

Wprowadzenie do programowania obiektowego w C++

1. Główne zasady programowania obiektowego: hermetyzacja, dziedziczenie, polimorfizm
2. Pojęcie klasy: sposoby deklarowania i definiowania składowych klasy,
3. Atrybuty dostępu do składowych - private, protected i public wspierające hermetyzację
4. Konstruktory i destruktory
5. Obiekty automatyczne, statyczne i dynamiczne
6. Składowe klasy typu static
7. Zmienne referencyjne, przekazywanie parametrów przez referencję, autoreferencja
8. **Pliki nagłówkowe, dyrektywy preprocesora, programy wieloplikowe**

#include <iostream.h> - dołączenie pliku nagłówkowego z katalogu standardowego
#include „TProdukt1.h” - dołączanie pliku nagłówkowego z katalogu bieżącego

Pliki nagłówkowe powinny zawierać:

- definicje typów np.

```
class punkt  
    { float x, y;  
    public:  
        punkt (float, float);  
        void przesun (float, float);  
    };
```

- szablony funkcji i klas

```
template <class T>  
    class stos { // .....}  
    void wyswietl_float(float);  
    inline int Większy(int x, int y) {return x > y; }  
    extern int zmienna;  
    const int max = 3;  
    enum Boolean {False, True};  
    class kolo;  
    #include <iostream.h>  
    #define MAX(x, y) ((x) > (y) ? (x) : (y))  
    #define TRUE 1
```

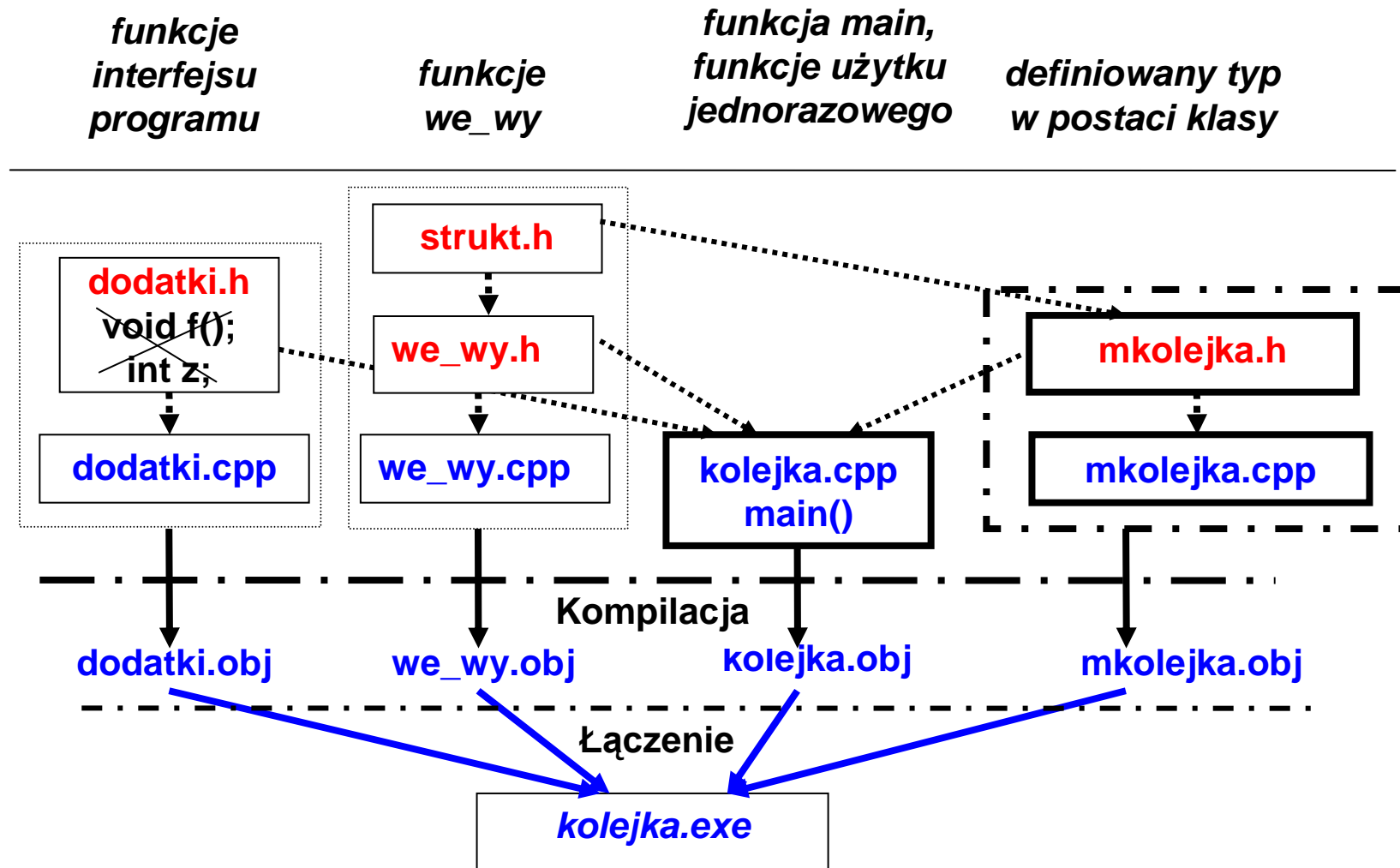
- prototypy funkcji np.
- definicje funkcji typu **inline**
- deklaracje zmiennych
- definicje stałych
- wyliczenia
- deklaracje nazw
- dyrektywy
- funkcje makra
- stałe jawne

Uwaga:

Nie należy nigdy wstawiać do pliku nagłówkowego:

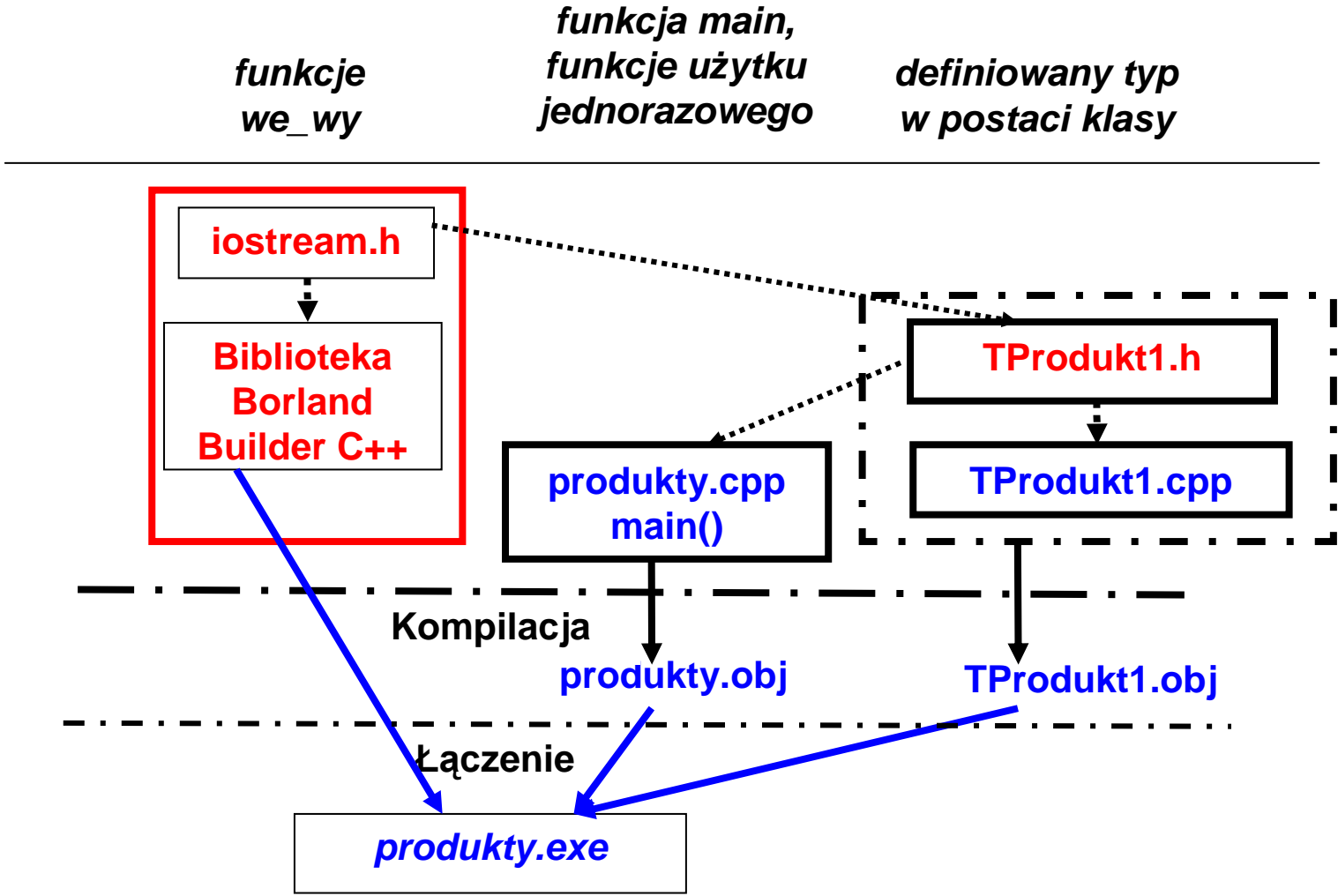
definicji zwykłych funkcji: **int Większy(int x, int y) {return x > y; }**
definicji zmiennych: **int zmienna;**
definicji stałych agregatów np. **const char tab[] = "aaa";**

Przykład podziału programu na moduły



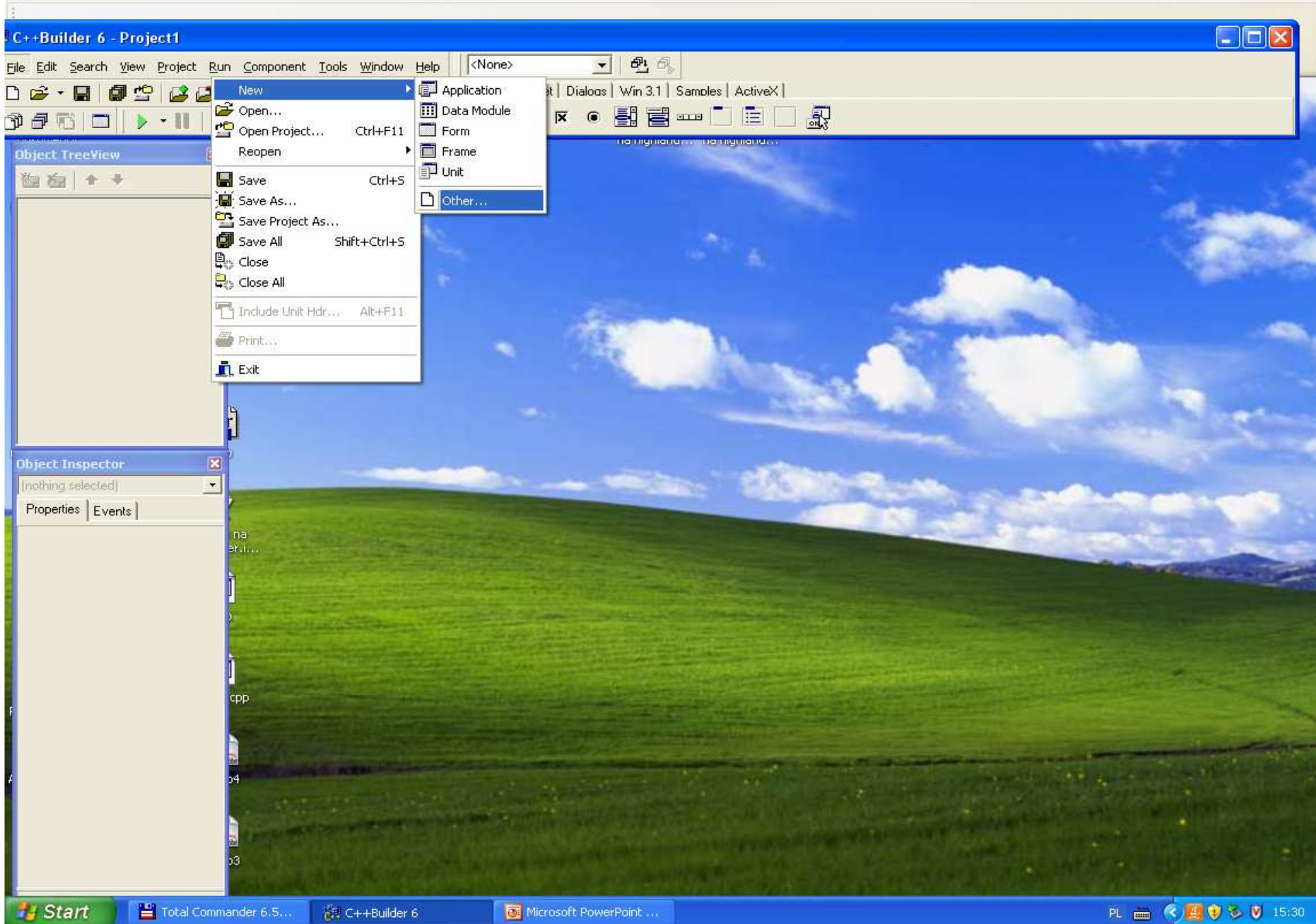
.....> tworzenie jednego pliku, definiowanie nazw z plików nagłówkowych

Przykład podziału programu na moduły

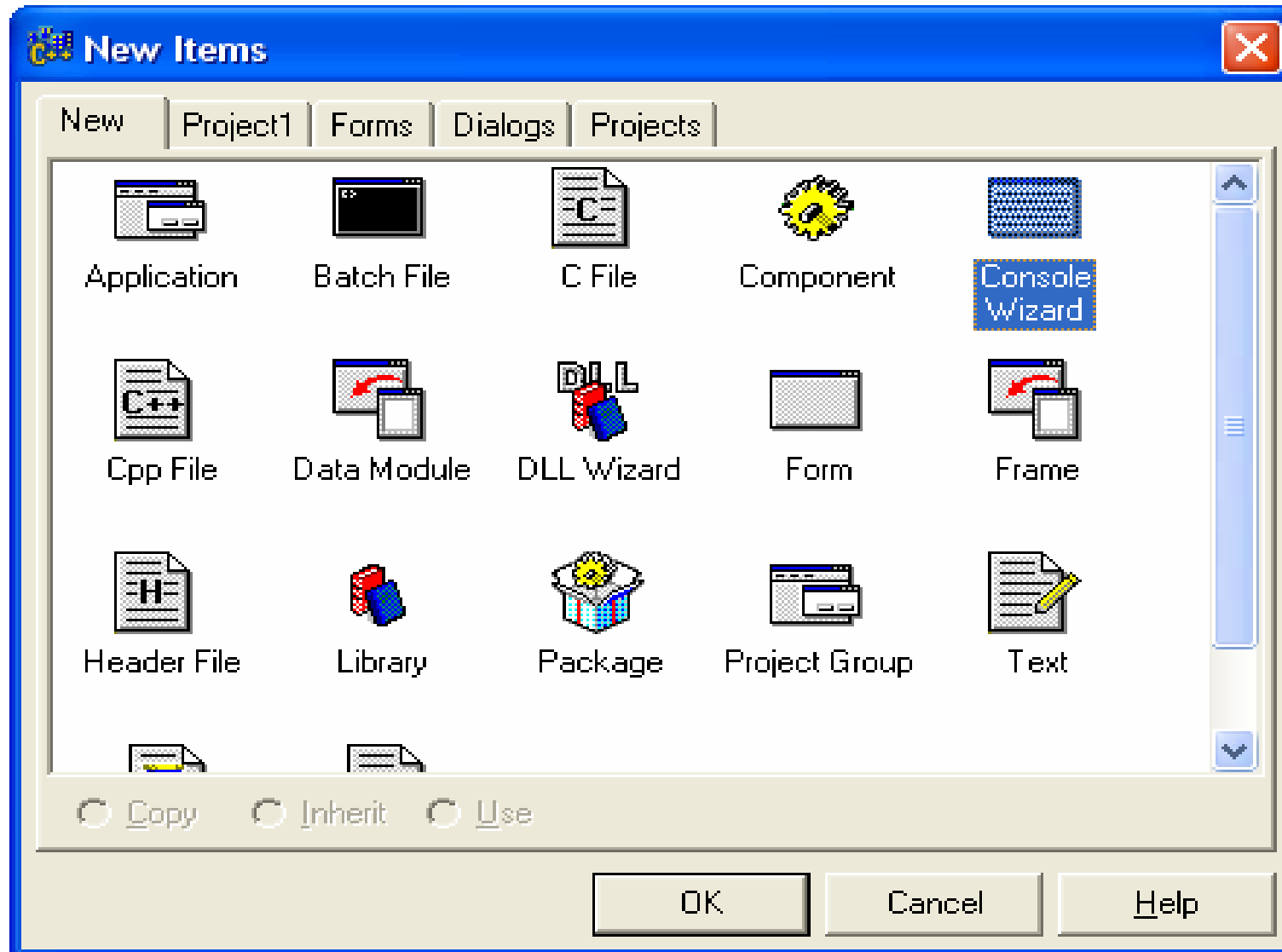


.....> tworzenie jednego pliku, definiowanie nazw z plików nagłówkowych

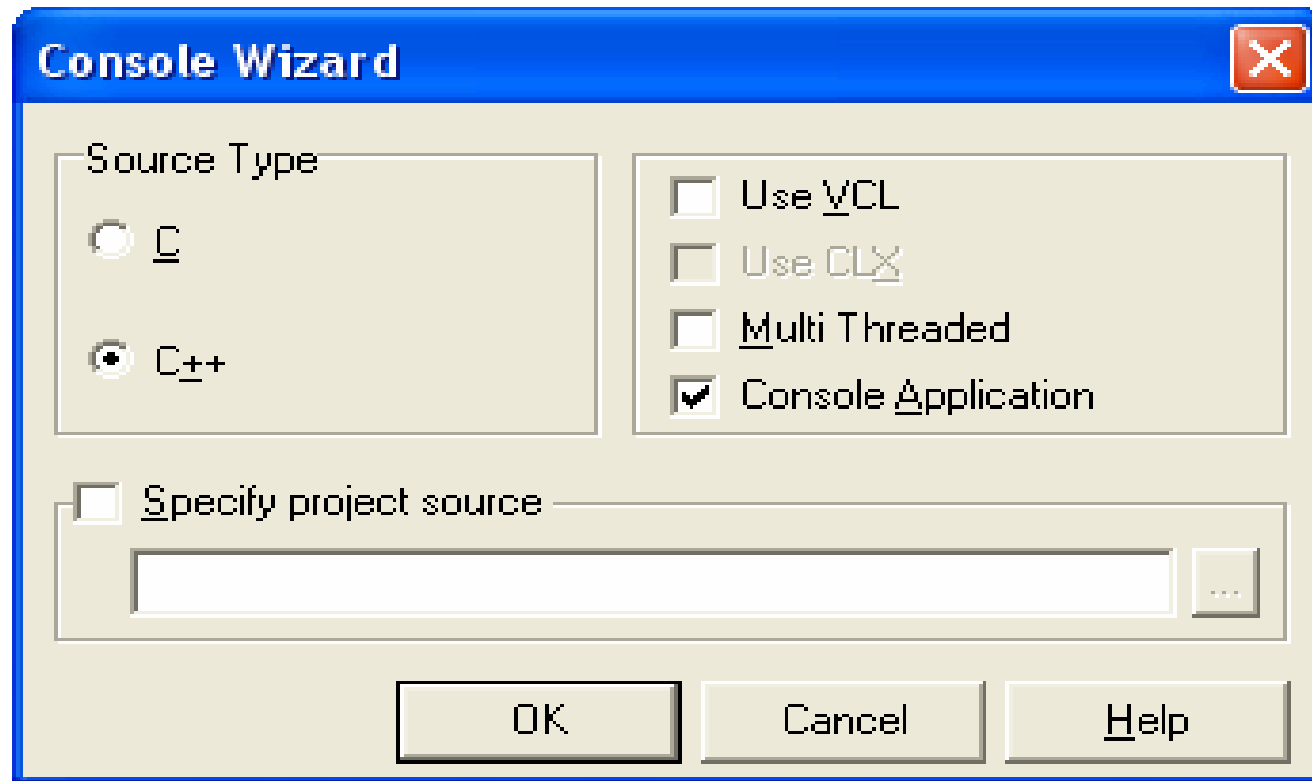
Zakładanie projektu wieloplikowego (**File/New/Other**)



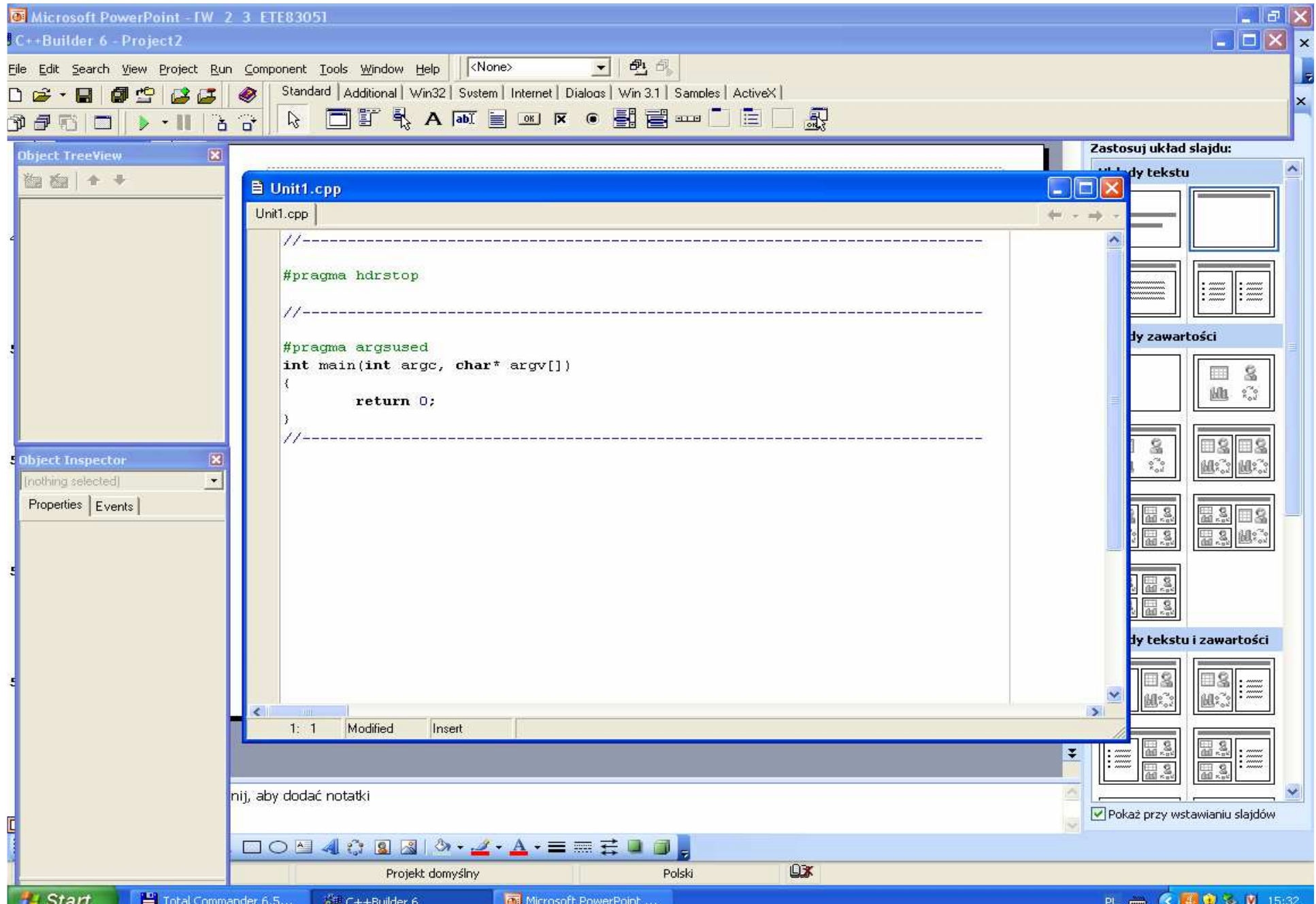
Wybór typu projektu na typ konsolowy (**Console Wizard**)



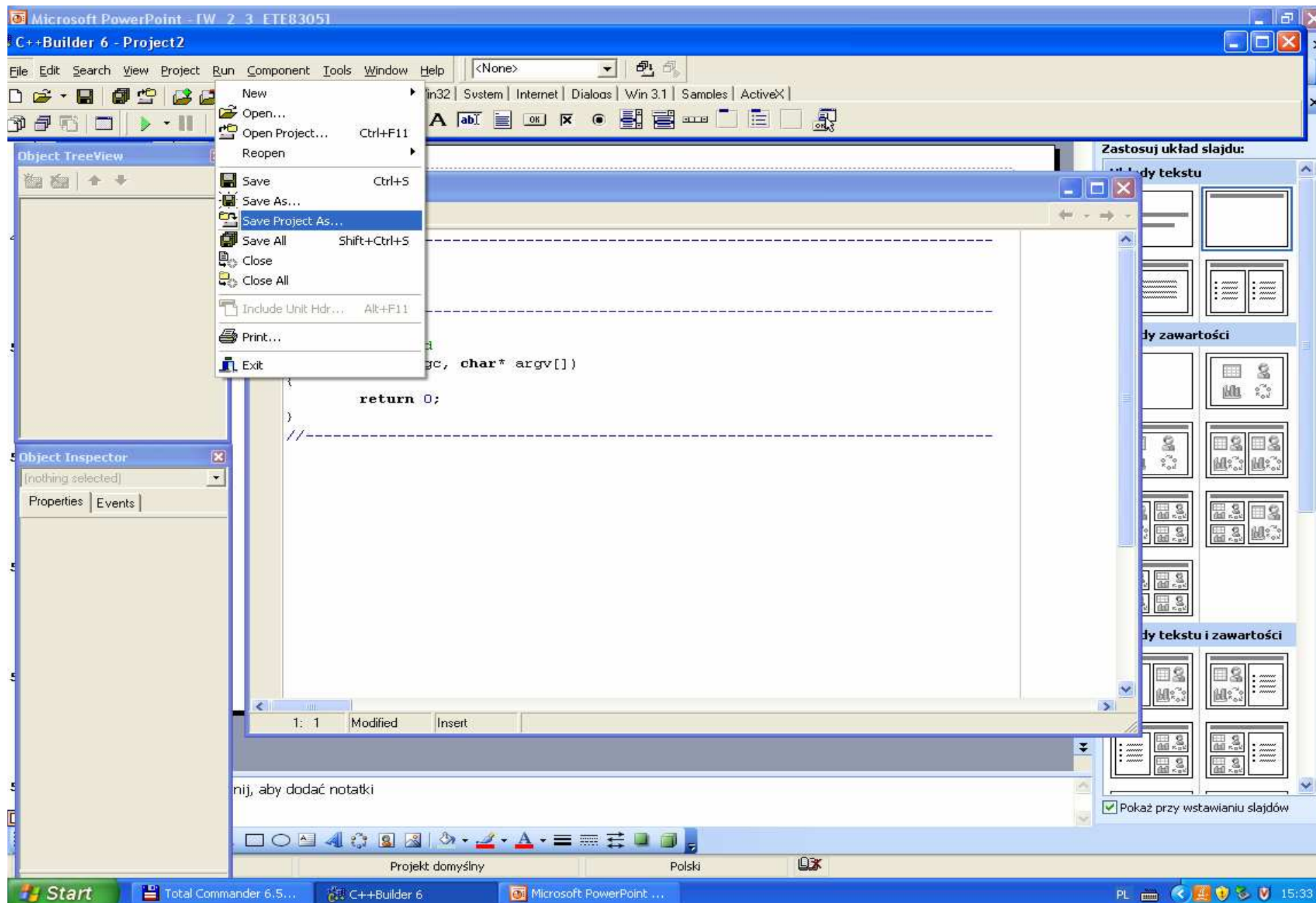
Ustawienie parametrów projektu



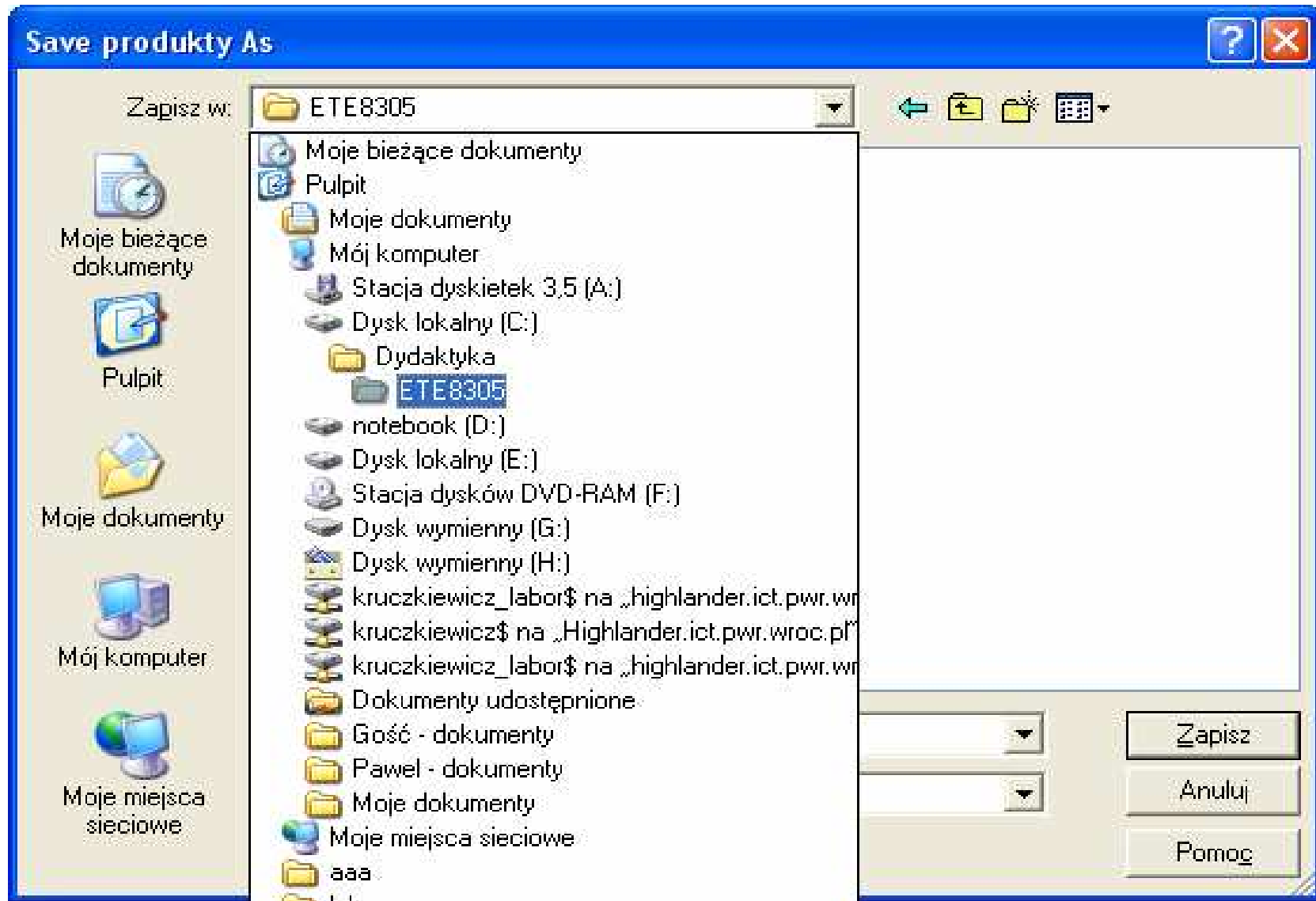
Projekt z domyślnym plikiem Unit1.cpp



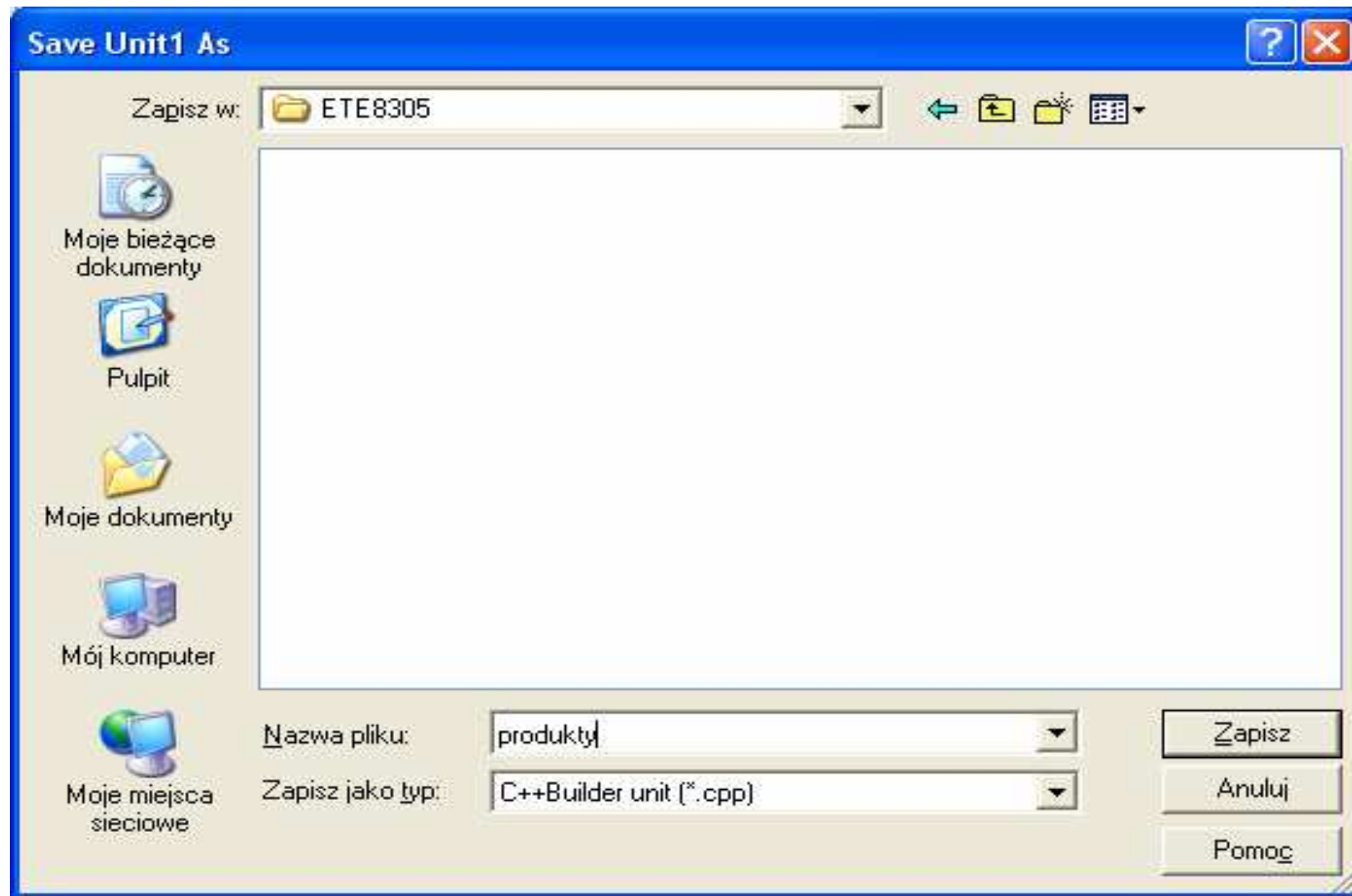
Zmiana nazwy projektu i nazwy domyślnej pliku – zapis projektu w wybranym miejscu na dysku



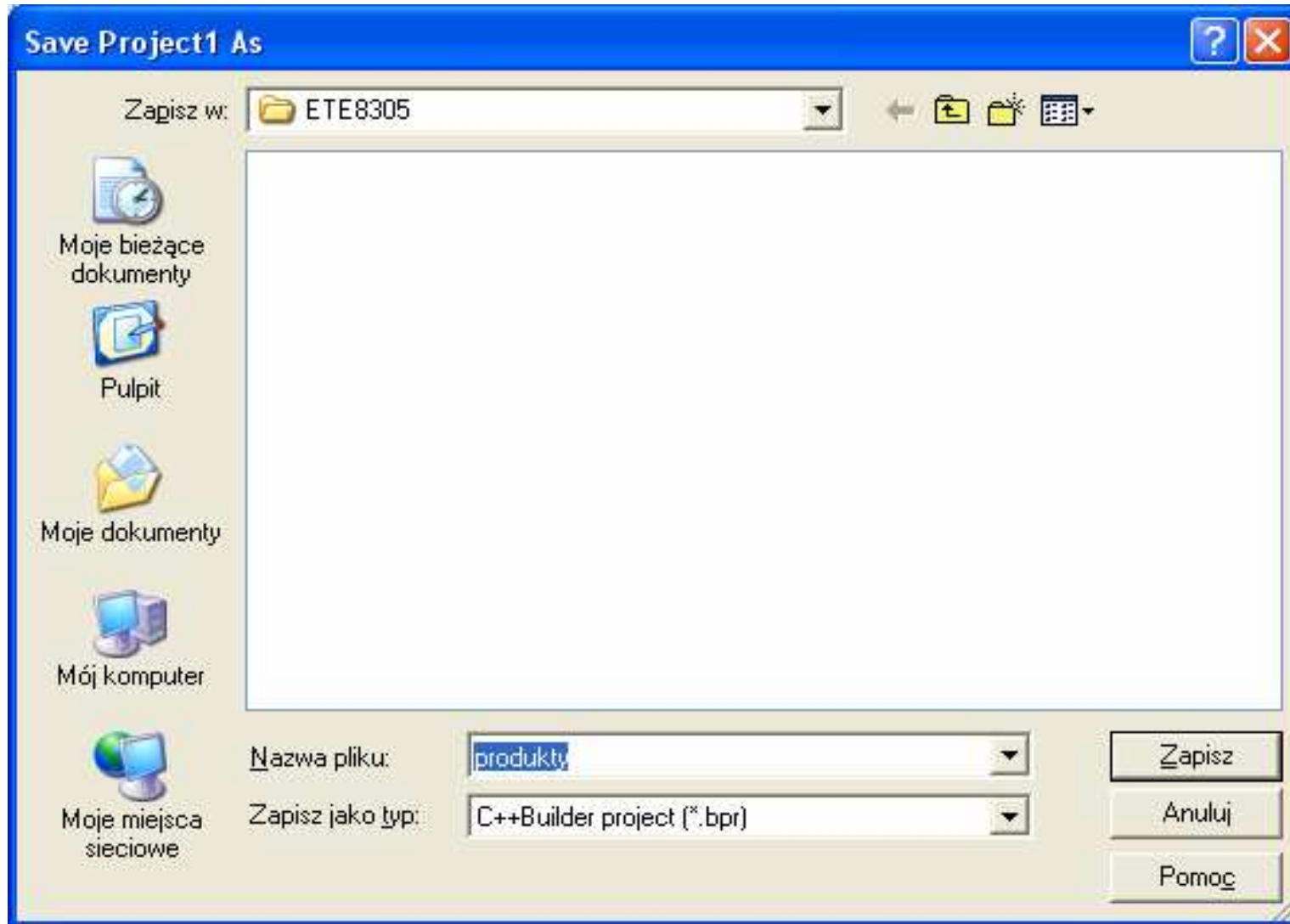
Wybór katalogu projektu



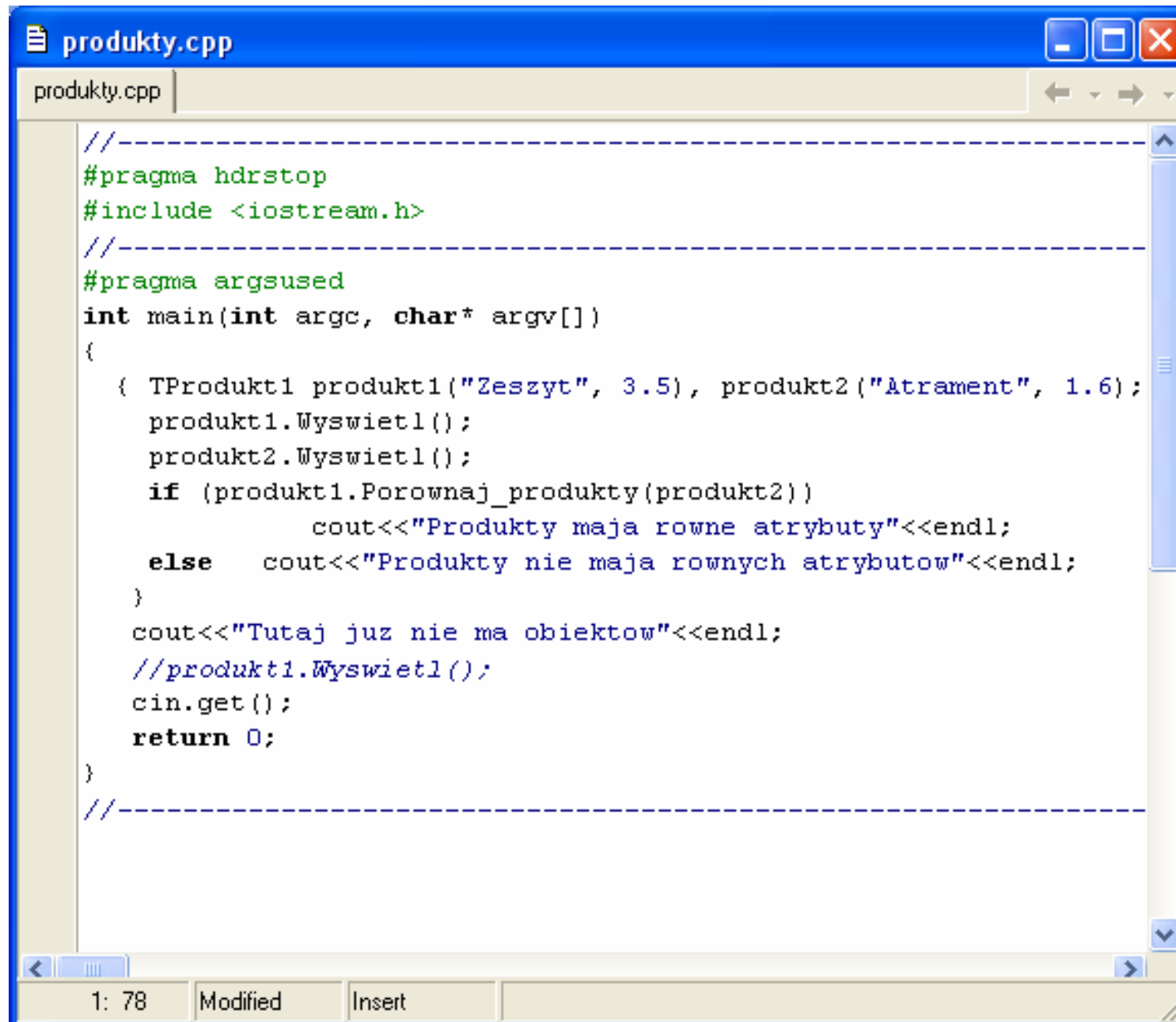
Zmiana nazwy pliku domyślnego zapisywanego w wybranym katalogu



Zmiana nazwy projektu tworzonego w nowym katalogu

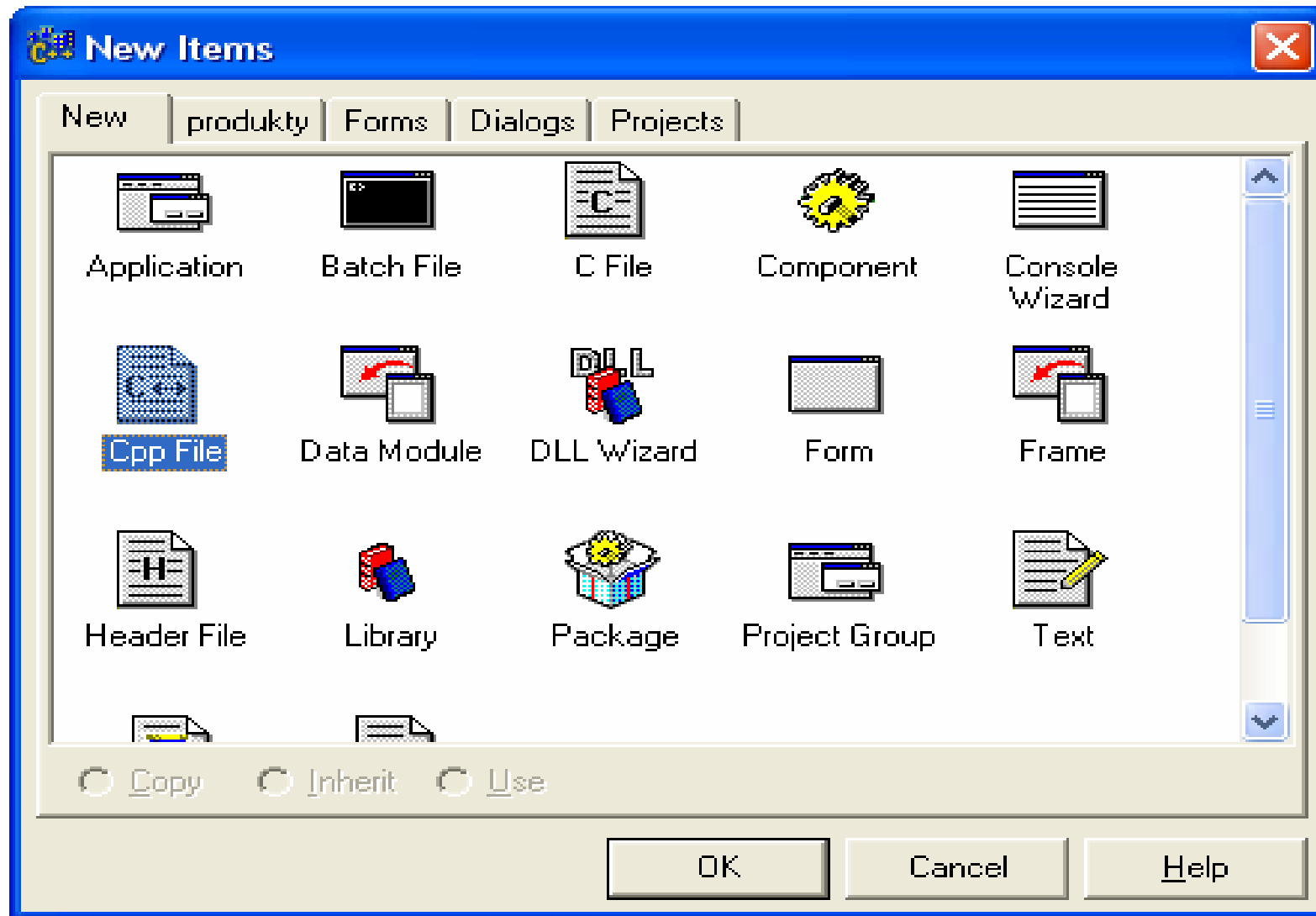


Tworzenie kodu pliku z funkcją main

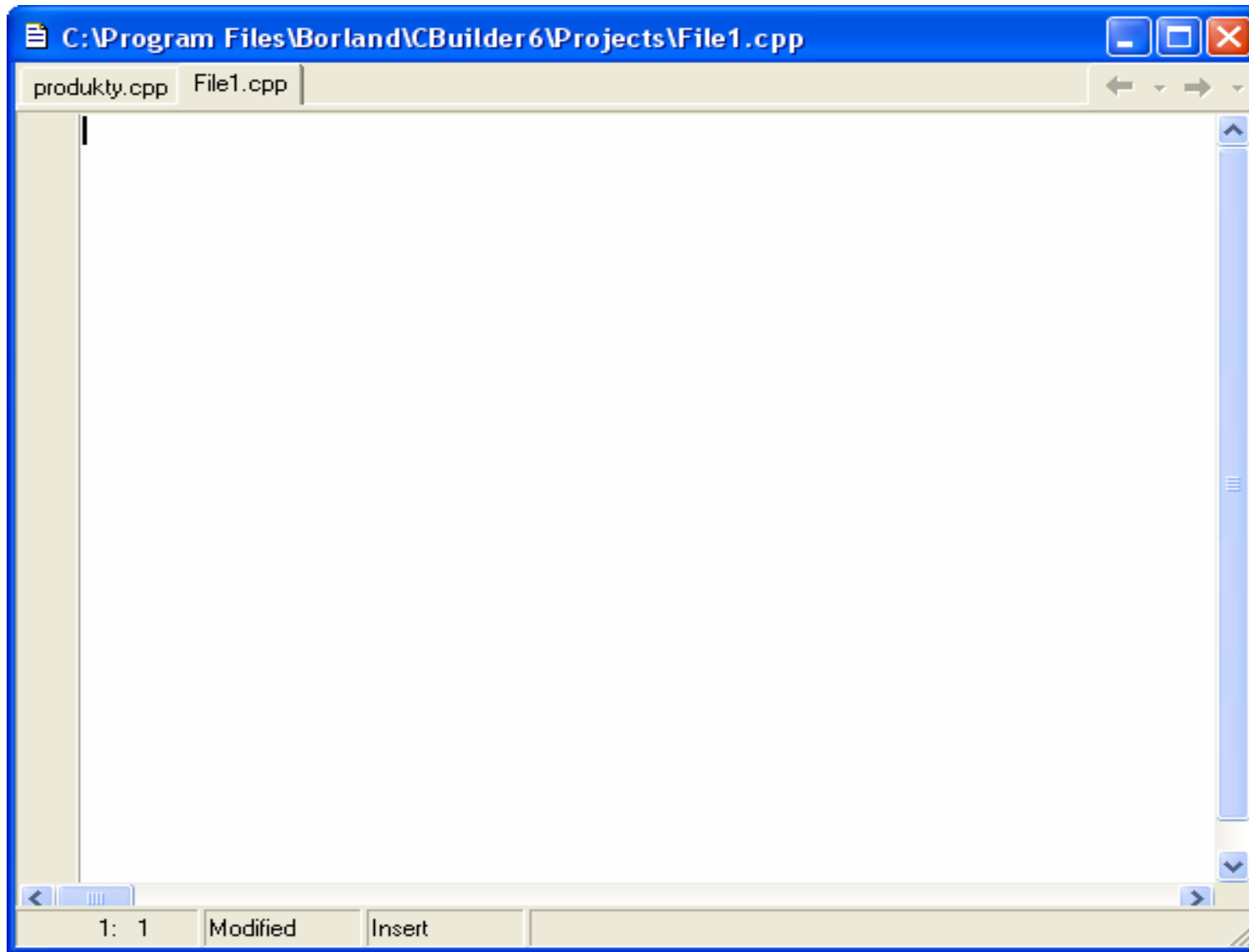


```
//-----  
#pragma hdrstop  
#include <iostream.h>  
//-----  
#pragma argsused  
int main(int argc, char* argv[])  
{  
    ( TProdukt1 produkt1("Zeszyt", 3.5), produkt2("Atrament", 1.6);  
    produkt1.Wyświetl();  
    produkt2.Wyświetl();  
    if (produkt1.Porównaj_produkty(produkt2))  
        cout<<"Produkty maja rowne atrybuty"<<endl;  
    else    cout<<"Produkty nie maja rownych atrybutow"<<endl;  
}  
cout<<"Tutaj juz nie ma obiektow"<<endl;  
//produkt1.Wyświetl();  
cin.get();  
return 0;  
}  
//-----  
1: 78 Modified Insert
```

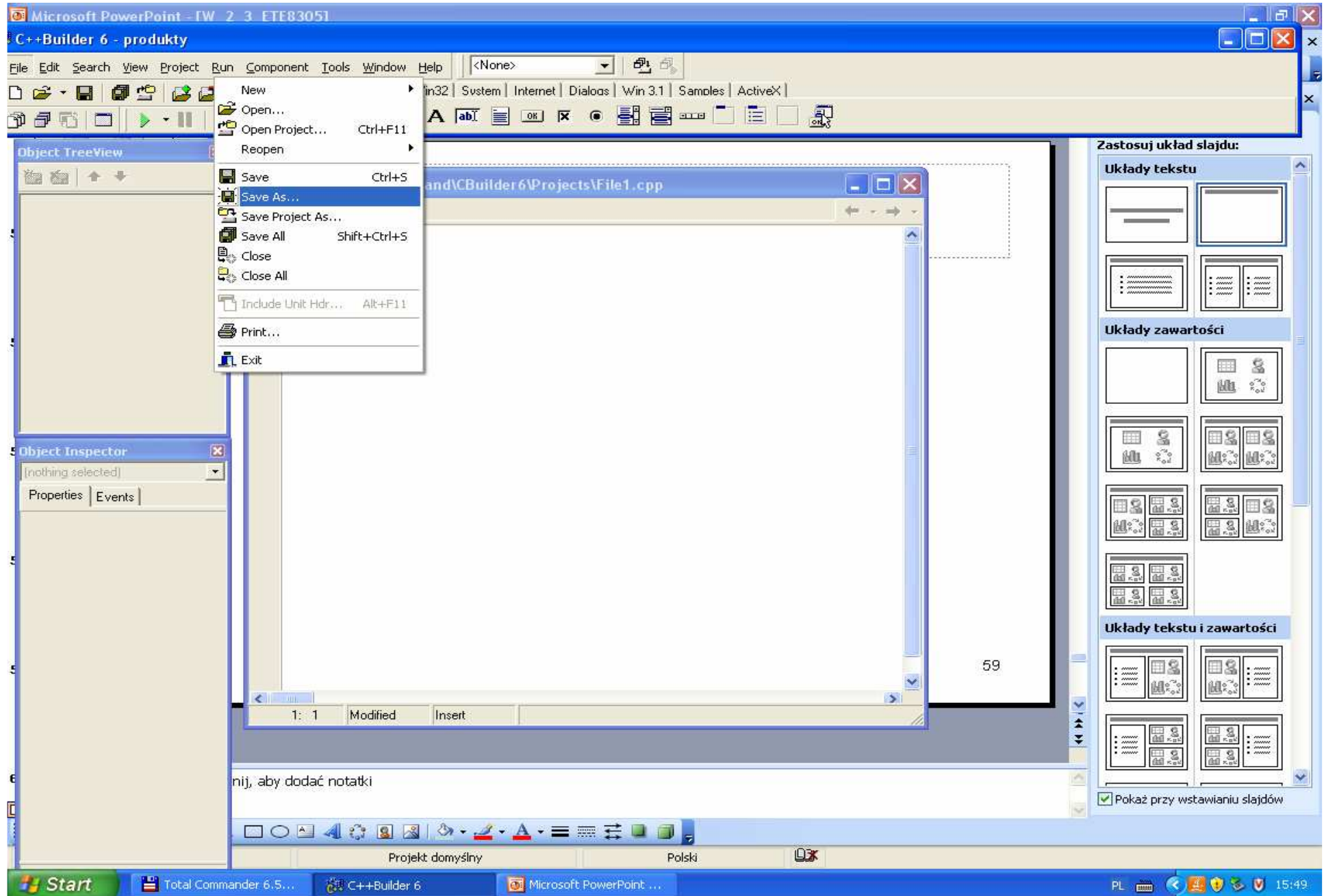

Wybór typu pliku



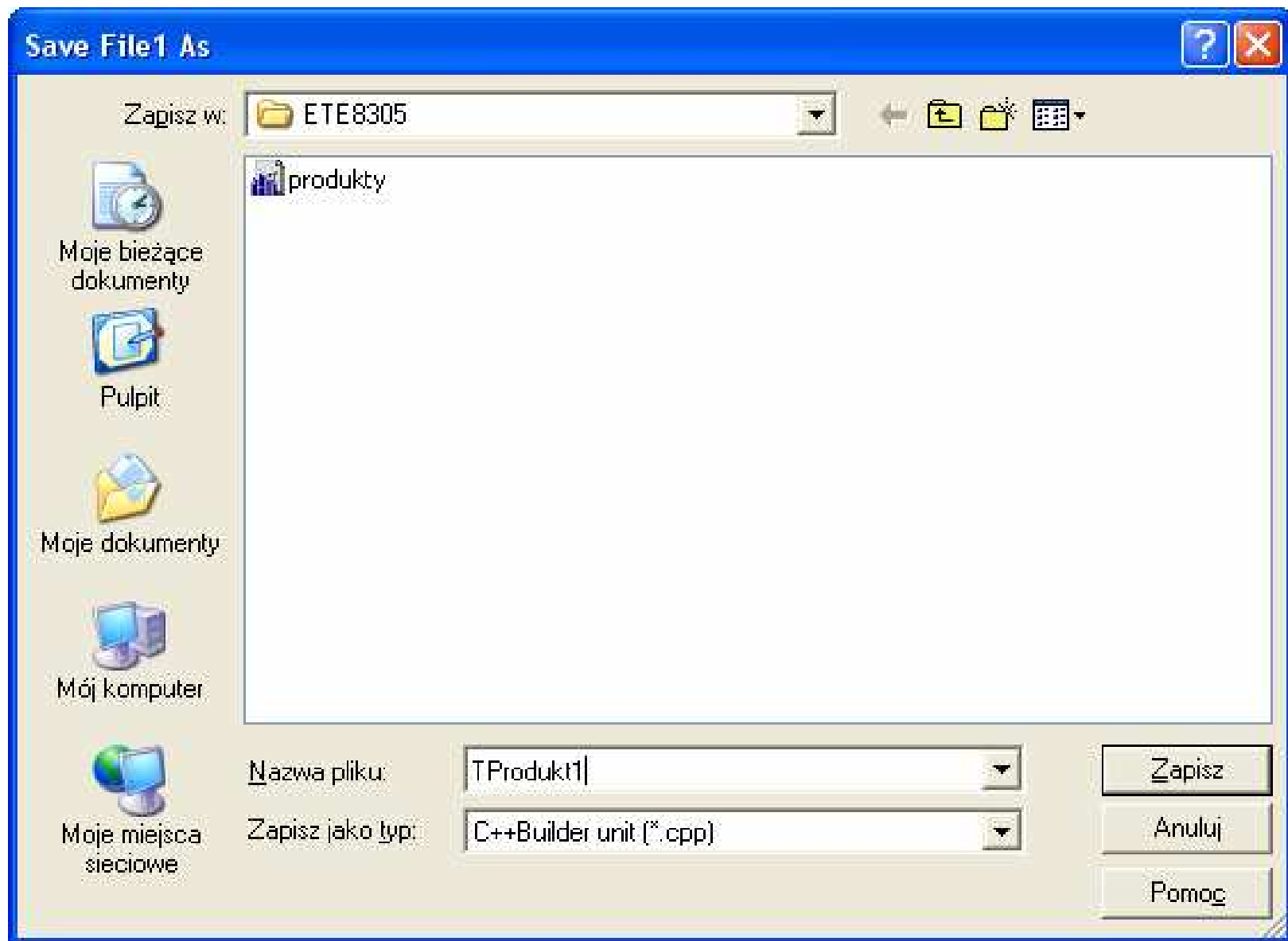
Nowy plik o domyślnej nazwie File1.cpp



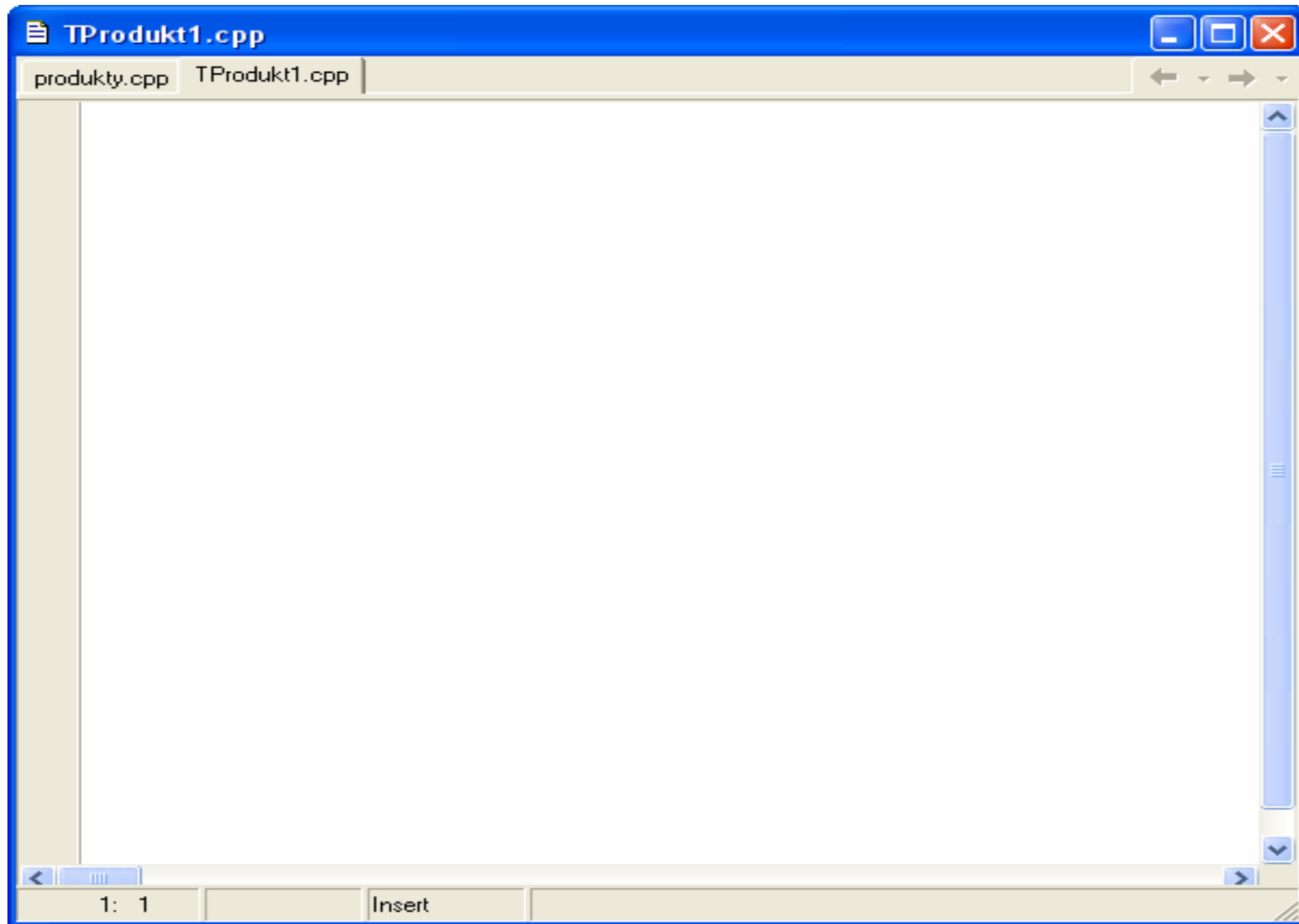
Nadanie nowej nazwy plikowi



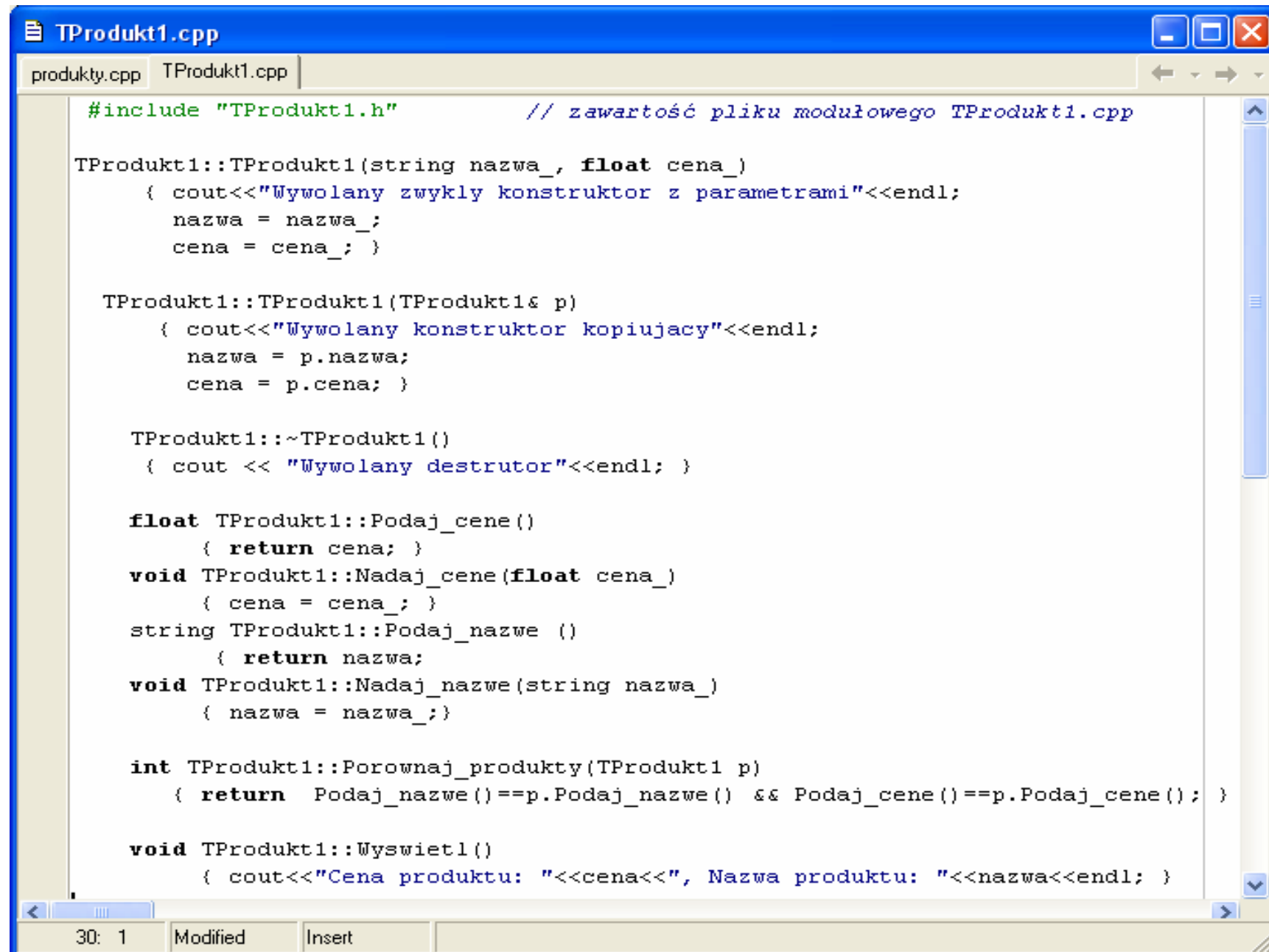
Podanie nowej nazwy TProdukt1.cpp



Plik TProdukt1.cpp - pusty



Wstawienie kodu – definicje metod klasy TProdukt1.cpp



```
TProdukt1.cpp
produkty.cpp TProdukt1.cpp
#include "TProdukt1.h" // zawartość pliku modułowego TProdukt1.cpp

TProdukt1::TProdukt1(string nazwa_, float cena_)
{ cout<<"Wywolany zwykly konstruktor z parametrami"<<endl;
  nazwa = nazwa_;
  cena = cena_; }

TProdukt1::TProdukt1(TProdukt1& p)
{ cout<<"Wywolany konstruktor kopiujacy"<<endl;
  nazwa = p.nazwa;
  cena = p.cena; }

TProdukt1::~TProdukt1()
{ cout << "Wywolany destrutor"<<endl; }

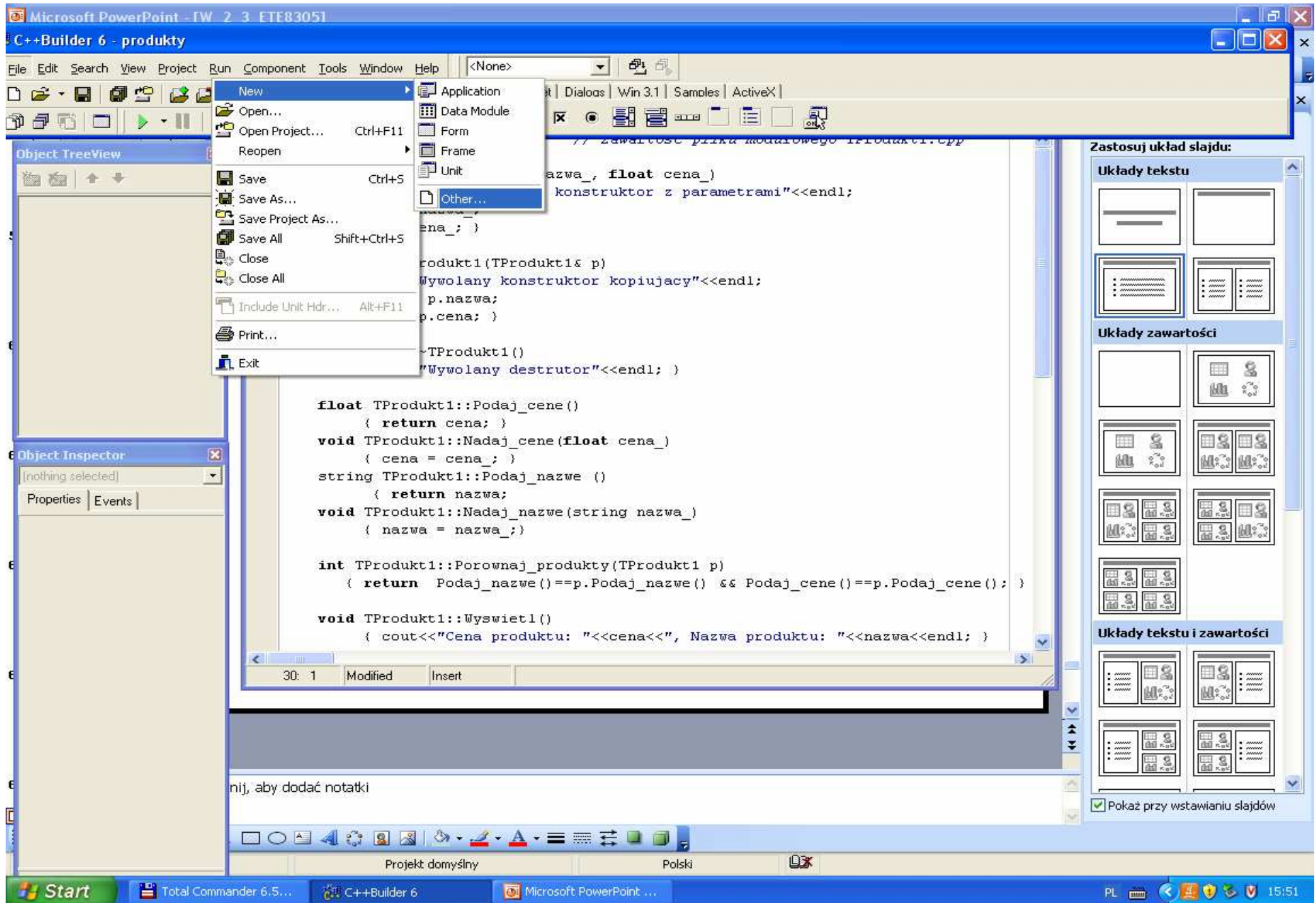
float TProdukt1::Podaj_cene()
{ return cena; }
void TProdukt1::Nadaj_cene(float cena_)
{ cena = cena_; }
string TProdukt1::Podaj_nazwe ()
{ return nazwa; }
void TProdukt1::Nadaj_nazwe(string nazwa_)
{ nazwa = nazwa_; }

int TProdukt1::Porownaj_produkty(TProdukt1 p)
{ return Podaj_nazwe()==p.Podaj_nazwe() && Podaj_cene()==p.Podaj_cene(); }

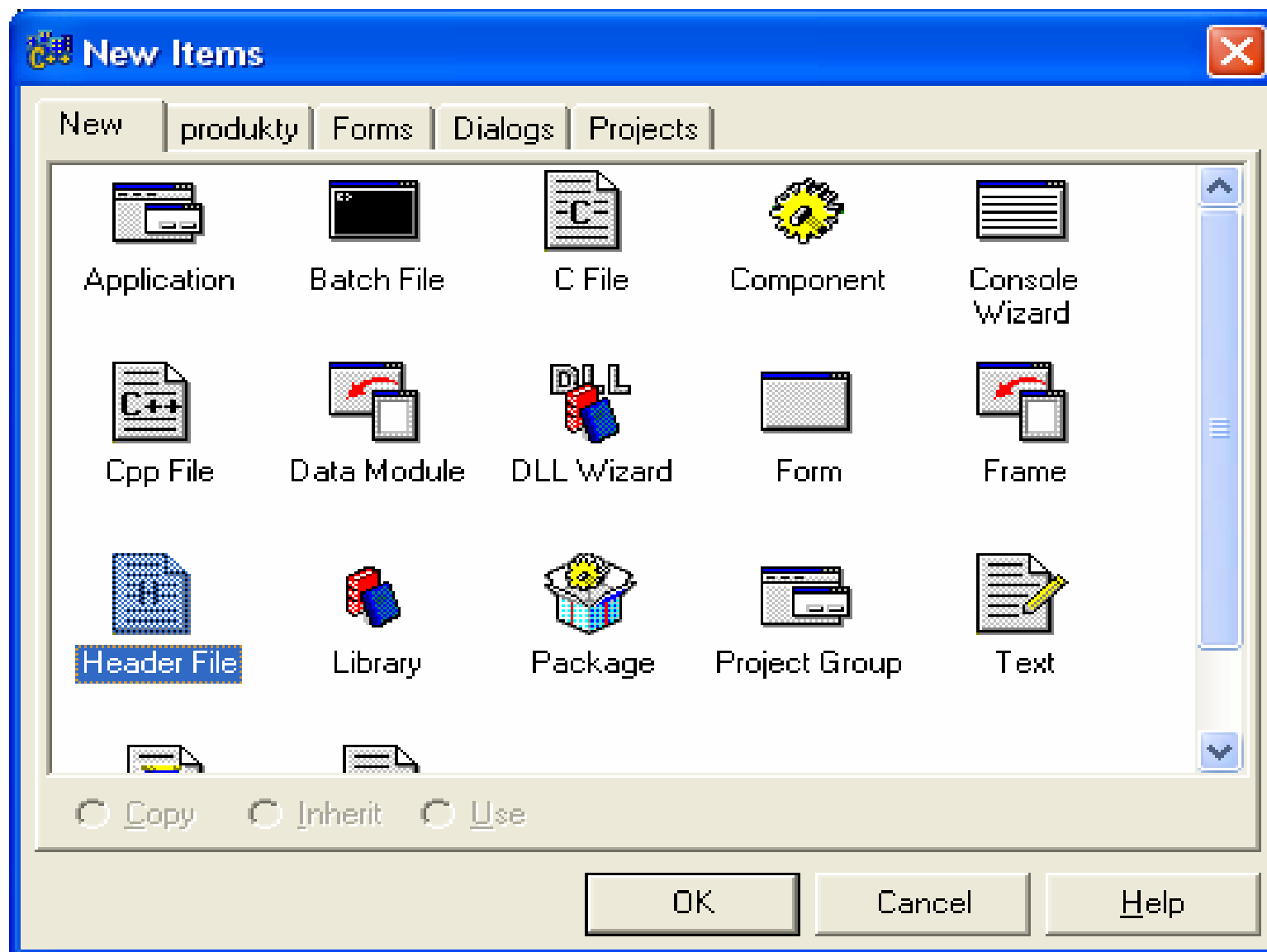
void TProdukt1::Wyswietl()
{ cout<<"Cena produktu: "<<cena<<" , Nazwa produktu: "<<nazwa<<endl; }
```

30: 1 Modified Insert

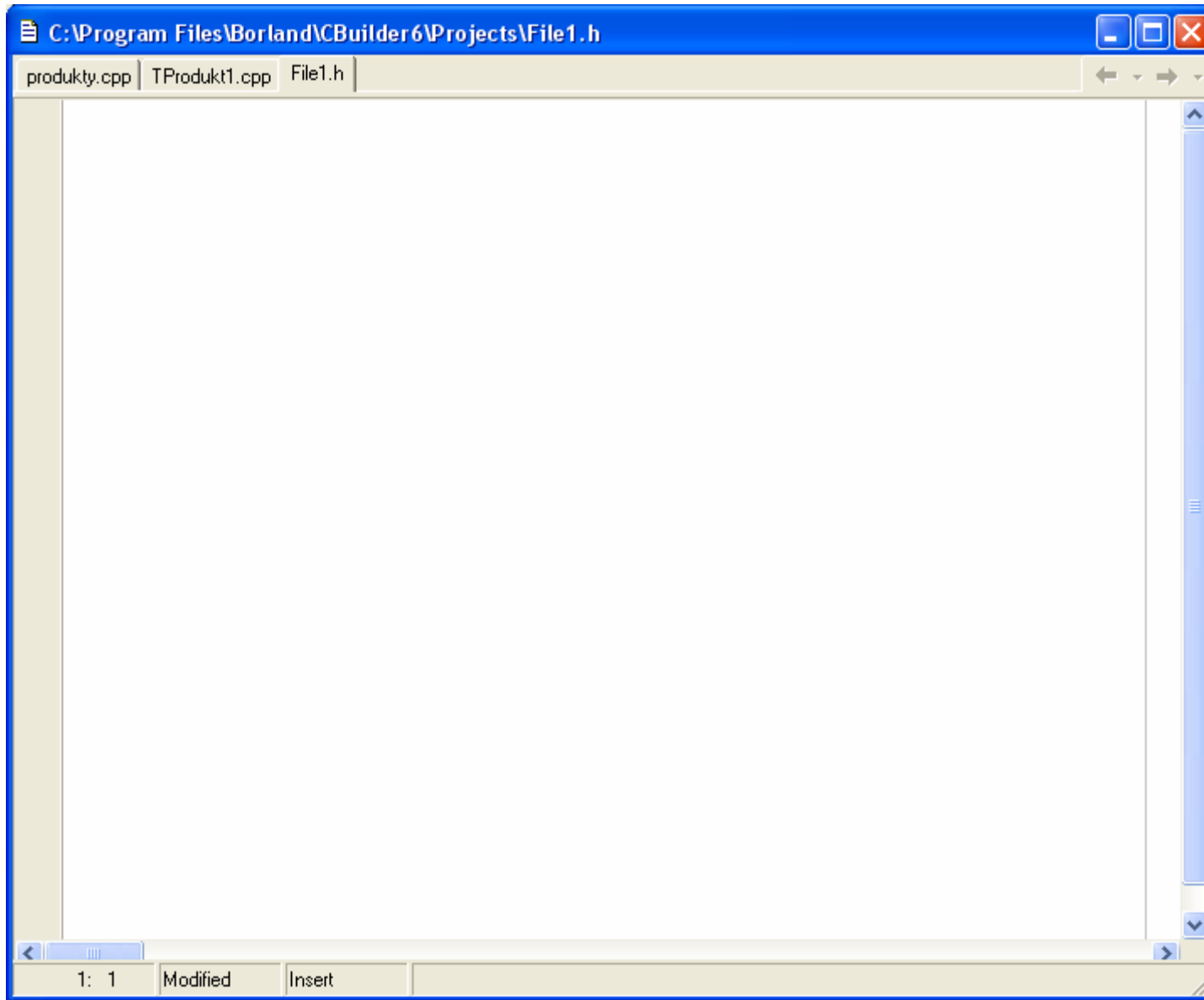
Tworzenie nowego pliku nagłówkowego



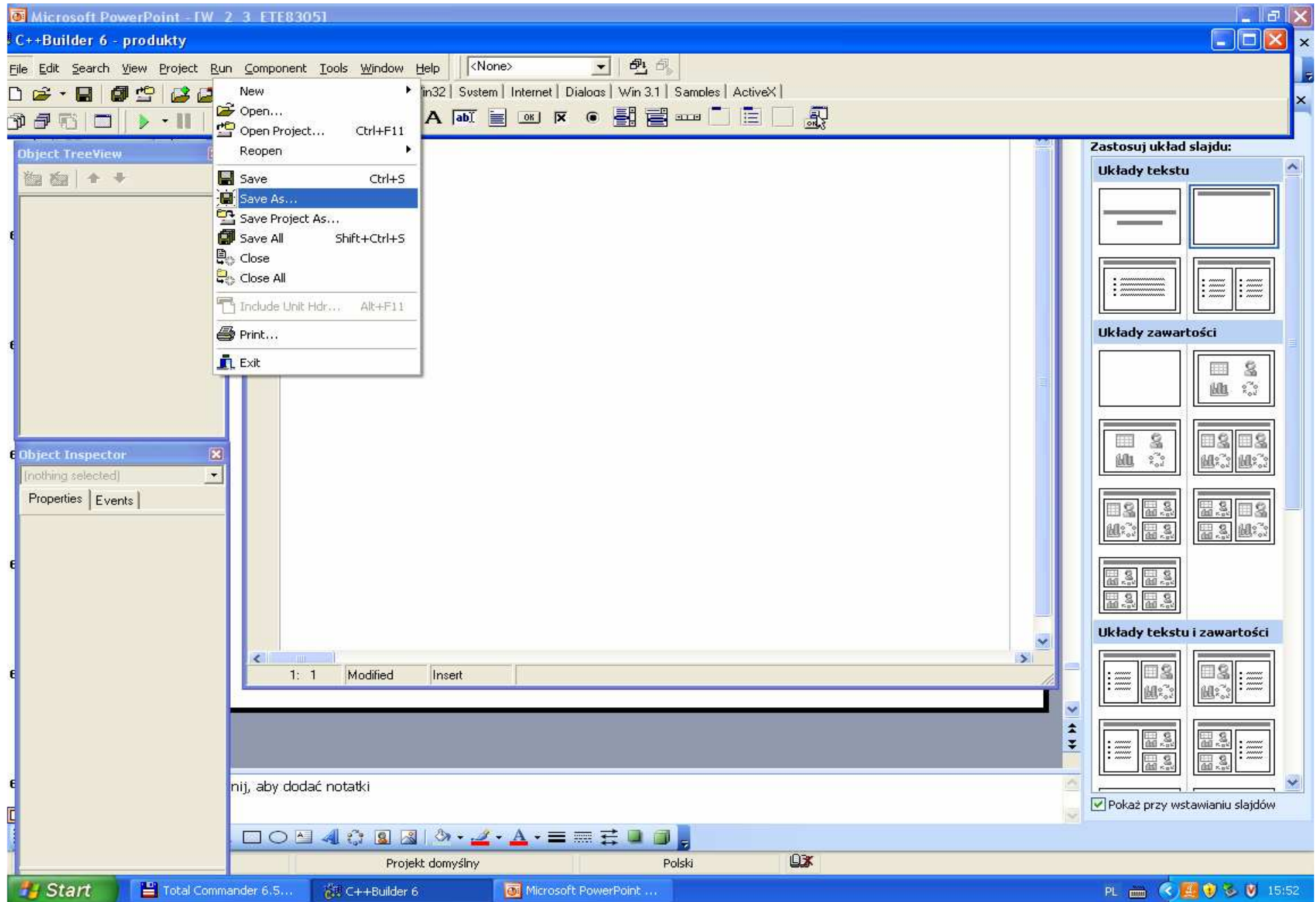
Wybór typu pliku nagłówkowego



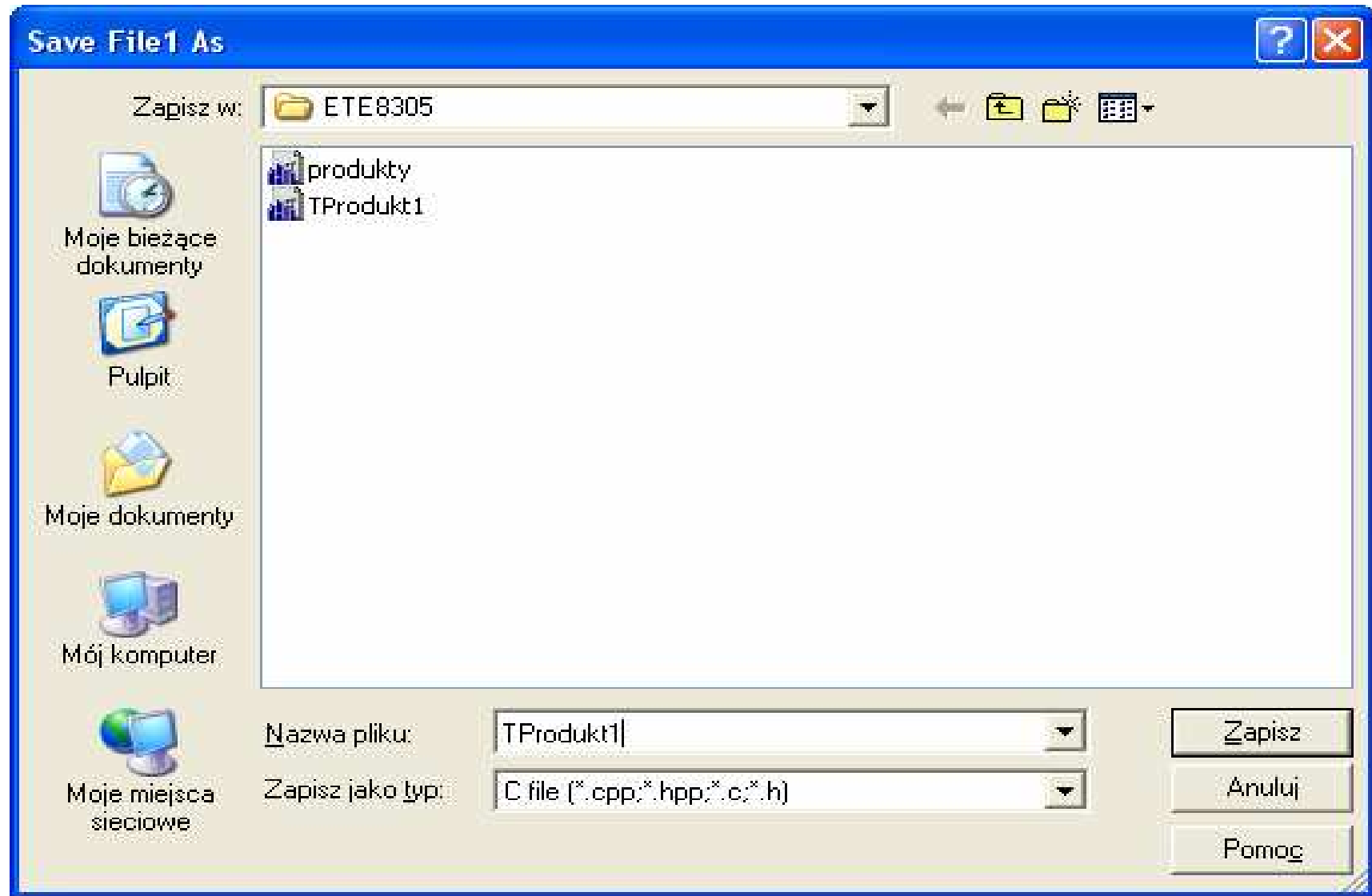
Nowy plik nagłówkowy o nazwie domyślnej File1.h



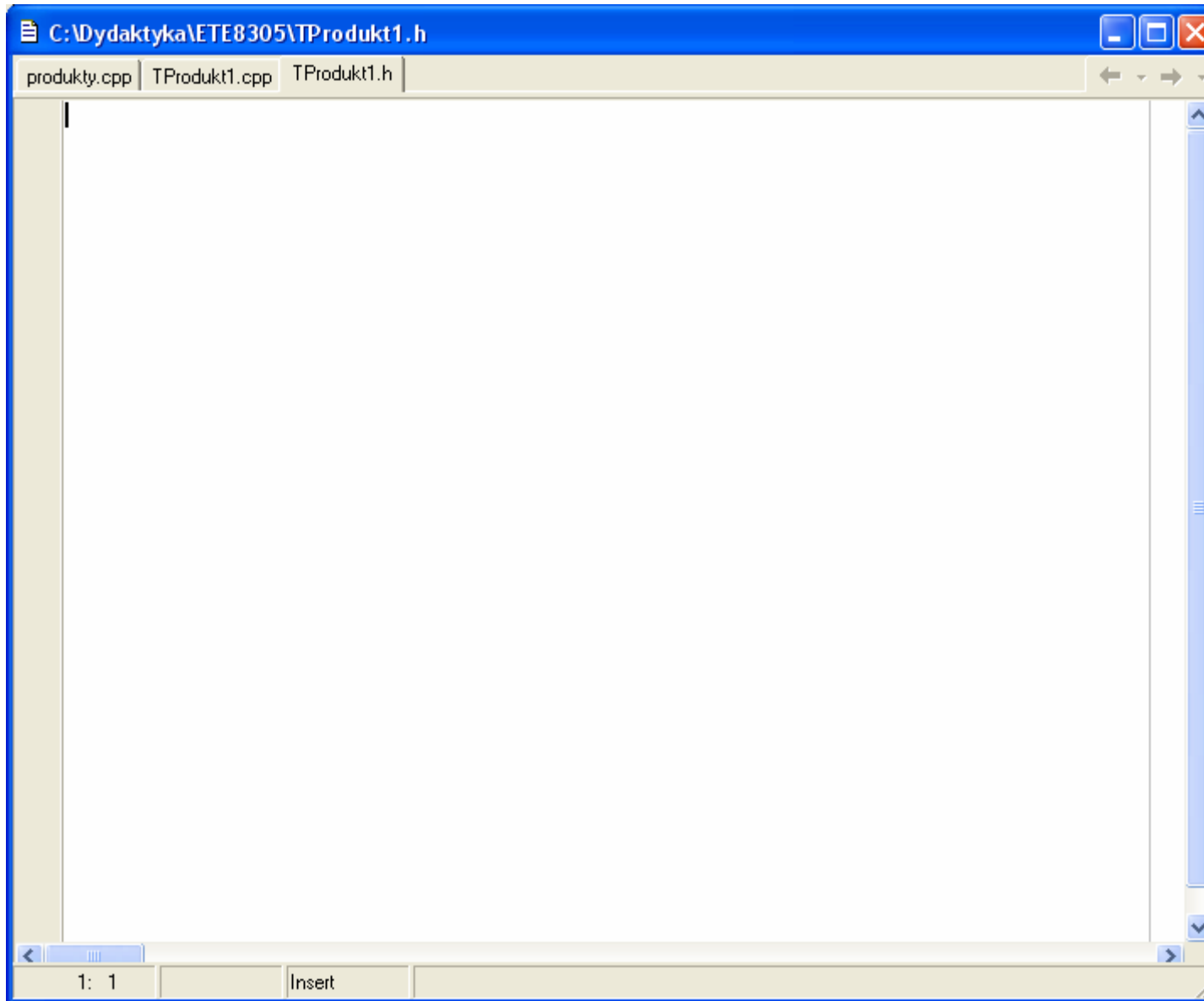
Zmiana nazwy pliku nagłówkowego



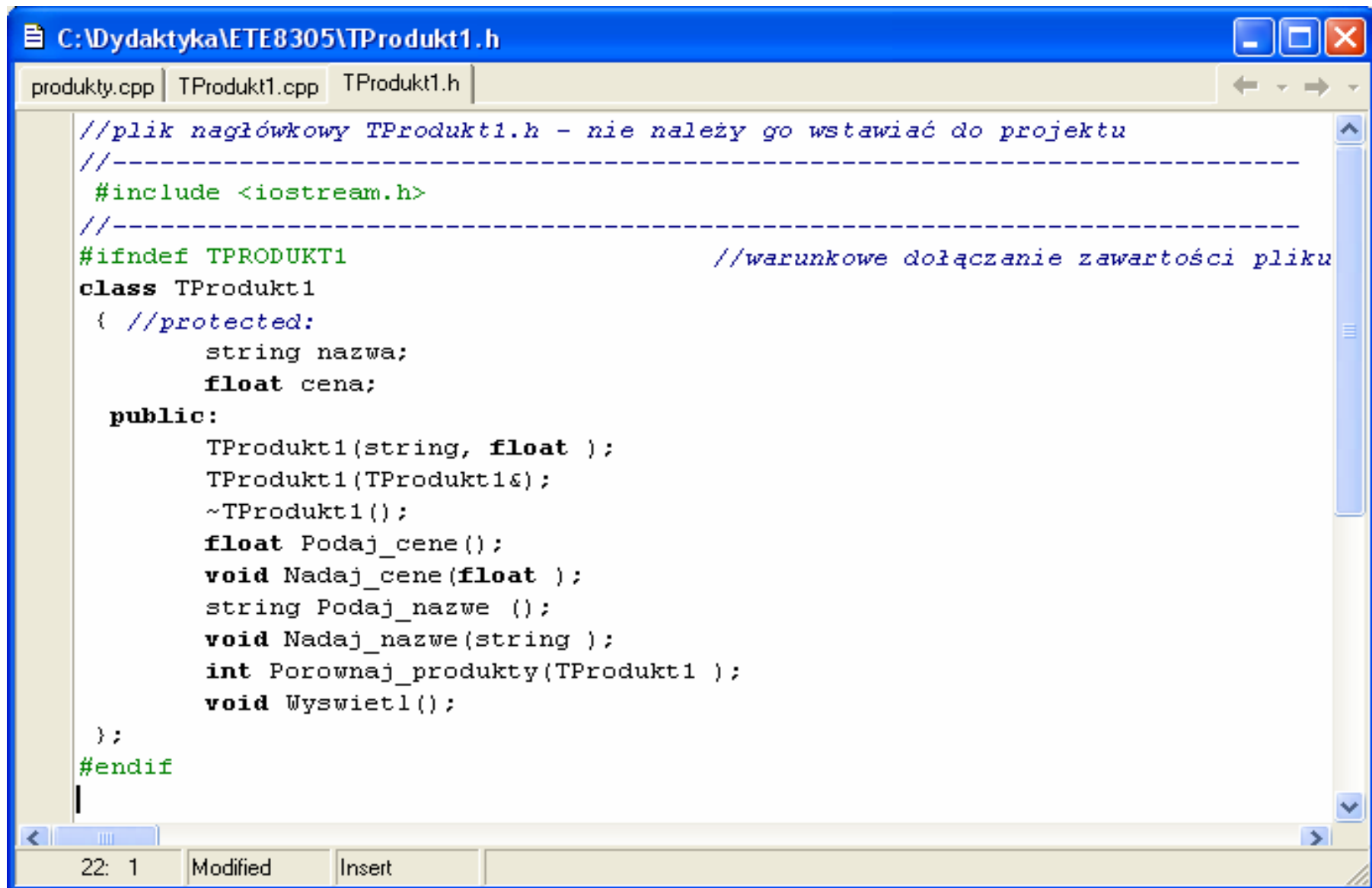
Podanie nazwy TProdukt1.h



Plik nagłówkowy TProdukt1.h - pusty

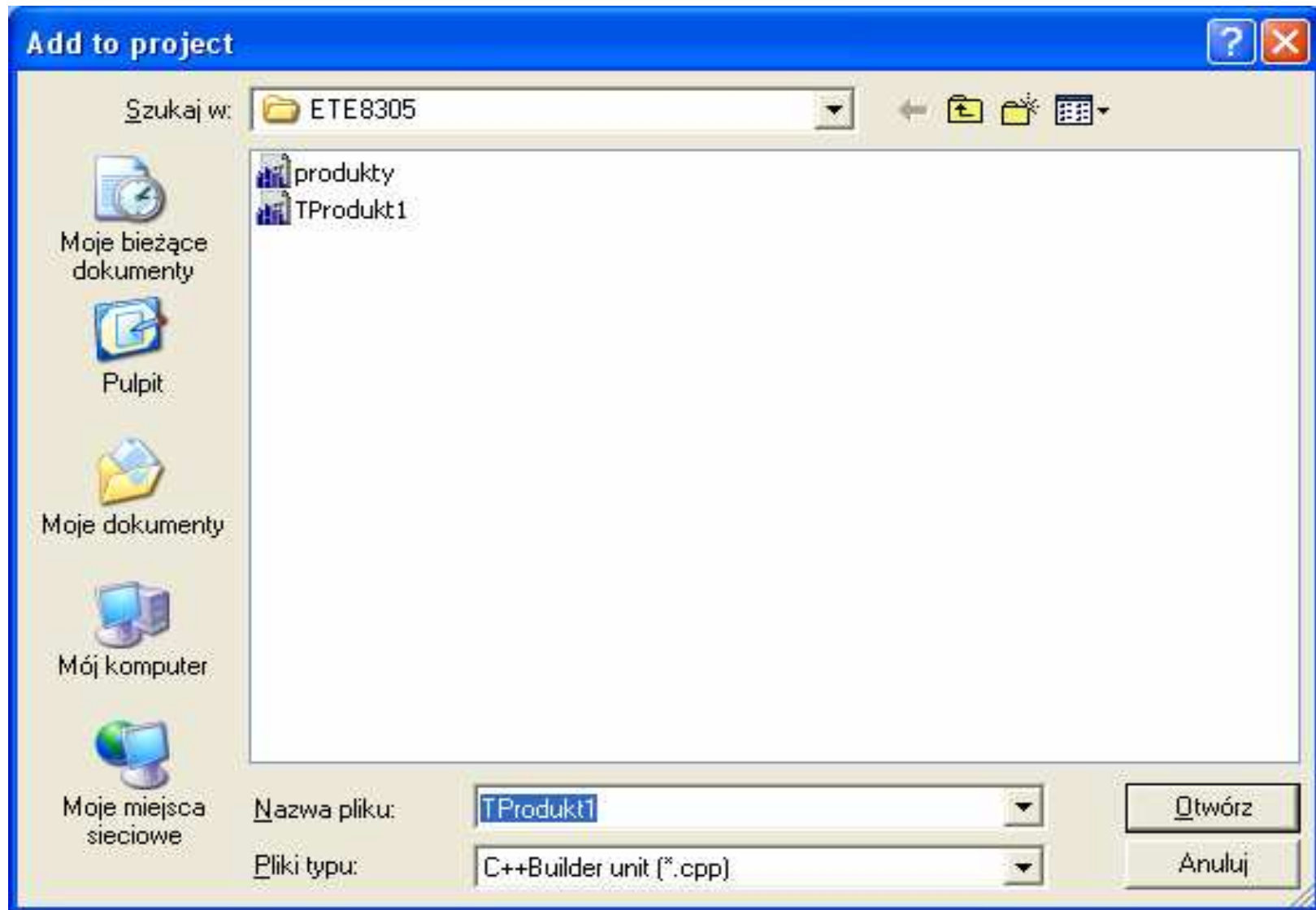


Plik nagłówkowy TProdukt1.h z kodem klasy TProdukt1

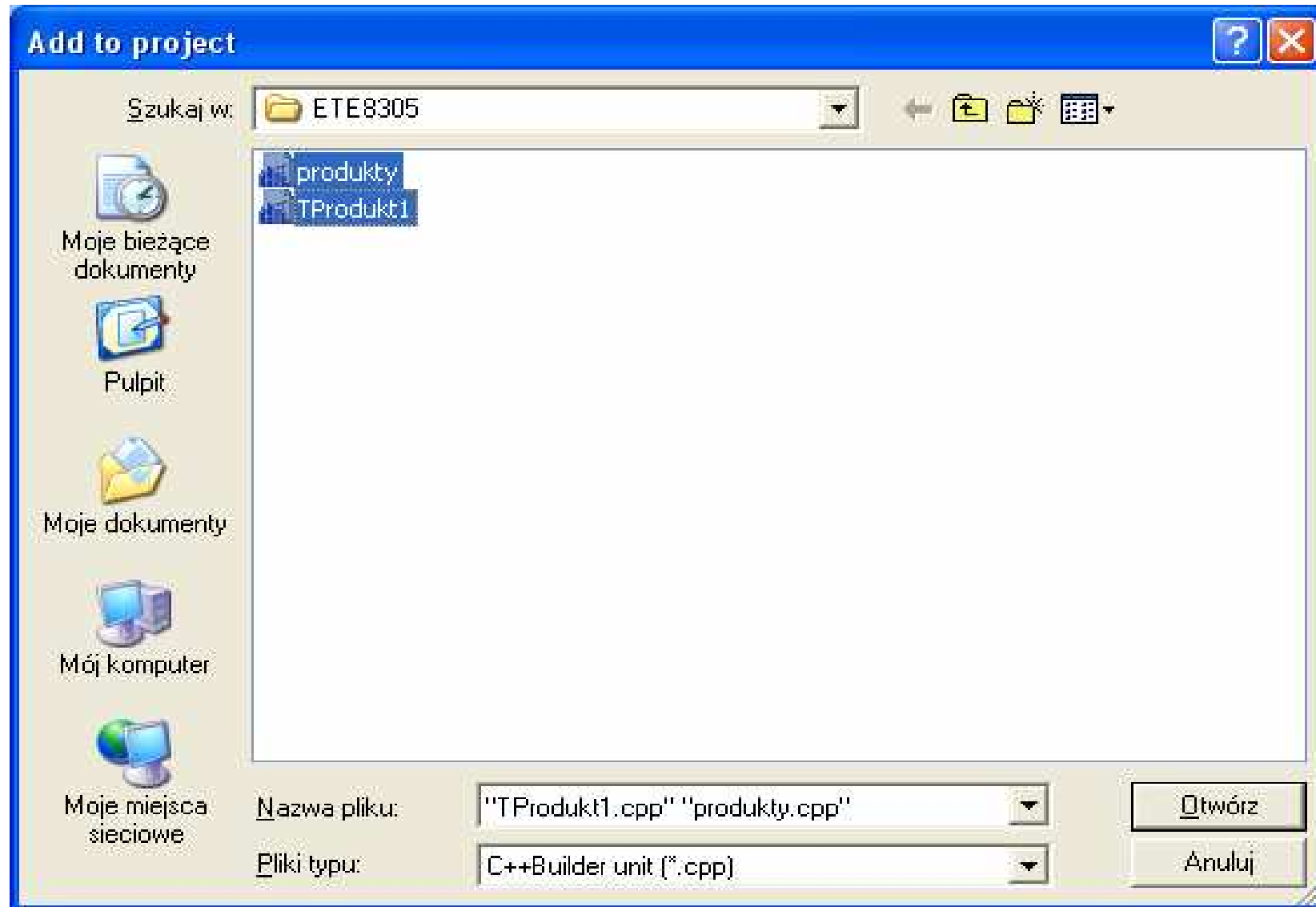


```
C:\Dydaktyka\ETE8305\TProdukt1.h
produkty.cpp | TProdukt1.cpp | TProdukt1.h
//plik nagłówkowy TProdukt1.h - nie należy go wstawiać do projektu
//-----
#include <iostream.h>
//-----
#ifndef TPRODUKT1 //warunkowe dołączanie zawartości pliku
class TProdukt1
{ //protected:
    string nazwa;
    float cena;
public:
    TProdukt1(string, float );
    TProdukt1(TProdukt1&);
    ~TProdukt1();
    float Podaj_cena();
    void Nadaj_cena(float );
    string Podaj_nazwe ();
    void Nadaj_nazwe(string );
    int Porownaj_produkty(TProdukt1 );
    void Wyswietl();
};
#endif
```


Dodawanie plików cpp do projektu



Wybór plików cpp



Kompilacja i linkowanie projektu – Build

The screenshot shows the Borland C++ Builder 6 IDE interface. The main window displays the source code for a project named 'produkty'. The 'Project' menu is open, and the 'Build produkty' option is highlighted. The code in the main window includes a destructor, methods for setting and getting price and name, and a comparison function. The status bar at the bottom indicates the current file is 'Projekt domyślny' and the language is 'Polski'. The taskbar at the bottom shows the Start button and several open applications: Total Commander 6.5..., C++Builder 6, and Microsoft PowerPoint.

```
...
    at << "Wywolany destrutor"<<endl; }

TProdukt1::Podaj_cena()
    return cena; }
TProdukt1::Nadaj_cena(float cena_)
    cena = cena; }
TProdukt1::Podaj_nazwe ()
    { return nazwa; }
TProdukt1::Nadaj_nazwe(string nazwa_)
    nazwa = nazwa_;}

TProdukt1::Porownaj_produkty(TProdukt1 p)
    { return Podaj_nazwe() == p.Podaj_nazwe() && Podaj_cena() == p.Podaj_cena() }

void TProdukt1::Wyswietl()
    { cout<<"Cena produktu: "<<cena<<"", Nazwa produktu: "<<nazwa<<endl; }
...

```

Zastosuj układ slajdu:

Układy tekstu

Układy zawartości

Układy tekstu i zawartości

Pokaż przy wstawianiu slajdów

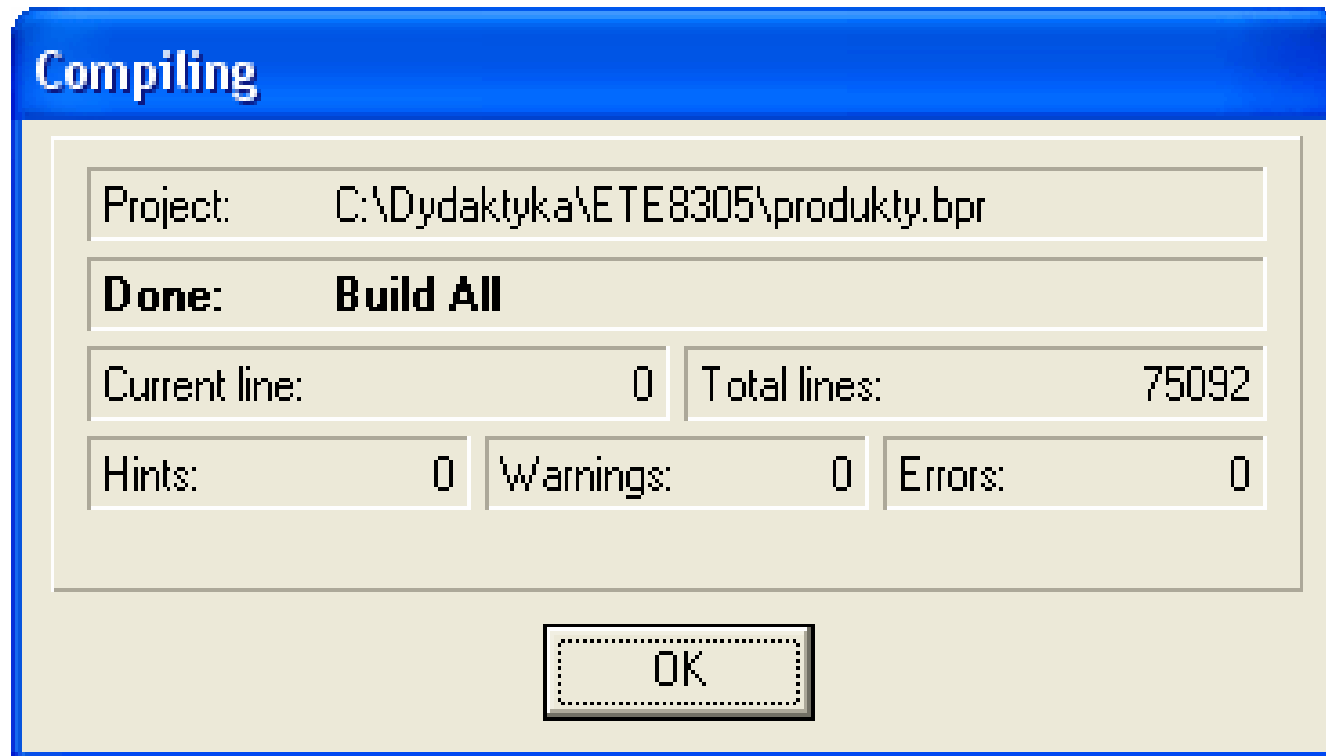
21: 27 Modified Insert

Zofia Kruczkiewicz, ETE8305_2 73

Projekt domyślny Polski

Start Total Commander 6.5... C++Builder 6 Microsoft PowerPoint ... PL 15:59

Okno potwierdzające poprawność kompilacji i linkowania



Działający program konsolowy

```
C:\dydaktyka\ETE8305\produkty.exe

Wywolany zwykly konstruktor z parametrami
Wywolany zwykly konstruktor z parametrami
Cena produktu: 3.5, Nazwa produktu: Zeszyt
Cena produktu: 1.6, Nazwa produktu: Atrament
Wywolany konstruktor kopiujacy
Wywolany destrutor
Produkty nie maja rownych atrybutow
Wywolany destrutor
Wywolany destrutor
Tutaj juz nie ma obiektow
```

Program 6 – program wieloplikowy

//plik nagłówekowy TProdukt1.h – nie należy go wstawiać do projektu

//-----

#include <iostream.h>

//-----

#ifndef TPRODUKT1

//warunkowe dołączanie zawartości pliku

#define TPRODUKT1

// nagłówekowego

class TProdukt1

{ **//protected:**

string nazwa;

float cena;

public:

TProdukt1(string, float);

TProdukt1(TProdukt1&);

~TProdukt1();

float Podaj_cene();

void Nadaj_cene(**float**);

string Podaj_nazwe ();

void Nadaj_nazwe(string);

int Porownaj_produkty(**TProdukt1**);

void Wyszwietl();

};

#endif

#include "TProdukt1.h"

// zawartość pliku modułowego TProdukt1.cpp

TProdukt1::TProdukt1(string nazwa_, float cena_)

```
{ cout<<"Wywołany zwykły konstruktor z parametrami"<<endl;
  nazwa = nazwa_;
  cena = cena_; }
```

TProdukt1::TProdukt1(TProdukt1& p)

```
{ cout<<"Wywołany konstruktor kopiujący"<<endl;
  nazwa = p.nazwa;
  cena = p.cena; }
```

TProdukt1::~~TProdukt1()

```
{ cout << "Wywołany destrutor"<<endl; }
```

float TProdukt1::Podaj_cene()

```
{ return cena; }
```

void TProdukt1::Nadaj_cene(float cena_)

```
{ cena = cena_; }
```

string TProdukt1::Podaj_nazwe ()

```
{ return nazwa; }
```

void TProdukt1::Nadaj_nazwe(string nazwa_)

```
{ nazwa = nazwa_;}
```

int TProdukt1::Porownaj_produkty(TProdukt1 p)

```
{ return Podaj_nazwe()==p.Podaj_nazwe() && Podaj_cene()==p.Podaj_cene(); }
```

void TProdukt1::Wyswietl()

```
{ cout<<"Cena produktu: "<<cena<<" , Nazwa produktu: "<<nazwa<<endl; }
```

```

//-----
#include "TProdukt1.h"      //zawartość pliku z funkcją main
//-----
int main(int argc, char* argv[])
{
    { TProdukt1 produkt1("Zeszyt", 3.5), produkt2("Atrament", 1.6);
      produkt1.Wyświetl();
      produkt2.Wyświetl();
      if (produkt1.Porównaj_produkty(produkt2))
          cout<<"Produkty maja rowne atrybuty"<<endl;
      else  cout<<"Produkty nie maja rownych atrybutow"<<endl;
    }
    cout<<"Tutaj juz nie ma obiektow"<<endl;
    //produkt1.Wyświetl();
    cin.get();
    return 0;
}

```

```

C:\Settings\dydaktyka\Programowanie_obiek...
Wywolany zwykly konstruktor z parametrami
Wywolany zwykly konstruktor z parametrami
Cena produktu: 3.5, Nazwa produktu: Zeszyt
Cena produktu: 1.6, Nazwa produktu: Atrament
Wywolany konstruktor kopiujacy
Wywolany destrutor
Produkty nie maja rownych atrybutow
Wywolany destrutor
Wywolany destrutor
Tutaj juz nie ma obiektow

```