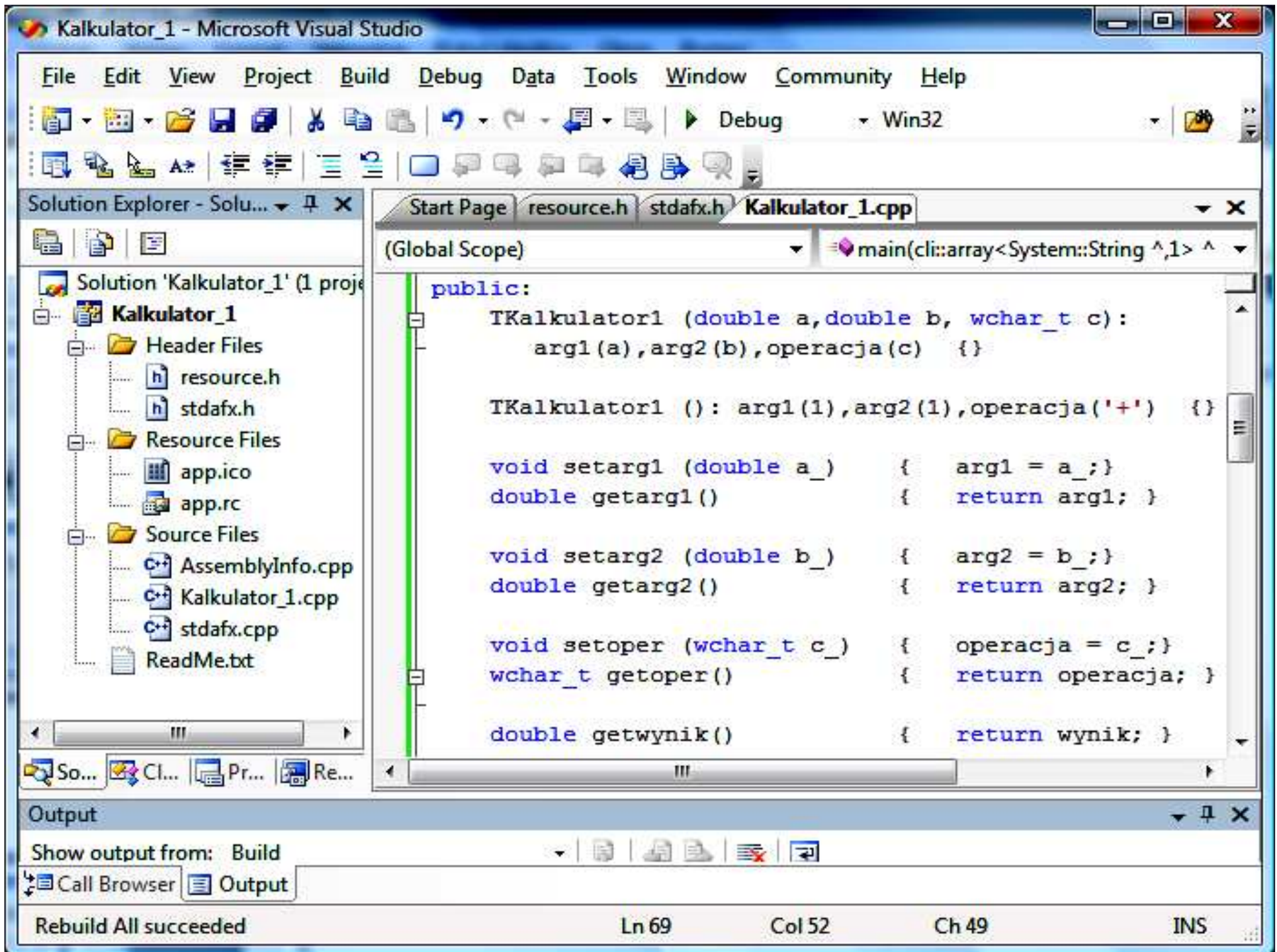
Solution:

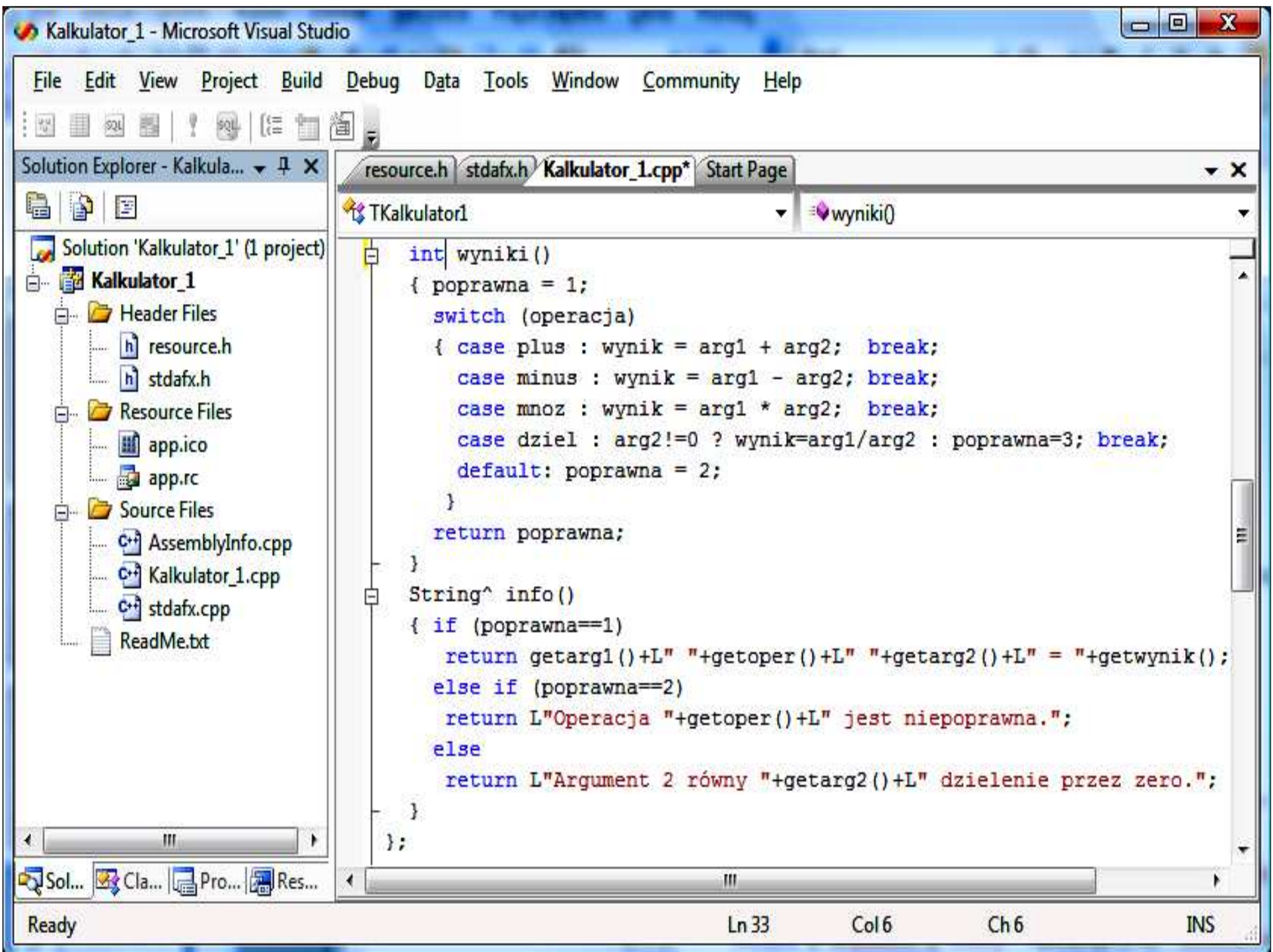
Create directory for solution

Solution Name:









```
#include "stdafx.h,, // Kalkulator_1.cpp : main project file.
```

```
using namespace System;
```

```
public ref class TKalkulator1
```

```
{ private:
```

```
double arg1, arg2, wynik;
```

```
wchar_t operacja;
```

```
int poprawna;
```

```
literal char plus = '+';
```

```
literal char minus = '-';
```

```
literal char mnoz = '*';
```

```
literal char dziel = '/';
```

```
public:
```

```
TKalkulator1 (double a, double b, wchar_t c) : arg1(a), arg2(b), operacja(c) {}
```

```
TKalkulator1 () : arg1(1), arg2(1), operacja('+') {}
```

```
void setarg1 (double a_) { arg1 = a_;}
```

```
double getarg1() { return arg1; }
```

```
void setarg2 (double b_) { arg2 = b_;}
```

```
double getarg2() { return arg2; }
```

```
void setoper (wchar_t c_) { operacja = c_;}
```

```
wchar_t getoper() { return operacja; }
```

```
double getwynik() { return wynik; }
```

2-bajtowy typ znakowy, który w operacjach arytmetycznych nie zachowuje się jak int

Klasa typu **ref** – jej obiekty mogą być dostępne tylko za pomocą referencji

konstruktory przeciążone z listą argumentów


```

int wyniki()
{
    poprawna = 1;
    switch (operacja)
    {
        case plus : wynik = arg1 + arg2;   break;
        case minus : wynik = arg1 - arg2; break;
        case mnoz : wynik = arg1 * arg2;  break;
        case dziel : arg2!=0 ? wynik=arg1/arg2 : poprawna=3; break;
        default: poprawna = 2;
    }
    return poprawna;
}

```

Modyfikator L oznacza, że łańcuchy składają się z dwubajtowych znaków

```

String^ info()
{
    if (poprawna==1)
        return getarg1()+L" "+getoper()+L" "+getarg2()+L" = "+getwynik();
    else if (poprawna==2)
        return L"Operacja "+getoper()+L" jest niepoprawna.";
    else
        return L"Argument 2 równy "+getarg2()+L" dzielenie przez zero.";
}
};

```

```
int main(array<System::String ^> ^args)
{
    TKalkulator1^ kalkulator1 = gcnew TKalkulator1(3,0,'/');
    kalkulator1->wyniki();
    Console::WriteLine(kalkulator1->info());

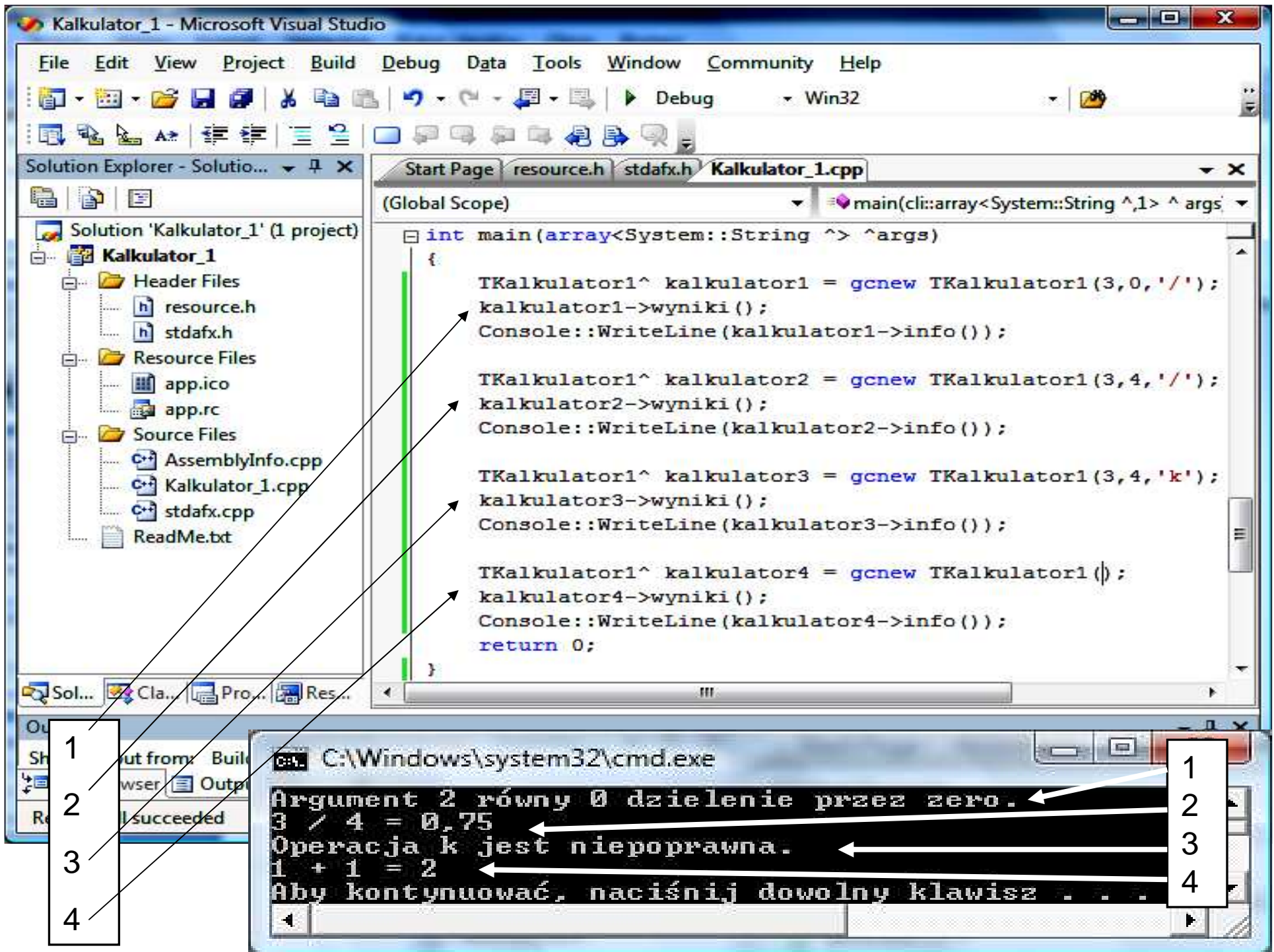
    TKalkulator1^ kalkulator2 = gcnew TKalkulator1(3,4,'/');
    kalkulator2->wyniki();
    Console::WriteLine(kalkulator2->info());

    TKalkulator1^ kalkulator3 = gcnew TKalkulator1(3,4,'k');
    kalkulator3->wyniki();
    Console::WriteLine(kalkulator3->info());

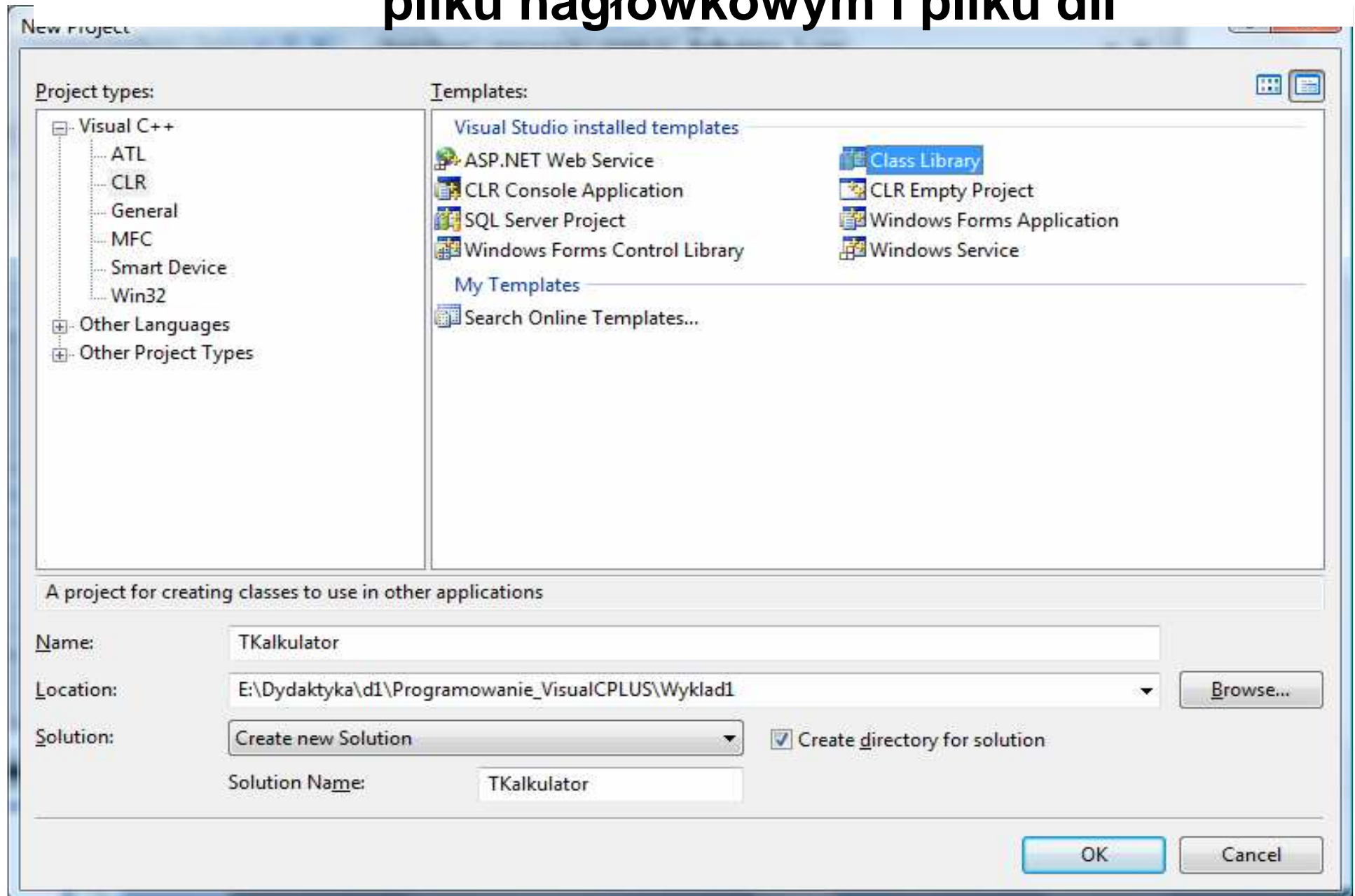
    TKalkulator1^ kalkulator4 = gcnew TKalkulator1();
    kalkulator4->wyniki();
    Console::WriteLine(kalkulator4->info());

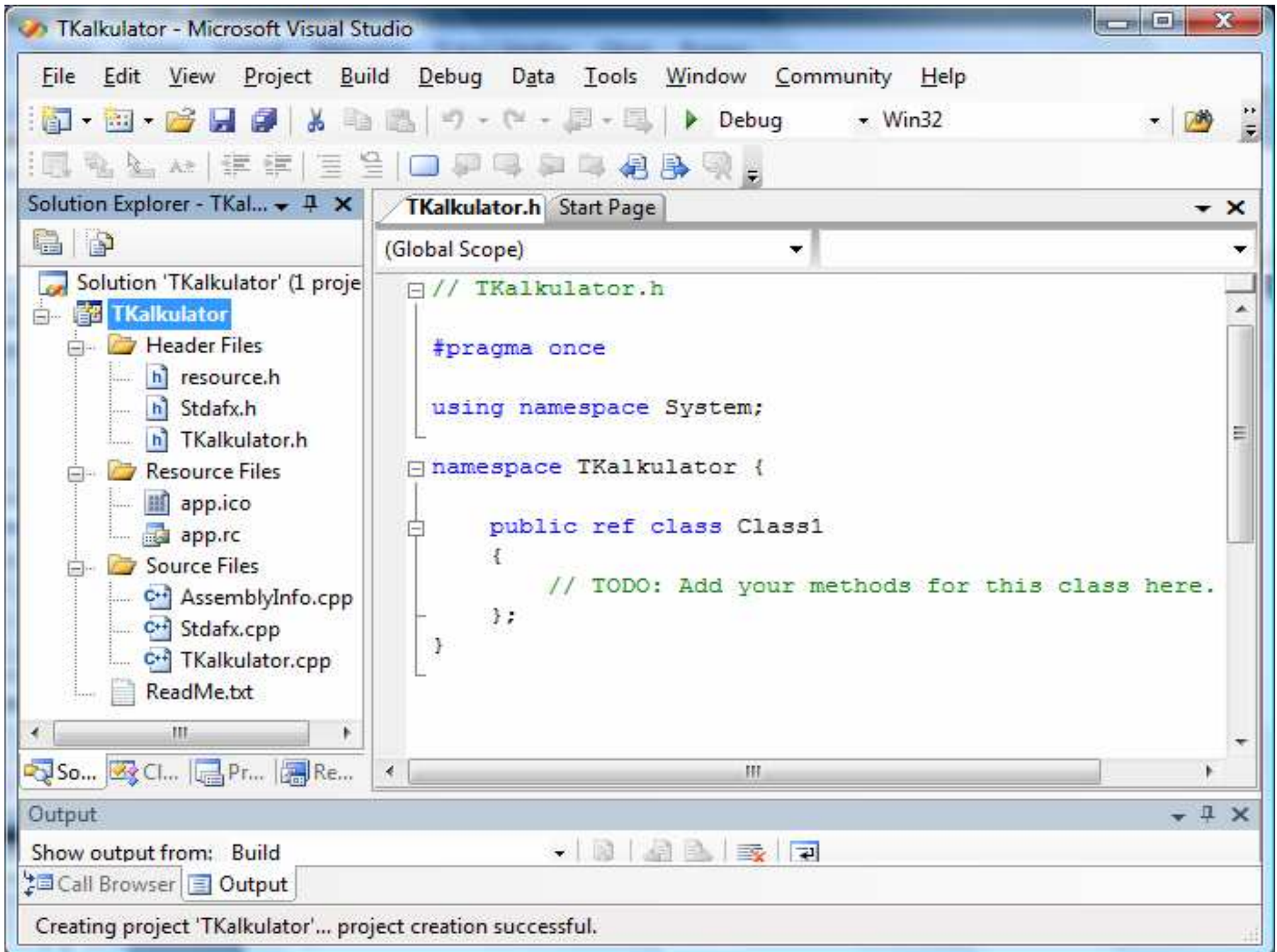
    return 0;
}
```

Klasa typu **ref**
– jej obiekty
mogą być
tylko dostępne
za pomocą
referencji.
Obiekt
wskazywany
przez
referencje
powstaje po
przydzieleniu
pamięci za
pomocą
operatora
gcnew

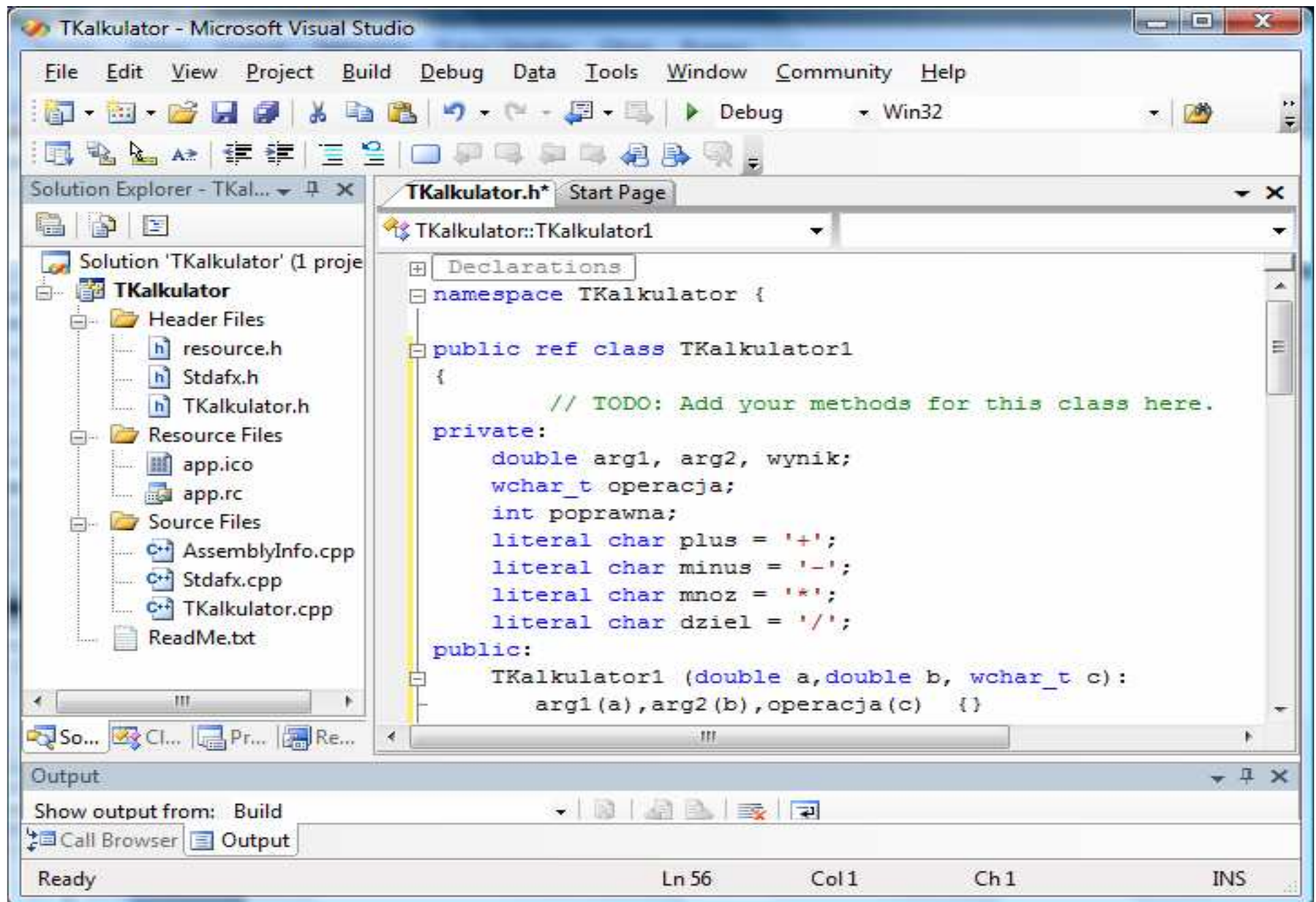


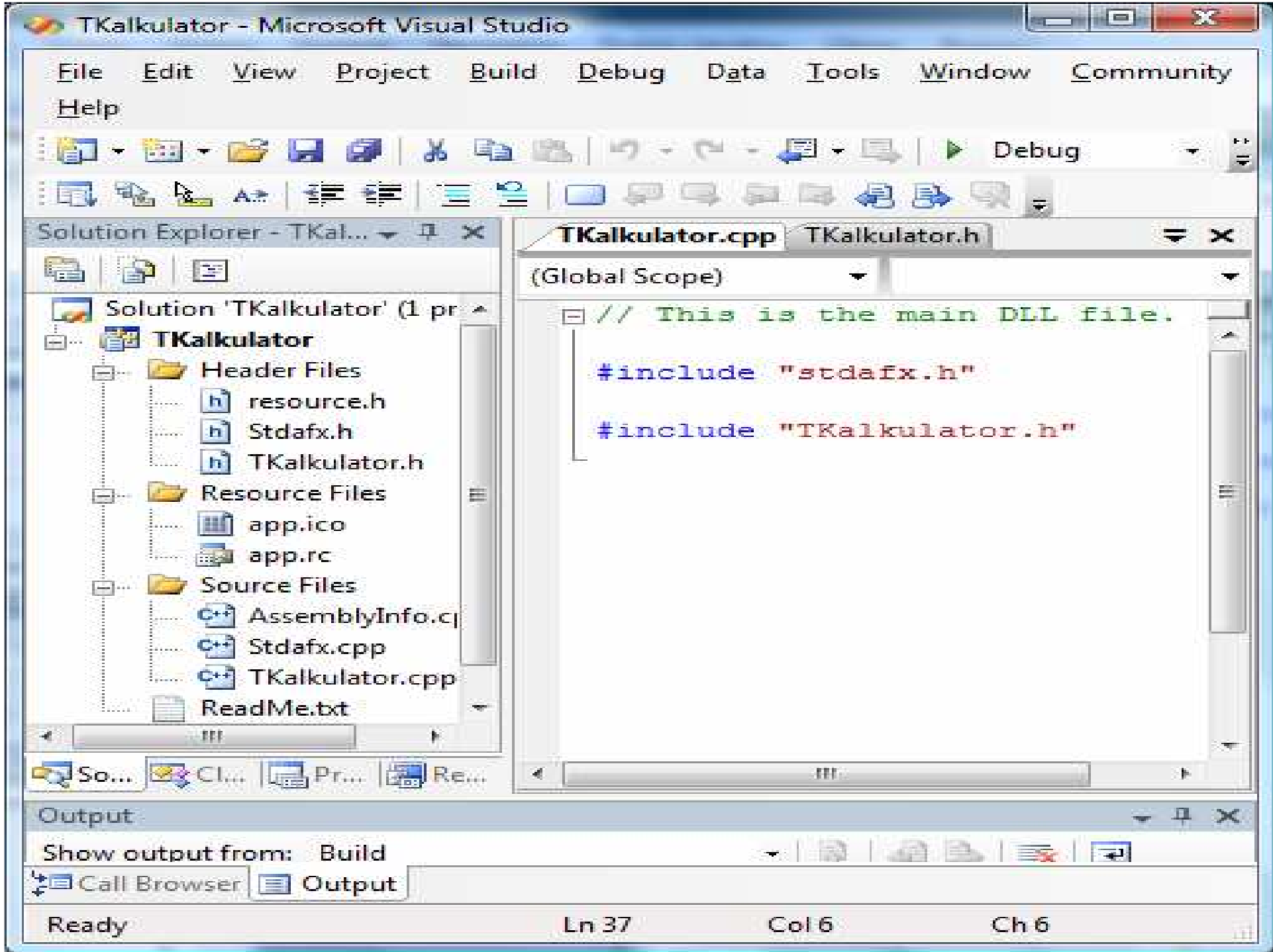
6. Tworzenie projektu do tworzenia definicji klasy w pliku nagłówkowym i pliku dll



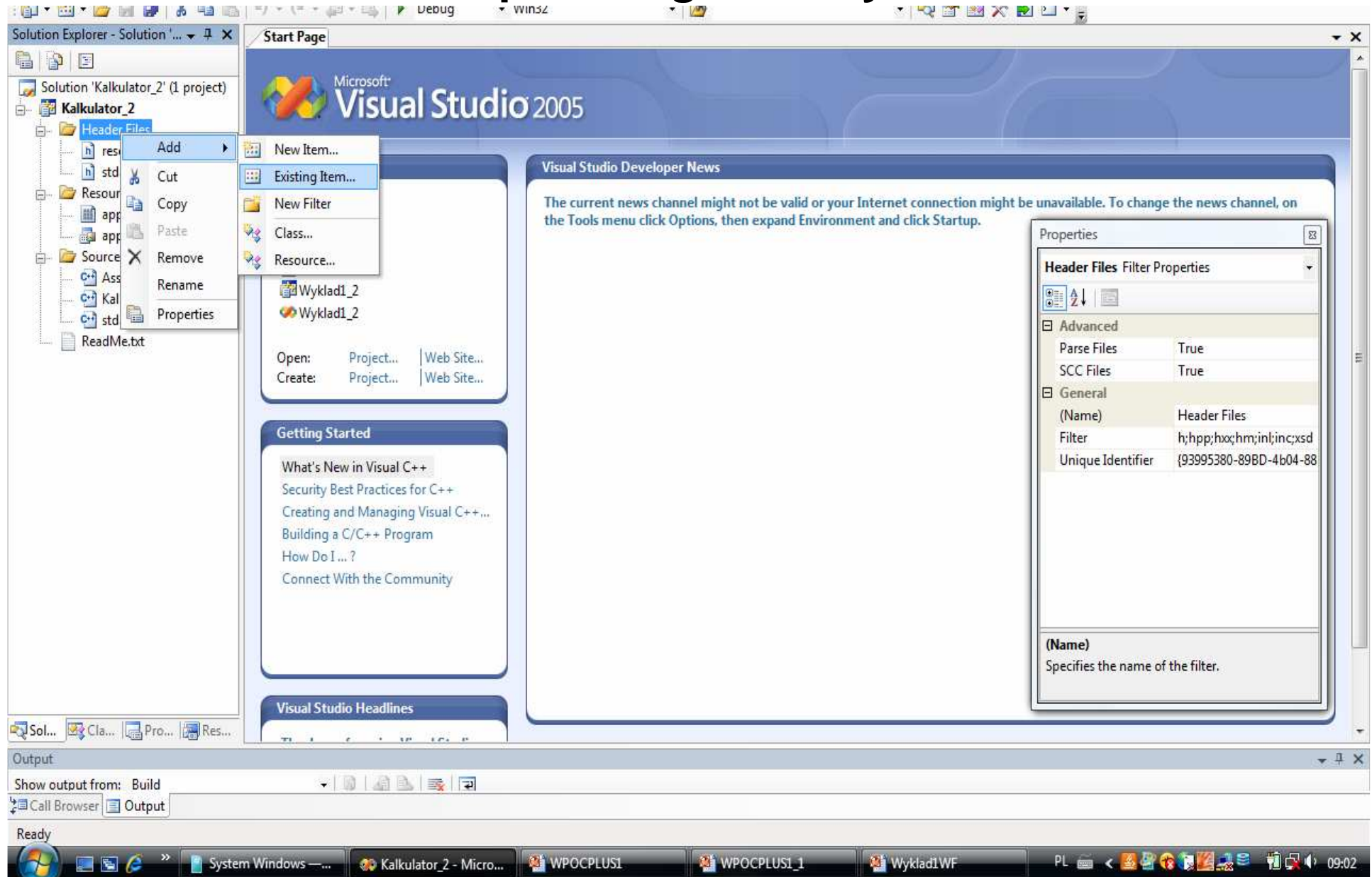


Utworzenie modułu

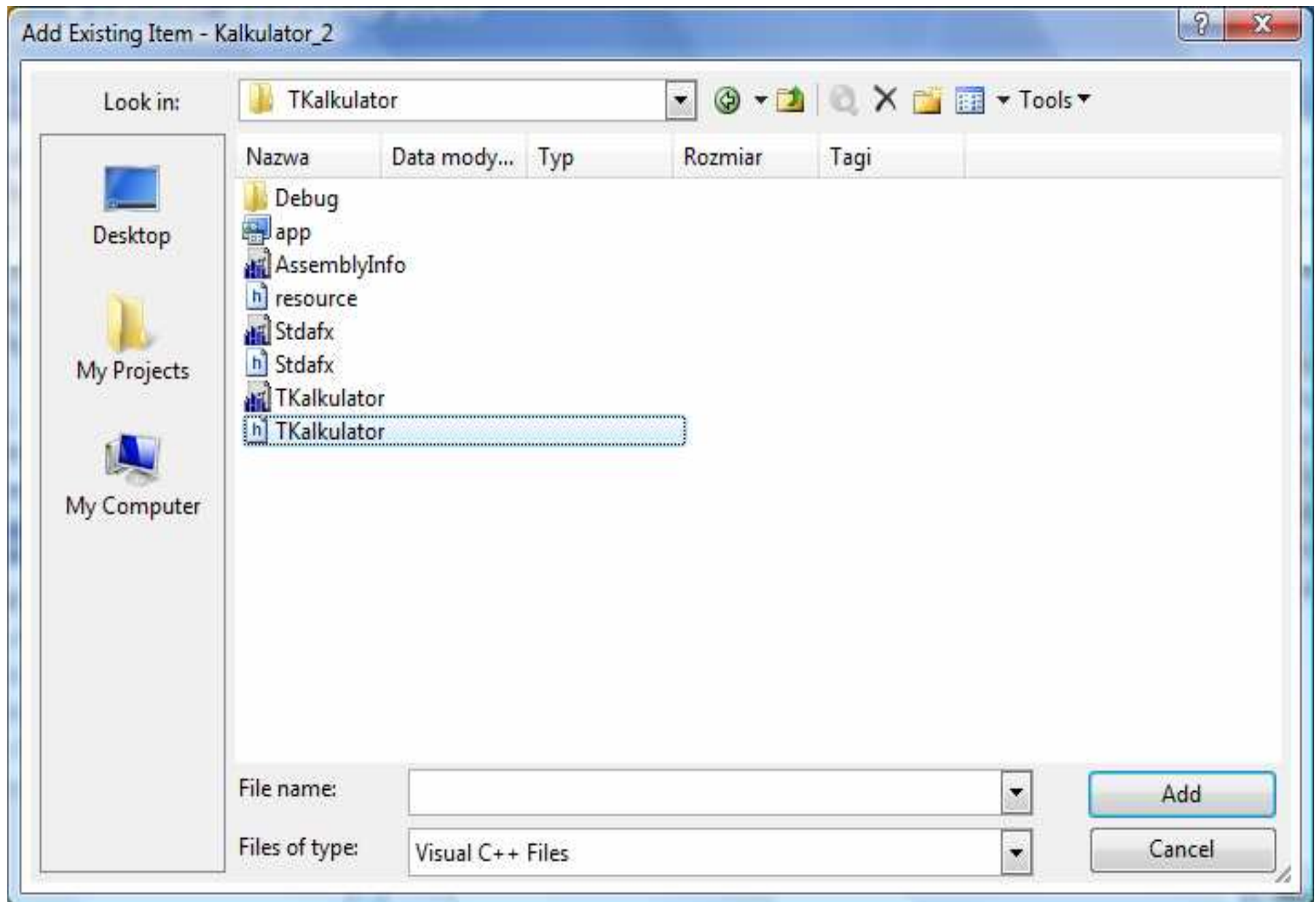




7. Tworzenie programu korzystającego z klasy zdefiniowanej w pliku nagłówkowym



Dodawanie istniejącego pliku nagłówkowego



Napisanie kodu programu testującego klasę zdefiniowaną w pliku nagłówkowym

Solution Explorer - Kalkulat... + ×

Solution 'Kalkulator_2' (1 project)

- Kalkulator_2
 - Header Files
 - resource.h
 - stdafx.h
 - TKalkulator.h
 - Resource Files
 - app.ico
 - app.rc
 - Source Files
 - AssemblyInfo.cpp
 - Kalkulator_2.cpp
 - stdafx.cpp
 - ReadMe.txt

Start Page **Kalkulator_2.cpp** + ×

(Global Scope)

```
// Kalkulator_2.cpp : main project file.  
  
#include "stdafx.h"  
#include "TKalkulator.h"  
using namespace System;  
using namespace TKalkulator;  
  
int main(array<System::String ^> ^args)  
{  
    TKalkulator1^ kalkulator1 = gcnew TKalkulator1(3,0,'/');  
    kalkulator1->wyniki();  
    Console::WriteLine(kalkulator1->info());  
  
    TKalkulator1^ kalkulator2 = gcnew TKalkulator1(3,4,'/');  
    kalkulator2->wyniki();  
    Console::WriteLine(kalkulator2->info());  
  
    TKalkulator1^ kalkulator3 = gcnew TKalkulator1(3,4,'k');  
    kalkulator3->wyniki();  
    Console::WriteLine(kalkulator3->info());  
  
    TKalkulator1^ kalkulator4 = gcnew TKalkulator1();  
    kalkulator4->wyniki();  
    Console::WriteLine(kalkulator4->info());  
    return 0;  
}
```

C:\Windows\system32\cmd.exe

```
Argument 2 równy 0 dzielenie przez zero.  
3 / 4 = 0,75  
Operacja k jest niepoprawna.  
1 + 1 = 2  
Aby kontynuować, naciśnij dowolny klawisz . . .
```

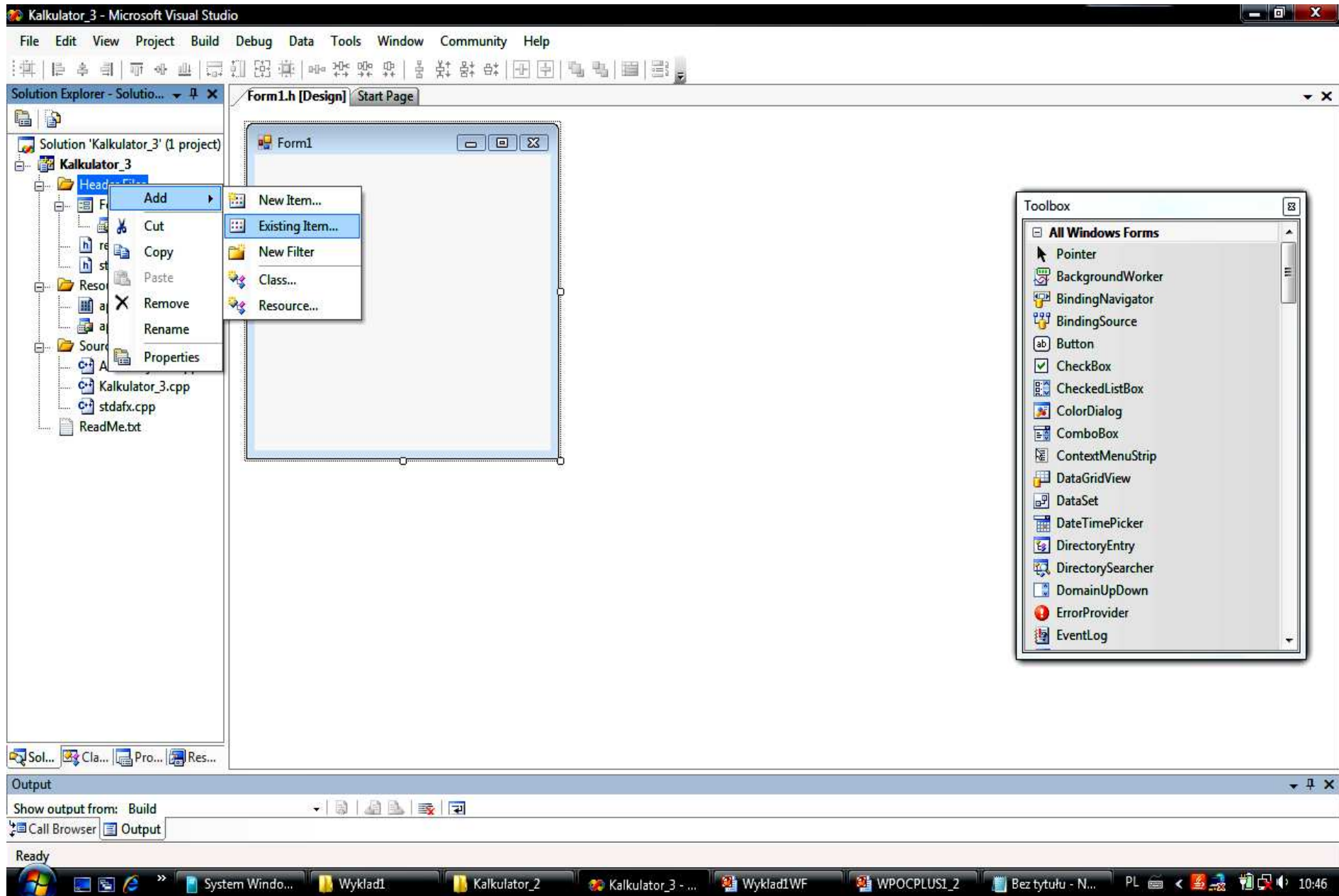
Output

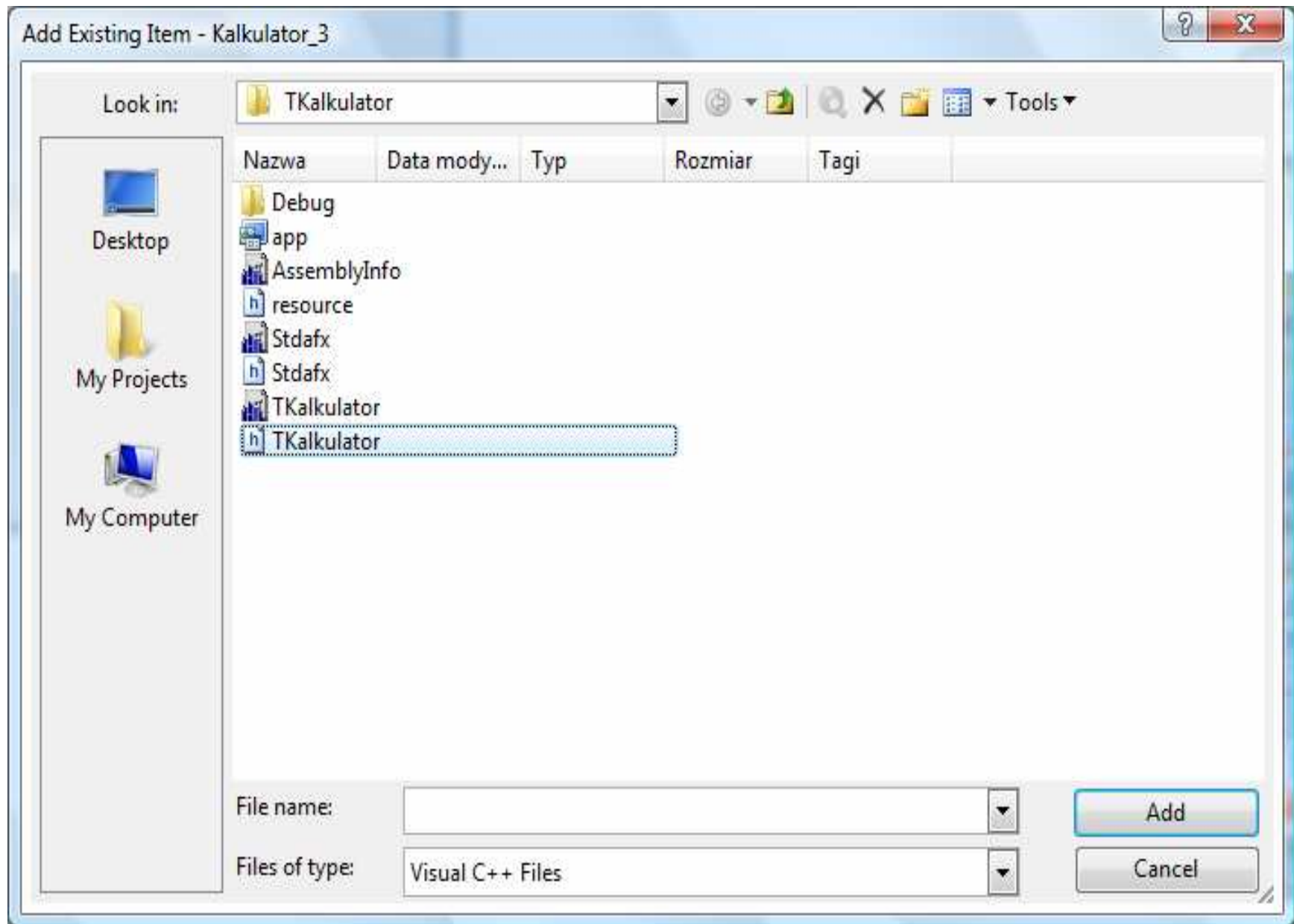
Show output from: Build

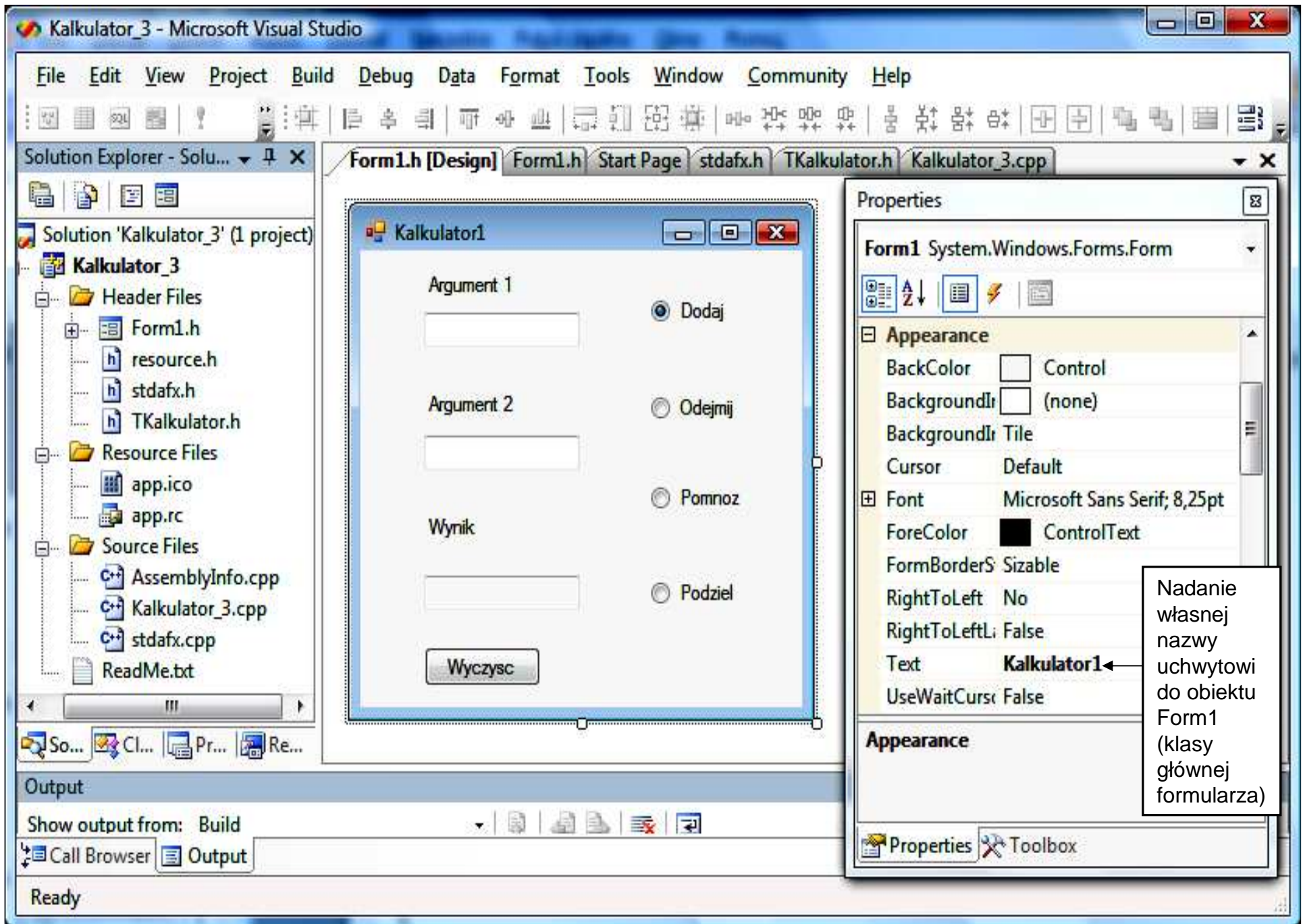
Call Browser Output

Ready

8. Interfejs graficzny aplikacji - obsługa zdarzeń







The image displays three instances of the Visual Studio Properties window, each showing the configuration for a different control:

- dodajButton (System.Windows.Forms.RadioButton):**
 - Appearance: Normal
 - BackColor: Control
 - BackgroundImage: (none)
 - BackgroundImageLayout: Tile
 - CheckAlign: MiddleLeft
 - Checked: **True**
 - Cursor: Default
 - FlatAppearance: FlatStyle: Standard
 - Font: Microsoft Sans Serif; 8
 - ForeColor: ControlText
 - Image: (none)
 - ImageAlign: MiddleCenter
 - ImageIndex: (none)
 - ImageKey: (none)
 - ImageList: (none)
 - RightToLeft: No
 - Text: **Dodaj**
 - TextAlign: MiddleLeft
 - TextImageRelation: Overlay
 - UseMnemonic: True
 - UseVisualStyleBackColor: **True**
 - UseWaitCursor: False
 - Behavior: (empty)
 - Data: (empty)
 - Design: (Name) **dodajButton**
- podzielButton (System.Windows.Forms.RadioButton):**
 - Appearance: Normal
 - BackColor: Control
 - BackgroundImage: (none)
 - BackgroundImageLayout: Tile
 - CheckAlign: MiddleLeft
 - Checked: **False**
 - Cursor: Default
 - FlatAppearance: FlatStyle: Standard
 - Font: Microsoft Sans Serif; 8
 - ForeColor: ControlText
 - Image: (none)
 - ImageAlign: MiddleCenter
 - ImageIndex: (none)
 - ImageKey: (none)
 - ImageList: (none)
 - RightToLeft: No
 - Text: **Podziel**
 - TextAlign: MiddleLeft
 - TextImageRelation: Overlay
 - UseMnemonic: True
 - UseVisualStyleBackColor: **True**
 - UseWaitCursor: False
 - Behavior: (empty)
 - Data: (empty)
 - Design: (Name) **podzielButton**
- usun (System.Windows.Forms.Button):**
 - Appearance: BackColor: Control; BackgroundImage: (none); BackgroundImageLayout: Tile; Cursor: Default
 - FlatAppearance: FlatStyle: Standard
 - Font: Microsoft Sans Serif; 8,2
 - ForeColor: ControlText
 - Image: (none)
 - ImageAlign: MiddleCenter
 - ImageIndex: (none)
 - ImageKey: (none)
 - ImageList: (none)
 - RightToLeft: No
 - Text: **Wyczysc**
 - TextAlign: MiddleCenter
 - TextImageRelation: Overlay
 - UseMnemonic: True
 - UseVisualStyleBackColor: **True**
 - UseWaitCursor: False
 - Behavior: (empty)
 - Data: (empty)
 - Design: (Name) **usun**; GenerateMember: True; Locked: False; Modifiers: Private

A callout box points to the 'usun' control's Name property with the text: "Nadanie własnej nazwy uchwytowi do obiektów kontrolki formularza".

Zakładka definicji właściwości w oknie Properties kontrolki typu RadioButton i Button

The image displays three instances of the Visual Studio Properties window for a `System.Windows.Forms.TextBox` control, each with a different name: `arg1`, `arg2`, and `wynik`. The `Behavior` tab is active in all three windows, showing a list of properties and their values. The `Design` tab is also visible at the bottom of each window, showing the control's name in parentheses.

Property	arg1	arg2	wynik
AcceptsReturn	False	False	False
AcceptsTab	False	False	False
AllowDrop	False	False	False
CharacterCasing	Normal	Normal	Normal
ContextMenuStrip	(none)	(none)	(none)
Enabled	True	True	True
HideSelection	True	True	True
ImeMode	NoControl	NoControl	NoControl
MaxLength	32767	32767	32767
Multiline	False	False	False
PasswordChar			
ReadOnly	False	False	True
ShortcutsEnabled	True	True	True
TabIndex	4	5	8
TabStop	True	True	True
UseSystemPasswordChar	False	False	False
Visible	True	True	True
WordWrap	True	True	True

Nadanie własnej nazwy uchwytowi do obiektu Form1 (klasy głównej formularza)

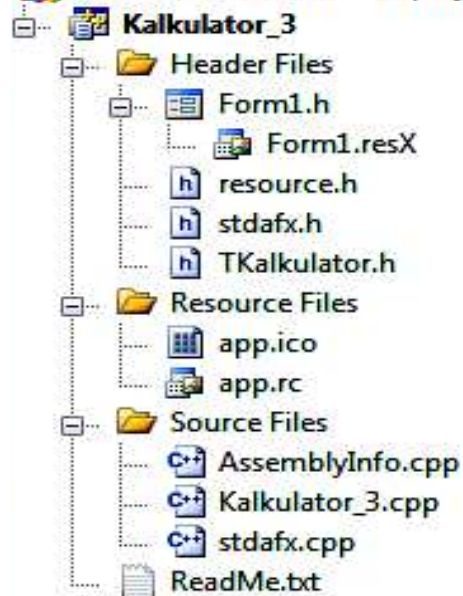
Zakładka definicji właściwości w oknie Properties kontrolki typu TextBox



Solution Explorer - Kalk... X



Solution 'Kalkulator_3' (1 proje...



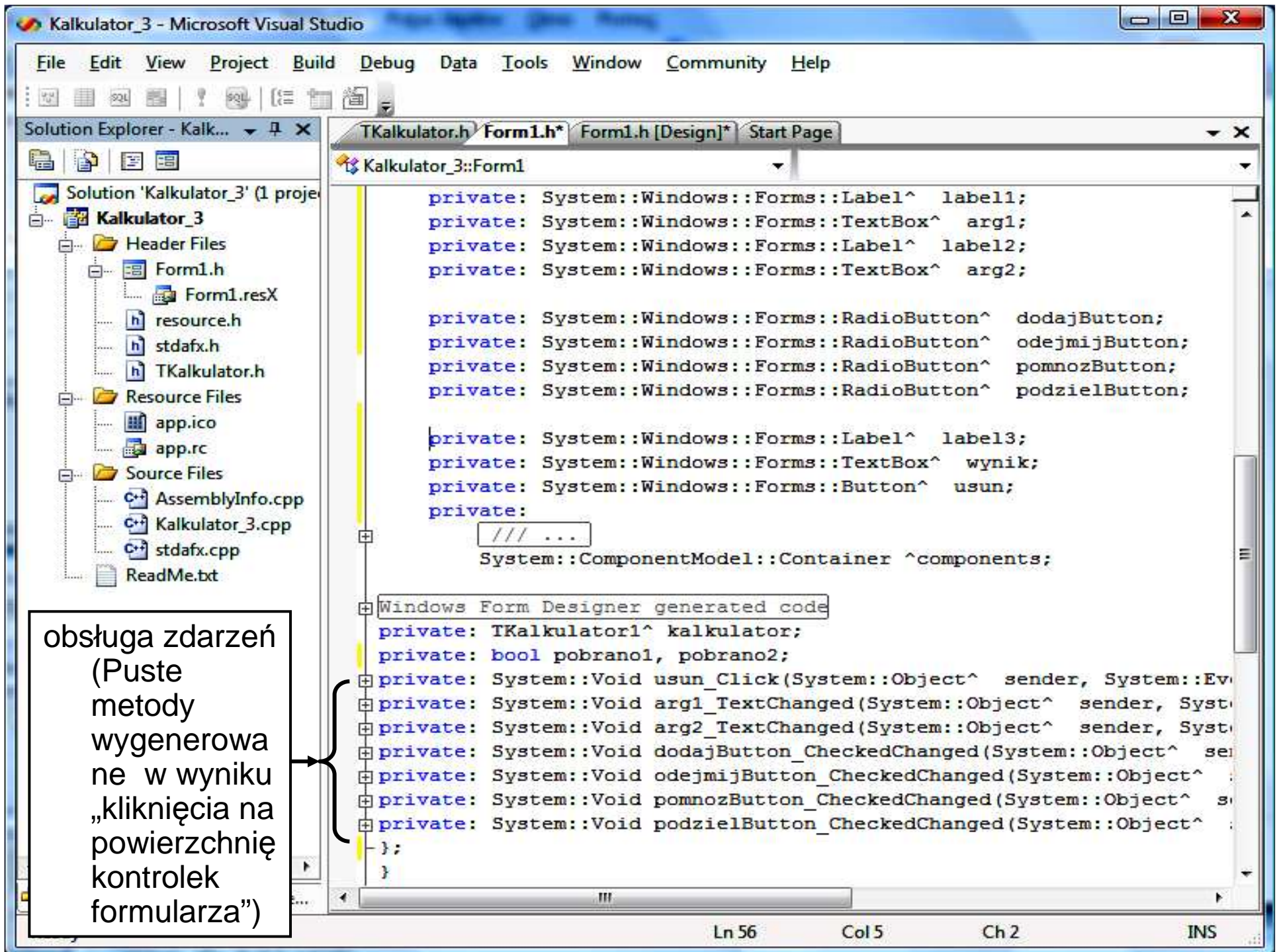
TKalkulator.h Form1.h* Form1.h [Design]* Start Page

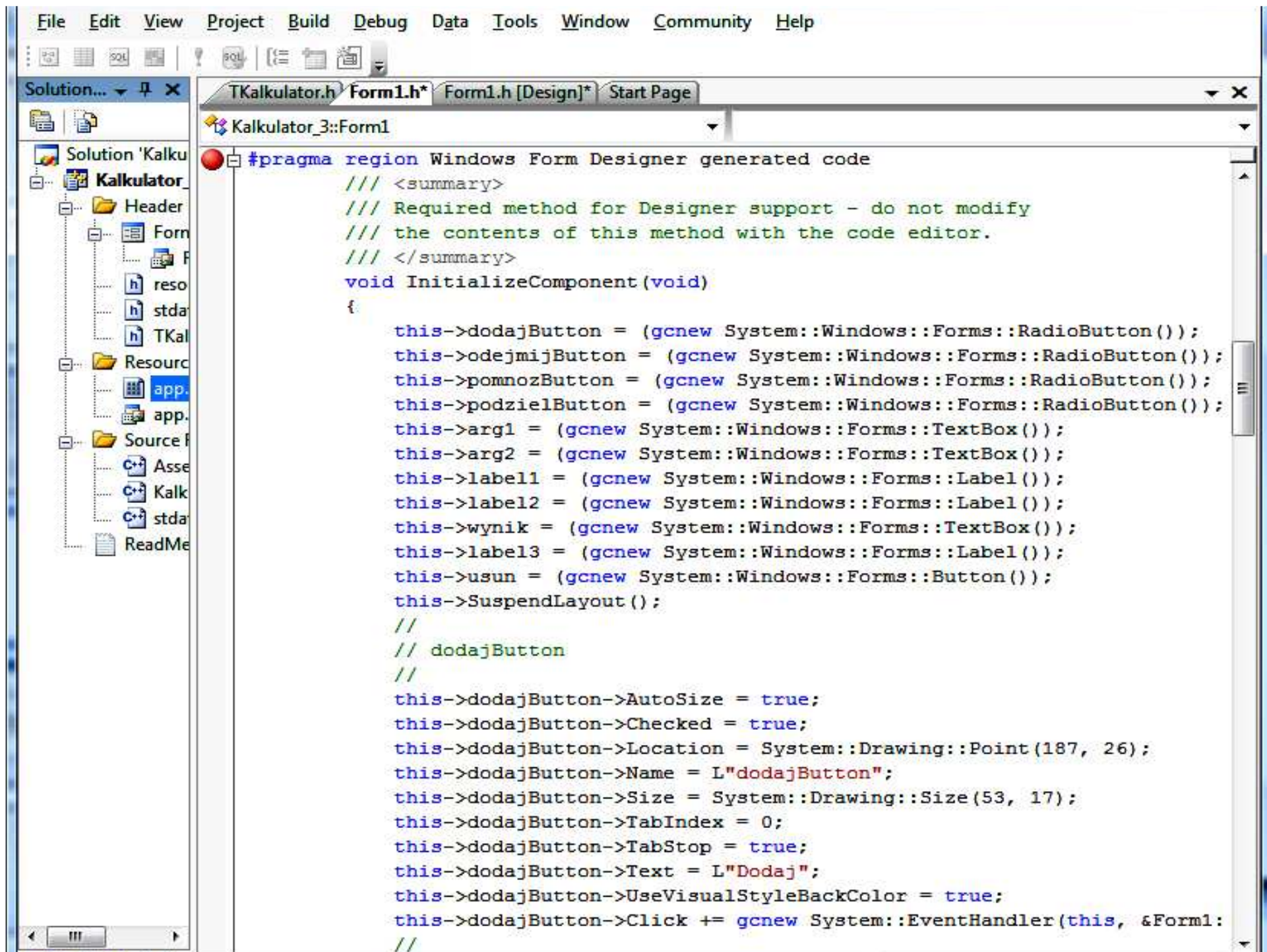
Kalkulator_3::Form1

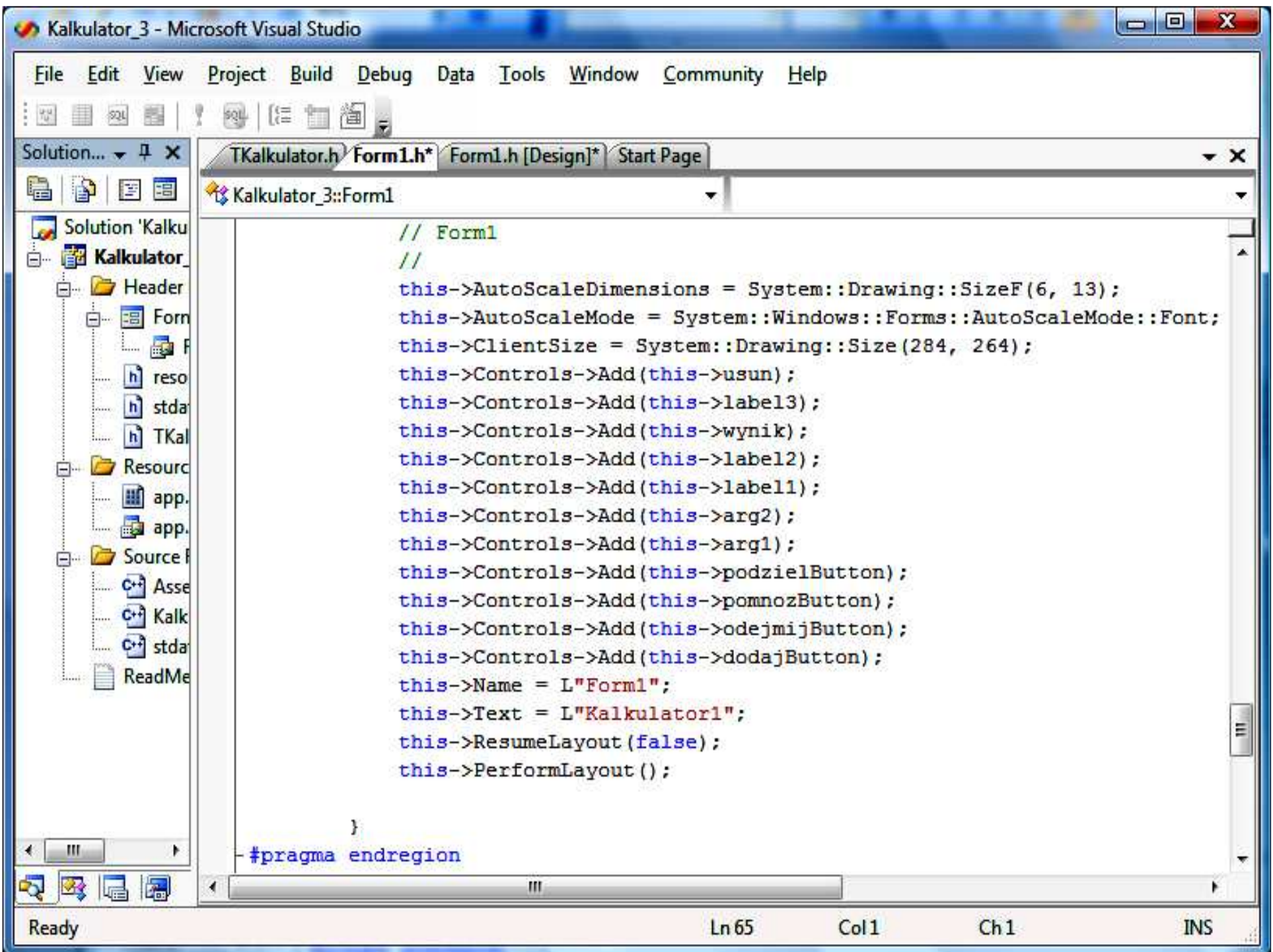
```
namespace Kalkulator_3 {
    #include "TKalkulator.h"
    using namespace System;
    using namespace System::ComponentModel;
    using namespace System::Collections;
    using namespace System::Windows::Forms;
    using namespace System::Data;
    using namespace System::Drawing;
    using namespace TKalkulator;
    /// ...
    public ref class Form1 : public System::Windows::Forms::Form
    {
    public:
        Form1(void)
        {
            kalkulator = gcnew TKalkulator1;
            pobrano1 = pobrano2 = false;
            InitializeComponent();
            /* ... */
        }

    protected:
        /// ...
        ~Form1()
        {
            if (components)
            {
                delete components;
            }
        }
    }
}
```

So... Cl... Pr... Re...

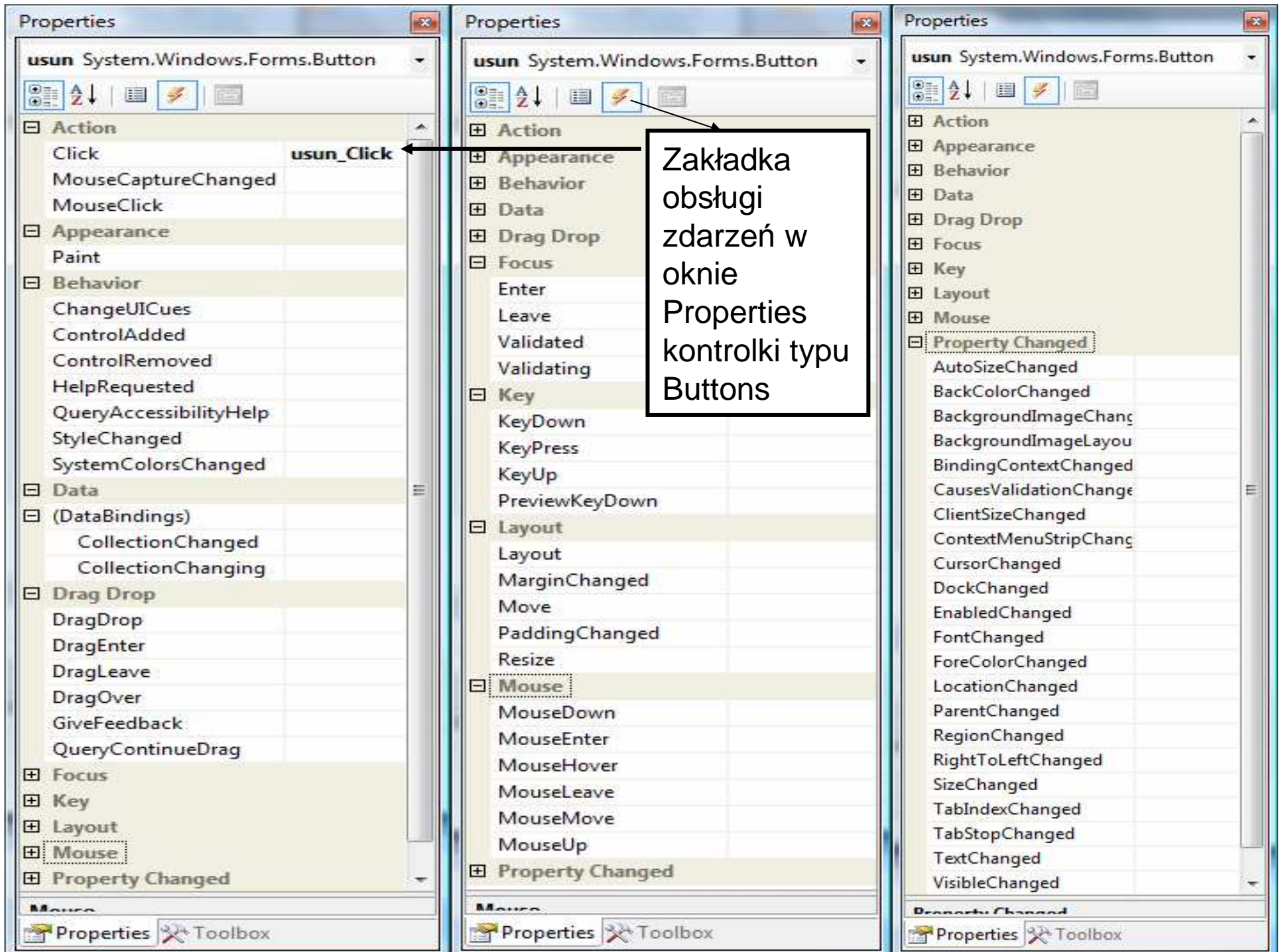


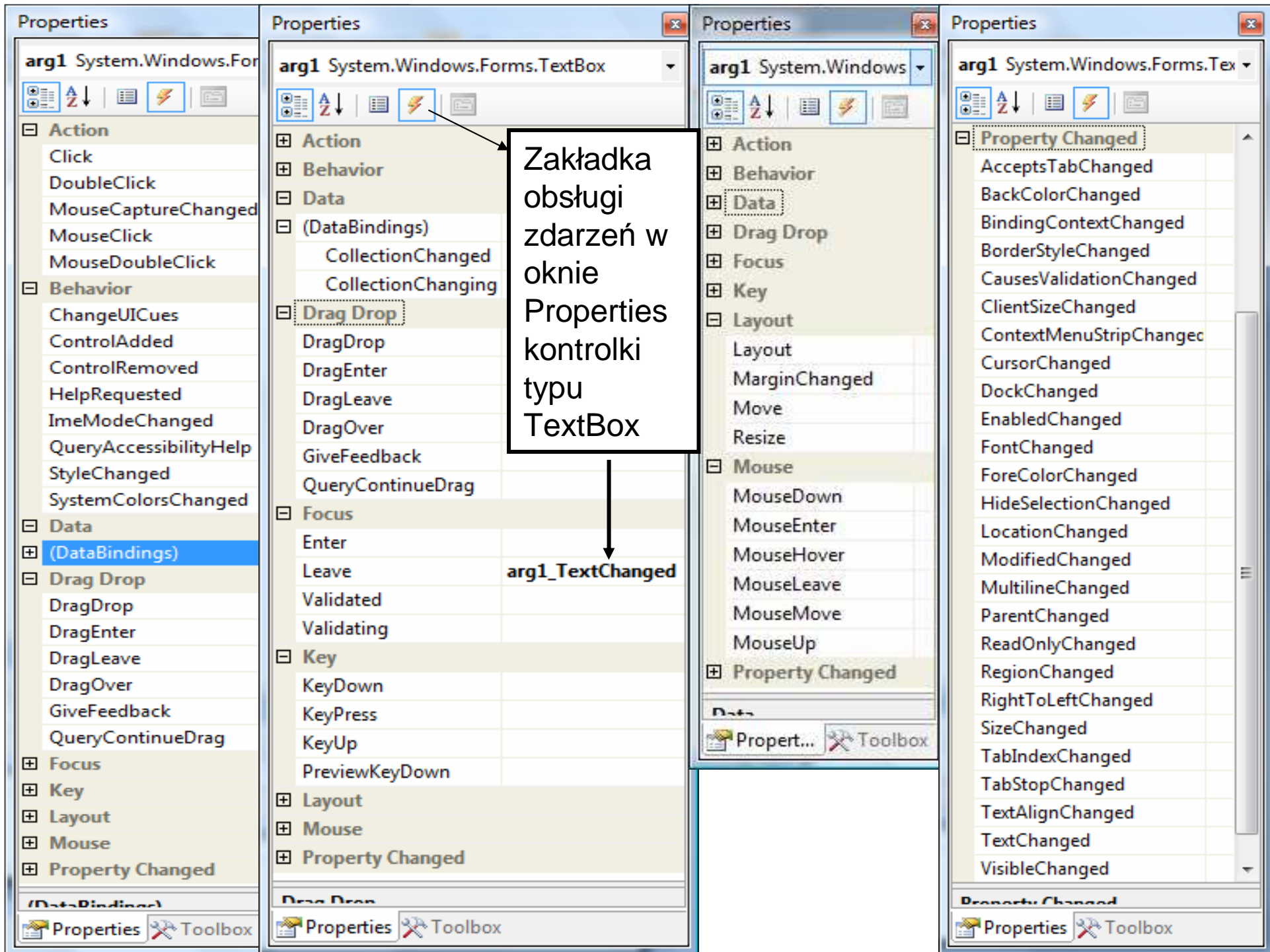


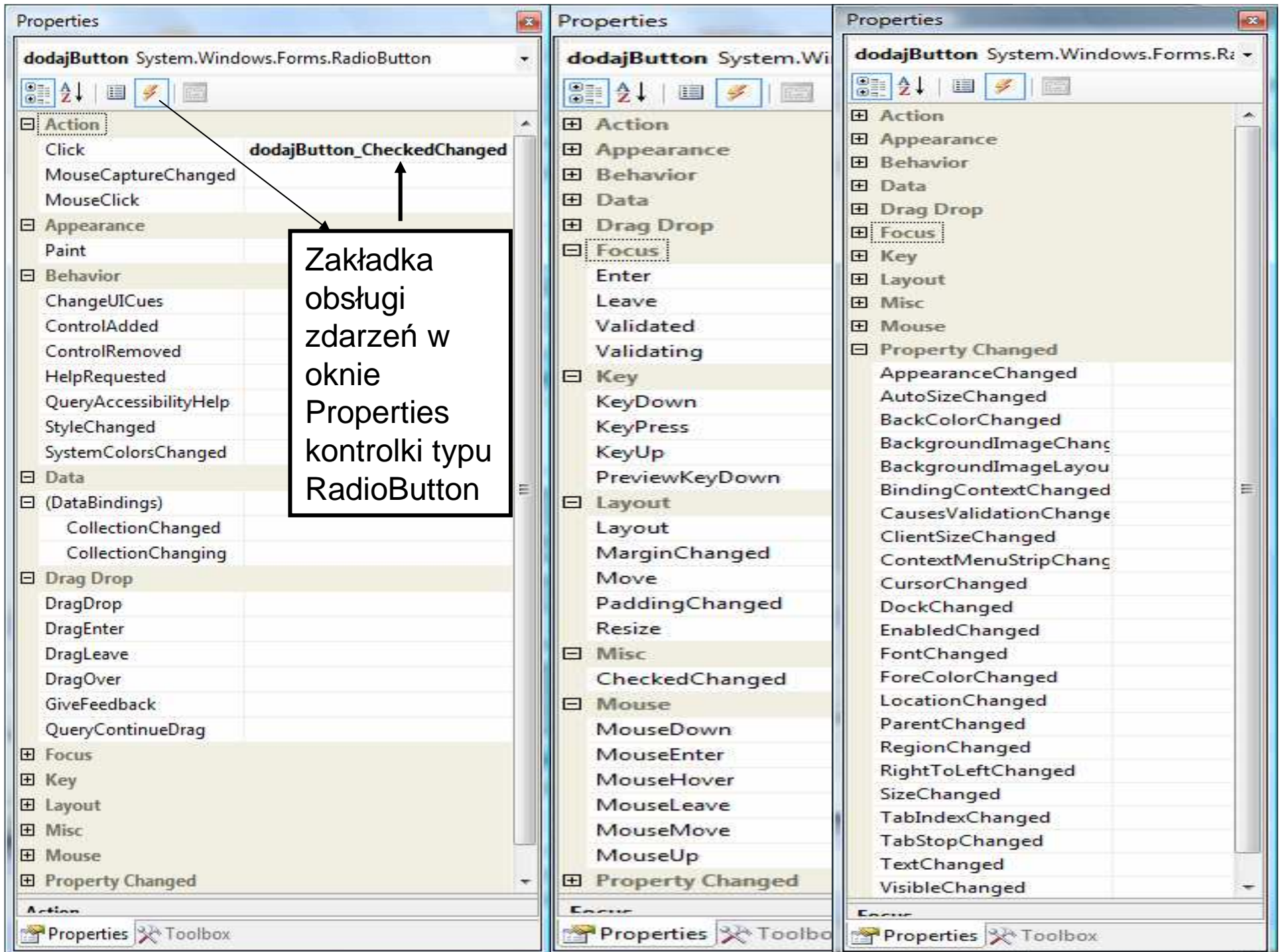




```
private: TKalkulator1^ kalkulator;
private: bool pobrano1, pobrano2;
private: System::Void usun_Click(System::Object^ sender, System::EventArgs^ e) { ... }
private: System::Void arg1_TextChanged(System::Object^ sender, System::EventArgs^ e) { ... }
private: System::Void arg2_TextChanged(System::Object^ sender, System::EventArgs^ e) { ... }
private: System::Void dodajButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) { .
private: System::Void odejmijButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) { {
private: System::Void pomnozButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) { {
private: System::Void podzielButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) { {
- };
}
```







```
Form1.h [Design] Form1.h Start Page stdafx.h TKalkulator.h Kalkulator_3.cpp
Kalkulator_3::Form1 usun_Click(System::Object ^ sender, System::EventArgs ^ e)
private: TKalkulator1^ kalkulator;
private: bool pobrano1, pobrano2;
private: System::Void usun_Click(System::Object^ sender, System::EventArgs^ e) {
    pobrano2=pobrano1=false;
    arg1->Text="";
    arg2->Text="";
    wynik->Text="";
}
private: System::Void arg1_TextChanged(System::Object^ sender, System::EventArgs^ e) {
    double warg1 = Double::Parse(arg1->Text);
    kalkulator->setarg1(warg1);
    pobrano1=true;
}
private: System::Void arg2_TextChanged(System::Object^ sender, System::EventArgs^ e) {
    double warg2 = Double::Parse(arg2->Text);
    kalkulator->setarg2(warg2);
    pobrano2=true;
}
ser
```


Solution 'Kalkul...' ▾

- Kalkulator_3
 - Header Files
 - Form1.h
 - Form1.h [Design]
 - TKalkulator.h
 - Resource Files
 - app.i
 - app.r
 - Source Files
 - Asser
 - Kalku
 - stdaf

```

Kalkulator_3::Form1
private: TKalkulator1^ kalkulator;
private: bool pobrano1, pobrano2;
private: System::Void usun_Click(System::Object^ sender, System::EventArgs^ e) { ... }
private: System::Void arg1_TextChanged(System::Object^ sender, System::EventArgs^ e) { ... }
private: System::Void arg2_TextChanged(System::Object^ sender, System::EventArgs^ e) { ... }
private: System::Void dodajButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
    if(pobrano1 && pobrano2)
    { kalkulator->setoper('+');
      kalkulator->wyniki();
      wynik->Text=kalkulator->info();
    }
}
private: System::Void odejmijButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
}
private: System::Void pomnozButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
}
private: System::Void podzielButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
}
};
}
    
```



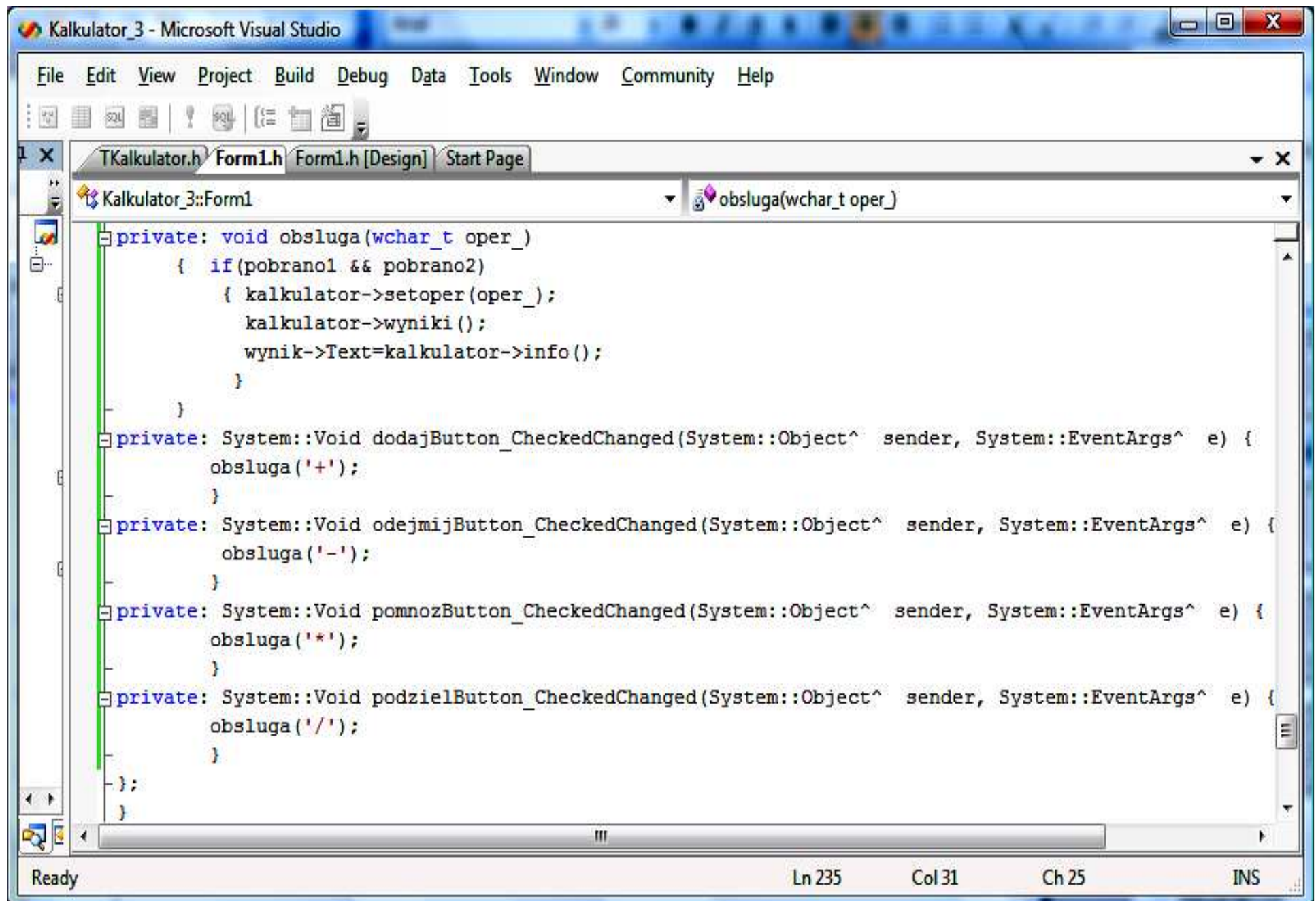
```
private: System::Void dodajButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
    if(pobranol && pobrano2)
    { kalkulator->setoper('+');
      kalkulator->wyniki();
      wynik->Text=kalkulator->info();
    }
}

private: System::Void odejmijButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
    if(pobranol && pobrano2)
    { kalkulator->setoper('-');
      kalkulator->wyniki();
      wynik->Text=kalkulator->info();
    }
}

private: System::Void pomnozButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
    if(pobranol && pobrano2)
    { kalkulator->setoper('*');
      kalkulator->wyniki();
      wynik->Text=kalkulator->info();
    }
}

private: System::Void podzielButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
    if(pobranol && pobrano2)
    { kalkulator->setoper('/');
      kalkulator->wyniki();
      wynik->Text=kalkulator->info();
    }
}
```

Poprawa kodu



```
Kalkulator_3 - Microsoft Visual Studio
File Edit View Project Build Debug Data Tools Window Community Help
TKalkulator.h Form1.h Form1.h [Design] Start Page
Kalkulator_3::Form1
obsługa(wchar_t oper_)
private: void obsługa(wchar_t oper_)
{
    if(pobrano1 && pobrano2)
    {
        kalkulator->setoper(oper_);
        kalkulator->wyniki();
        wynik->Text=kalkulator->info();
    }
}
private: System::Void dodajButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
    obsługa('+');
}
private: System::Void odejmijButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
    obsługa('-');
}
private: System::Void pomnozButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
    obsługa('*');
}
private: System::Void podzielButton_CheckedChanged(System::Object^ sender, System::EventArgs^ e) {
    obsługa('/');
}
};
}
```

Ready Ln 235 Col 31 Ch 25 INS

Kalkulator1

Argument 1

Argument 2

Wynik

Dodaj
 Odejmij
 Pomnoz
 Podziel

Wyczysc

Kalkulator1

Argument 1
2

Argument 2
3

Wynik
2 + 3 = 5

Dodaj
 Odejmij
 Pomnoz
 Podziel

Wyczysc

Kalkulator1

Argument 1
2

Argument 2
0

Wynik
Argument 2 równy 0

Dodaj
 Odejmij
 Pomnoz
 Podziel

Wyczysc

Kalkulator1

Argument 1
2

Argument 2
3

Wynik
2 / 3 = 0,666666666

Dodaj
 Odejmij
 Pomnoz
 Podziel

Wyczysc