

Budowa aplikacji z graficznym interfejsem użytkownika - GUI (Graphic User Interface)

- 1. Udostępnianie wszystkich prywatnych atrybutów do prezentacji, wprowadzenie standardu nazewnictwa plików – nazwy plików aplikacji poprzedzone literą T**
- 2. Budowa głównego formularza GUI**
- 3. Budowa okienek dialogowych do wprowadzania danych**
- 4. Budowa okienek dialogowych do wyświetlania danych**
- 5. Aplikacja do sporządzania rachunków - uzupełnienie**

Budowa aplikacji z graficznym interfejsem użytkownika - GUI (Graphic User Interface)

- 1. Udostępnianie wszystkich prywatnych atrybutów do prezentacji, wprowadzenie standardu nazewnictwa plików – nazwy plików aplikacji poprzedzone literą T**

```
#ifndef _Produkt1
#define _Produkt1
#include <iostream.h>
#include <string.h>
#include <iomanip.h>
#include <stdlib.h>
#include "TABSTRAKCYJNY.h"
class TProdukt1: public TAbstrakcyjny
{protected:
    string nazwa;
    float cena;
public:
    TProdukt1(string nazwa_="bez nazwy",float cena_=0);
    TProdukt1(TProdukt1&);
    ~TProdukt1();
    string Podaj_nazwe() {return nazwa;}//dodano po dodaniu GUI
    virtual float Podaj_cene();
    virtual float Podaj_podatek() { return -1; }
    virtual float Podaj_promocje() { return -1; }
    void operator+=(TAbstrakcyjny&){}
    int operator==(TAbstrakcyjny&);
    string toString();
    friend ostream& operator<<(ostream&, TProdukt1&);
};
#endif
```

```
#ifndef _Produkt2
#define _Produkt2
#include "TPRODUKT1.h"
class TProdukt2: virtual public TProdukt1
{
protected:
    float podatek;
public:
    TProdukt2(string nazwa_="bez nazwy",float cena_=0,
              float podatek=0);
    TProdukt2(TProdukt2&);
    ~TProdukt2();
    float Czesc_brutto();
    float Podaj_cene();
    float Podaj_podatek();
    string toString();
    friend ostream& operator<<(ostream&, TProdukt2&);
};
#endif
```



```
#ifndef _Produkt3
#define _Produkt3
#include "TPRODUKT1.h"
class TProdukt3: virtual public TProdukt1
{
protected:
    float promocja;
public:
    TProdukt3(string nazwa_="bez nazwy",float cena_=0,
              float promocja=0);
    TProdukt3(TProdukt3&);
    ~TProdukt3();
    float Czesc_brutto();
    float Podaj_cene();
    float Podaj_promocje();
    string toString();
    friend ostream& operator<<(ostream& wy, TProdukt3& p);
};
#endif
```



```
#ifndef _Produkt4
#define _Produkt4
#include "TPRODUKT2.h"
#include "TPRODUKT3.h"
class TProdukt4: public TProdukt3, public TProdukt2
{
    public:
        TProdukt4(string nazwa_="bez nazwy",float cena_=0,
                float podatek_=0, float promocja_=0);
        TProdukt4(TProdukt4&);
        ~TProdukt4();
        float Podaj_cene();
        float Czesc_brutto();
        string toString();
        friend ostream& operator<<(ostream&, TProdukt4&);
};
#endif
```



```
#ifndef _Zakup
#define _Zakup
#include "TPRODUKT1.h"
class TZakup: public TAbstrakcyjny
{
    protected:
        TProdukt1* produkt;
        float ilosc;
    public:
        TZakup(TProdukt1*produkt_=NULL,float ilosc_=0);
        TZakup(TZakup&);
        ~TZakup();
        float Podaj_wartosc();
        float Podaj_ilosc();
        TProdukt1* Podaj_produkt() { return produkt;} //dodano po dodaniu GUI
        void operator+=(TAbstrakcyjny&);
        int operator==(TAbstrakcyjny&);
        string toString();
        friend ostream& operator<<(ostream&, TZakup&);
};
#endif
```

```
#ifndef _Rachunek
#define _Rachunek
#include "TZAKUP.h"
#include "TKOL2.h"
class TRachunek:public TAbstrakcyjny
{protected:
    int numer;
    TKol2<TZakup> zakupy;
    void kopia(TRachunek& r);
public:
    TRachunek(int=0);
    TRachunek(TRachunek& r);
    ~TRachunek();
    TKol2<TZakup>& Podaj_zakupy() { return zakupy;} //dodano po dodaniu GUI
    int Podaj_numer() { return numer; } //dodano po dodaniu GUI
    void operator=(TRachunek&);
    float Podaj_wartosc();
    int Dodaj_zakup(TZakup*);
    void operator+=(TAbstrakcyjny&) {}
    int operator==(TAbstrakcyjny&);
    string toString();
    friend ostream& operator<<(ostream&, TRachunek&);
};
#endif
```



```
#ifndef _Aplikacja
#define _Aplikacja
#include <iostream.h>
#include <string.h>
#include <iomanip.h>
#include <stdlib.h>
#include "TPRODUKT4.h"
#include "TRACHUNEK.h"
class TAplikacja
{
    TKol2<TProdukt1> produkty;
    TKol2<TRachunek> rachunki;
    TProdukt1* Wykonaj_produkt(string* atrybuty);
public:
    ~TAplikacja()
    {
        produkty.Usun_kolekcje();
        rachunki.Usun_kolekcje();
    }
    int Wstaw_zakup(string* produkt,int ilosc, int numer);
    int Wstaw_produkt(string* wstawianyprodukt);
    int Wstaw_rachunek(int nr);
    TKol2<TProdukt1>& Podaj_produkty();
    TKol2<TRachunek>& Podaj_rachunki();
    TRachunek* Podaj_rachunek(int nr);
};
#endif
```

int Wstaw_zakup(string* produkt,int ilosc, int numer);
int Wstaw_produkt(string* wstawianyprodukt);
int Wstaw_rachunek(int nr);
TKol2<TProdukt1>& Podaj_produkty();
TKol2<TRachunek>& Podaj_rachunki();
TRachunek* Podaj_rachunek(int nr);

```
#include "T Aplikacja.h"
int T Aplikacja::Wstaw_zakup(string* produkt, int ilosc, int numer)
{
    int wynik;
    TRachunek* poszukiwanyrachunek=new TRachunek(numer);
    TProdukt1* poszukiwanyprodukt=Wykonaj_produkt(produkt);
    if (poszukiwanyrachunek!=NULL && poszukiwanyprodukt!=NULL)
    {
        TRachunek* znalezionyrachunek;
        znalezionyrachunek=rachunki.Podaj(poszukiwanyrachunek);
        if (znalezionyrachunek!=NULL)
        {
            TProdukt1* znalezionyprodukt=
                produkty.Podaj(poszukiwanyprodukt);
            if (znalezionyprodukt!=NULL)
                wynik=
                    znalezionyrachunek->Dodaj_zakup(new TZakup(znalezionyprodukt,ilosc));
            else
                wynik=5;
        }
        else
            wynik=4;
    }
    else
        wynik=3;
    delete poszukiwanyrachunek;
    delete poszukiwanyprodukt;
    return wynik;
}
```

```
int TAplikacja::Wstaw_produkt(string* wstawianyprodukt)
{
    TProdukt1* pom= Wykonaj_produkt(wstawianyprodukt);
    int wynik=3;
    if (pom!=NULL)
        wynik=produkty.Wstaw(pom);
    return wynik;
}

int TAplikacja::Wstaw_rachunek(int nr)
{
    TRachunek* r=new TRachunek(nr);
    int wynik=3;
    if(r!=NULL)
        wynik=rachunki.Wstaw(r);
    return wynik;
}

TKol2<TProdukt1>& TAplikacja::Podaj_produkty()
{
    return produkty;
}

TKol2<TRachunek>& TAplikacja::Podaj_rachunki()
{
    return rachunki;
}

TRachunek* TAplikacja::Podaj_rachunek(int nr)
{
    TRachunek r(nr);
    return rachunki.Podaj(&r);
}
```

```
TProdukt1* TApplikacja::Wykonaj_produkt(string* atrybuty)
{
    TProdukt1* produkt=NULL;
    if (atrybuty[0]=="1")
        produkt=new TProdukt1(atrybuty[1], atof(atrybuty[2].c_str()));
    else
        if (atrybuty[0]=="2")
            produkt=new TProdukt2(atrybuty[1], atof(atrybuty[2].c_str()),
                                   atof(atrybuty[3].c_str()));
        else
            if (atrybuty[0]=="3")
                produkt=new TProdukt3(atrybuty[1], atof(atrybuty[2].c_str()),
                                       atof(atrybuty[3].c_str()));
            else
                if (atrybuty[0]=="4")
                    produkt=new TProdukt4(atrybuty[1], atof(atrybuty[2].c_str()),
                                           atof(atrybuty[3].c_str()),
                                           atof(atrybuty[4].c_str()));

    return produkt;
}
```

Budowa aplikacji z graficznym interfejsem użytkownika - GUI (Graphic User Interface)

- 1. Udostępnianie wszystkich prywatnych atrybutów do prezentacji, wprowadzenie standardu nazewnictwa plików – nazwy plików aplikacji poprzedzone literą T**
- 2. Budowa głównego formularza GUI**

Gotowa aplikacja

The screenshot displays the C++Builder 6 IDE interface. At the top, the title bar reads "C++Builder 6 - Rachunki [Running] [Built: 9.47 secs]". The menu bar includes "File", "Edit", "Search", "View", "Project", "Run", "Component", "Tools", "Window", and "Help". The toolbar contains various icons for file operations and development tools. A red-bordered box highlights the text "Gotowa aplikacja" in the upper right corner.

The main workspace is divided into several panes. On the left, a "Konspekt" (Outline) pane shows a list of slides numbered 6 through 10. The central pane displays a code editor for "T Aplikacja.cpp" with the following visible code:

```
#include <...>
int T...
( int...
  TR...
  TP...
  if...
  {
    ...
  }
  else
    wynik=4;
```

Overlaid on the code editor is a window titled "Sporządzanie rachunkow" with a menu bar containing "Pliki", "Dane", "Wyswietl", and "O programie". The "Wyswietl" menu is open, showing two options: "Wyswietl produkty" and "Wyswietl rachunki".

At the bottom of the IDE, a "Build" pane shows two warning messages:

```
[C++ Warning] produkt1.h(13): W8058 Cannot create pre-compiled header: initialized data in header
[C++ Warning] produkt1.h(13): W8058 Cannot create pre-compiled header: initialized data in header
```

The status bar at the bottom indicates "Slajd 14 z 15", "Projekt domyślny", "Polski", and the system clock shows "14:44".

Zakładanie projektu

The screenshot shows the C++Builder 6 IDE. The 'New' menu is open, and 'Application' is selected. The code editor displays the following C++ code:

```
int TApplikacja::Wstaw_zakup(string* produkt, int ilosc, int numer)
{
    int wynik;
    TRachunek* poszukiwanyrachunek=new TRachunek(numer);
    TProdukt1* poszukiwanyprodukt=Wykonaj_produkt(produkt);
    if (poszukiwanyrachunek!=NULL && poszukiwanyprodukt!=NULL)
    {
        TRachunek* znalezionyrachunek;
        znalezionyrachunek=rachunki.Podaj(poszukiwanyrachunek);
        if (znalezionyrachunek!=NULL)
        {
            TProdukt1* znalezionyprodukt=
                produkty.Podaj(poszukiwanyprodukt);
            if (znalezionyprodukt!=NULL)
                wynik=
                    znalezionyrachunek->Dodaj_zakup(new TZakup(znalezionyprodukt, ilosc, numer));
        }
        else
            wynik=5;
    }
    else
        wynik=4;
}
```

The Object Inspector shows the properties of the selected component, OKBottomDlg2. The 'Caption' property is set to 'swietl produkty'. The 'Color' property is set to 'clBtnFace'. The 'Constraints' property is set to '(TSizeConstrain)'. The 'Build' window shows two warnings:

```
[C++ Warning] produkt1.h(13): W8058 Cannot create pre-compiled header: initialized data in header
[C++ Warning] produkt1.h(13): W8058 Cannot create pre-compiled header: initialized data in header
```

The status bar at the bottom shows 'Projekt domyślny' and 'Polski'.

Widok projektu z formularzem

The screenshot displays the C++Builder 6 IDE interface. The main workspace shows a grid-based form design area labeled 'Form1'. To the left, the 'Object TreeView' shows a tree structure with 'Form1' selected. Below it, the 'Object Inspector' displays the properties of the selected form, such as 'Caption', 'ClientHeight', and 'ClientWidth'. The top menu bar includes 'File', 'Edit', 'Search', 'View', 'Project', 'Run', 'Component', 'Tools', 'Window', and 'Help'. A red box highlights the 'Form' icon in the toolbar, with red arrows pointing to the 'Object TreeView' and the 'Form1' design area. A text box on the right contains the text 'Widok projektu z formularzem'. Another text box on the left contains the text 'Widok formularzy' and 'Widok modułów'. The bottom status bar shows 'Projekt domyślny' and 'Polski'.

Widok formularzy

Widok modułów

z 5 plików wybranych 0 z 32 998 k w 0 z 56 plików wybranych

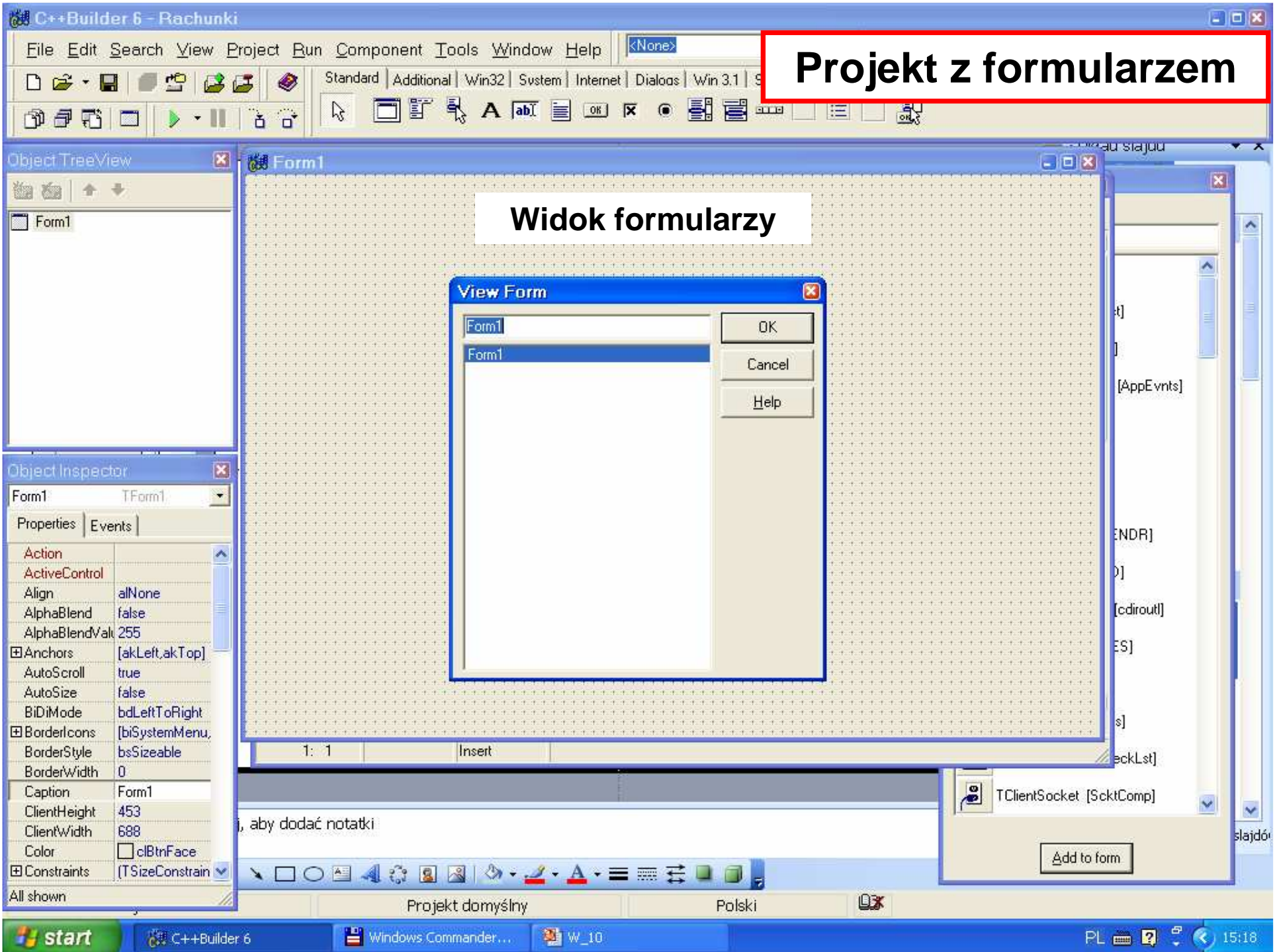
tyka\Programowanie_obiektowe>

gd F4 Edycja F5 Kopiowanie F6 ZmPrzes F7 UtwKat F8 Usuń

TClientSocket [ScktComp]

Add to form

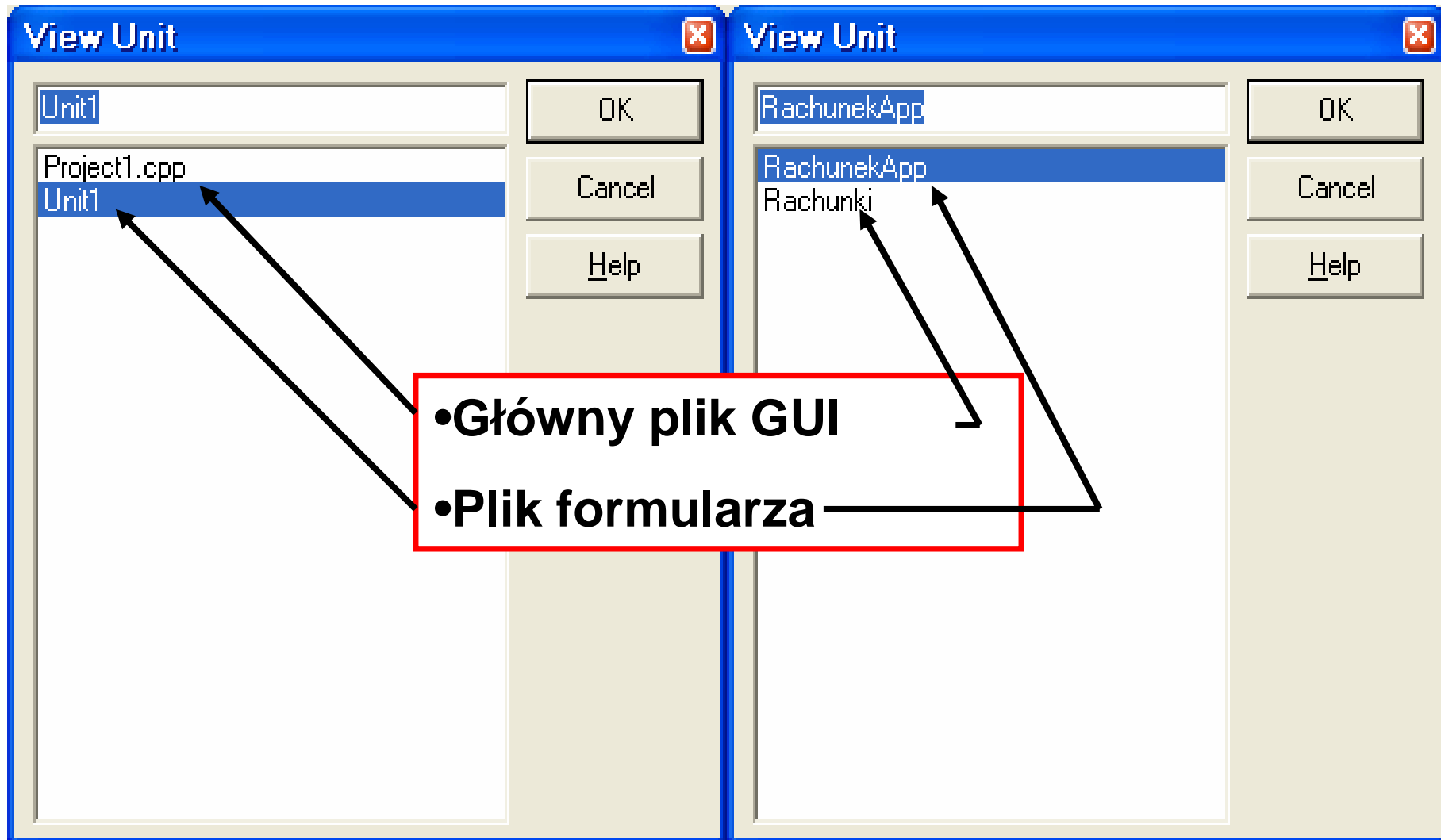
Projekt domyślny Polski

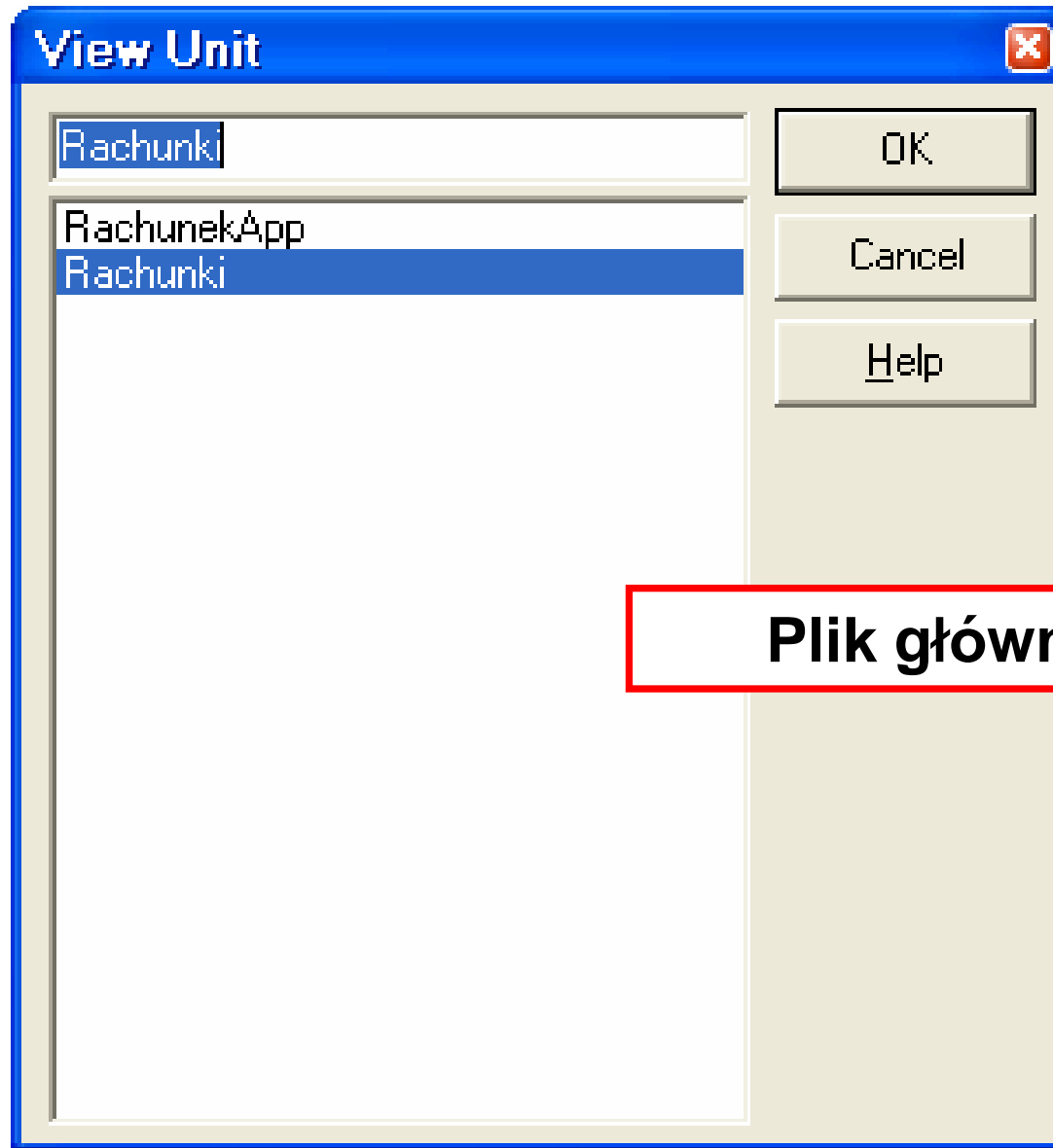


Widok modułów

Przed **Save Project As**

Po **Save Project As**





Plik główny GUI

```
Rachunki.cpp
RachunekApp.h  Rachunki.cpp
//-----
#include <vc1.h>
#pragma hdrstop
//-----
USEFORM("RachunekApp.cpp", Form1);
//-----
WINAPI WinMain(HINSTANCE, HINSTANCE, LPST
{
    try
    {
        Application->Initialize();
        Application->CreateForm(__classid(TForm1), &Form1);
        Application->Run();
    }
    catch (Exception &exception)
    {
        Application->ShowException(&exception);
    }
    catch (...)
    {
        try
        {
            throw Exception("");
        }
        catch (Exception &exception)
        {
            Application->ShowException(&exception);
        }
    }
    return 0;
}
//-----
```

**Plik główny GUI z
automatycznie dołączonym
obiektem formularza
głównego typu **TForm1**,
zdefiniowanym w pliku
RachunekApp**

Plik formularza – plik nagłówkowy

```
//-----  
  
#ifndef RachunekAppH  
#define RachunekAppH  
  
//-----  
#include <Classes.hpp>  
#include <Controls.hpp>  
#include <StdCtrls.hpp>  
#include <Forms.hpp>  
  
//-----  
class TForm1 : public TForm  
{  
    __published:      // IDE-managed Components  
private:             // User declarations  
public:              // User declarations  
    __fastcall TForm1(TComponent* Owner);  
};  
  
//-----  
extern PACKAGE TForm1 *Form1;  
  
//-----  
#endif
```

Plik formularza – plik
modułowy

```
//-----  
  
#include <vcl.h>  
#pragma hdrstop  
  
#include "RachunekApp.h"  
//-----  
#pragma package(smart_init)  
#pragma resource "*.dfm"  
TForm1 *Form1;  
//-----  
__fastcall TForm1::TForm1(TComponent* Owner)  
    : TForm(Owner)  
{  
}  
//-----
```


The image displays two side-by-side screenshots of the Delphi Components palette. The left window, titled 'Components', has a search bar containing 'TCheckBox [CheckLst]'. The list of components includes 'TButton [StdCtrls]', which is highlighted with a red rectangular box and followed by the number '(1)'. The right window, also titled 'Components', has a search bar containing 'TChartfx [Chartfx3]'. The list of components includes 'TCheckBox [StdCtrls]', which is highlighted with a red rectangular box and followed by the number '(1)'. Both windows feature an 'Add to form' button at the bottom center.

**Wybrane
komponenty w
aplikacji
(1) przycisk**

Components Search by name: TDateTimePicker [ComCtrls]

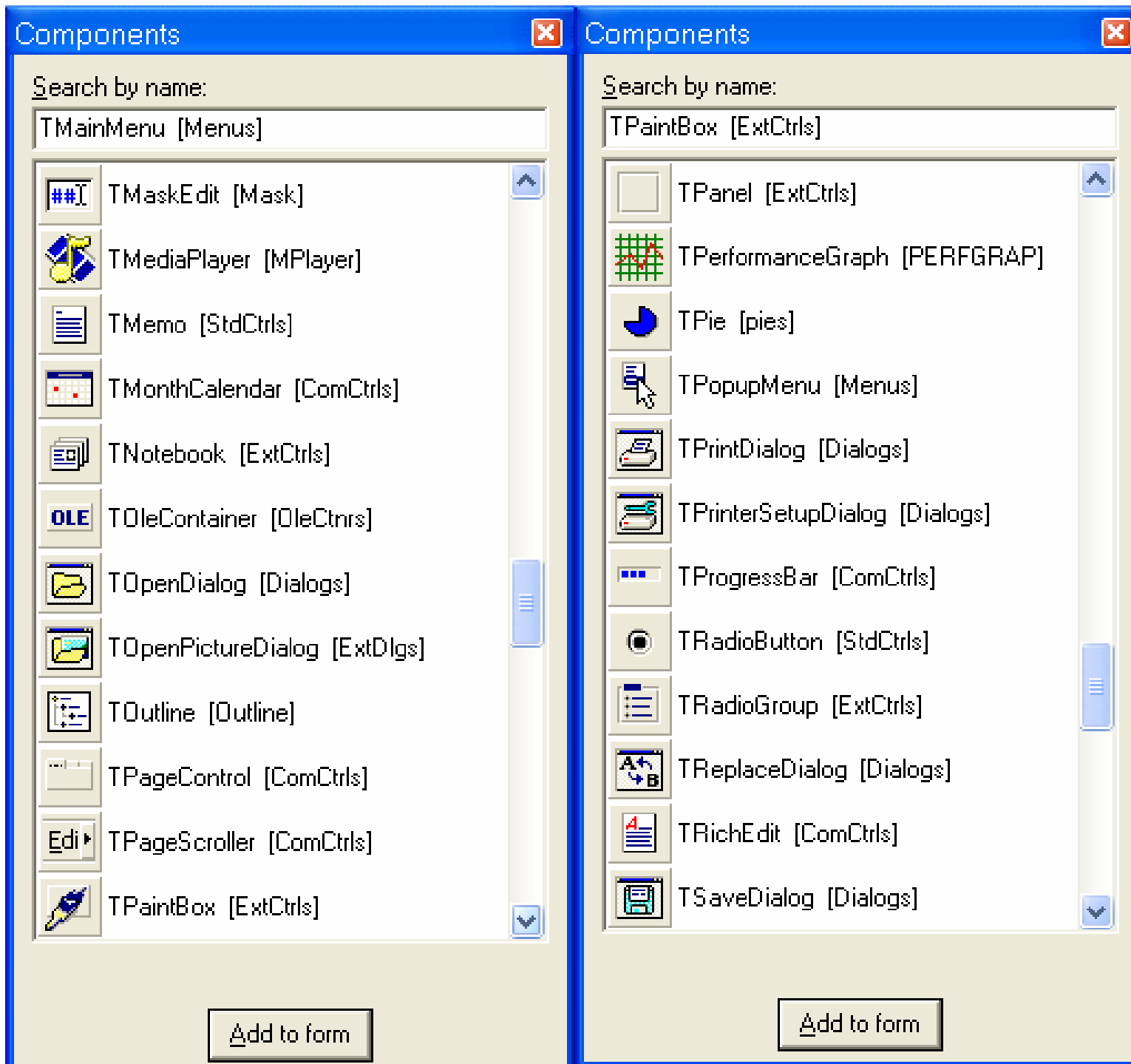
- TDdeClientConv [DdeMan]
- TDdeClientItem [DdeMan]
- TDdeServerConv [DdeMan]
- TDdeServerItem [DdeMan]
- TDirectoryListBox [FileCtrl]
- TDrawGrid [Grids]
- TDriveComboBox [FileCtrl]
- TEdit [StdCtrls] (2)**
- TF1Book [VCF1]
- TFileListBox [FileCtrl]
- TFilterComboBox [FileCtrl]
- TFindDialog [Dialogs]

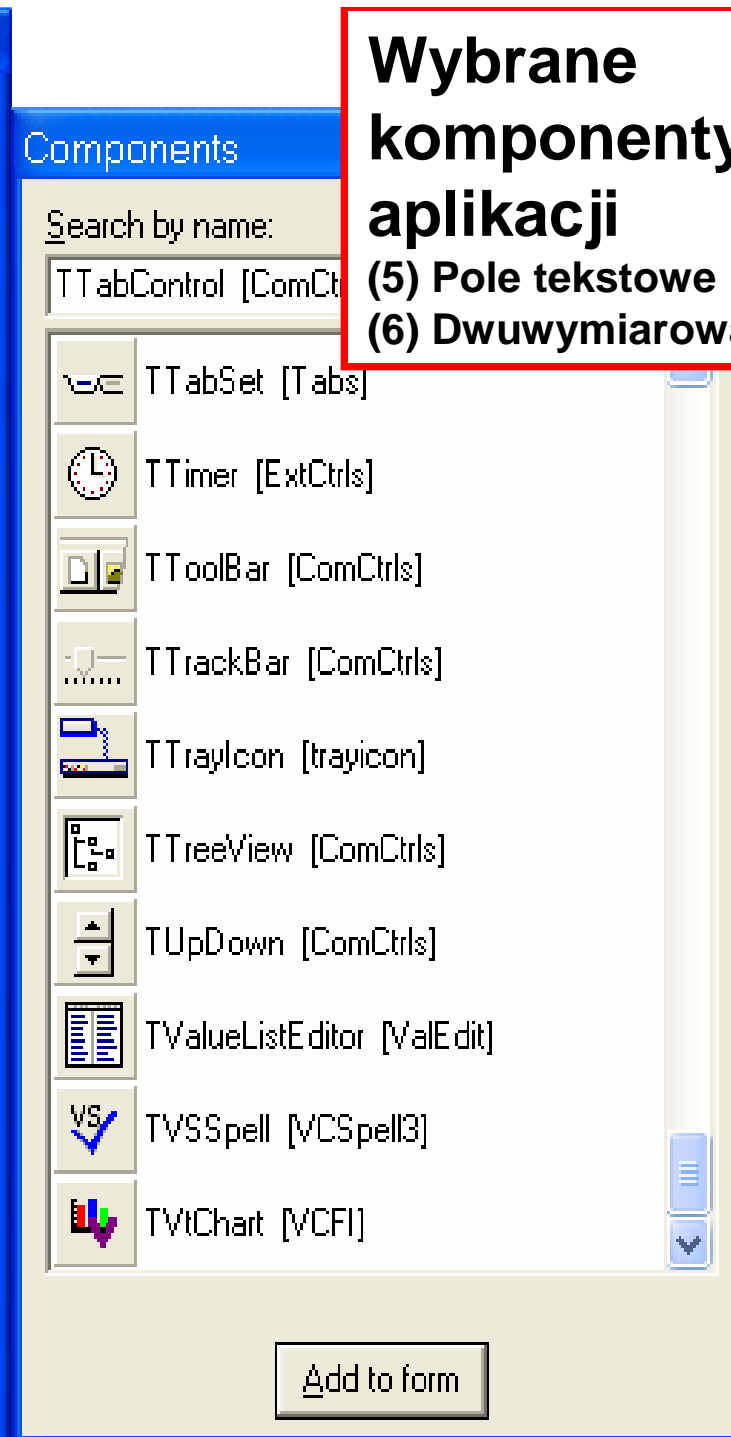
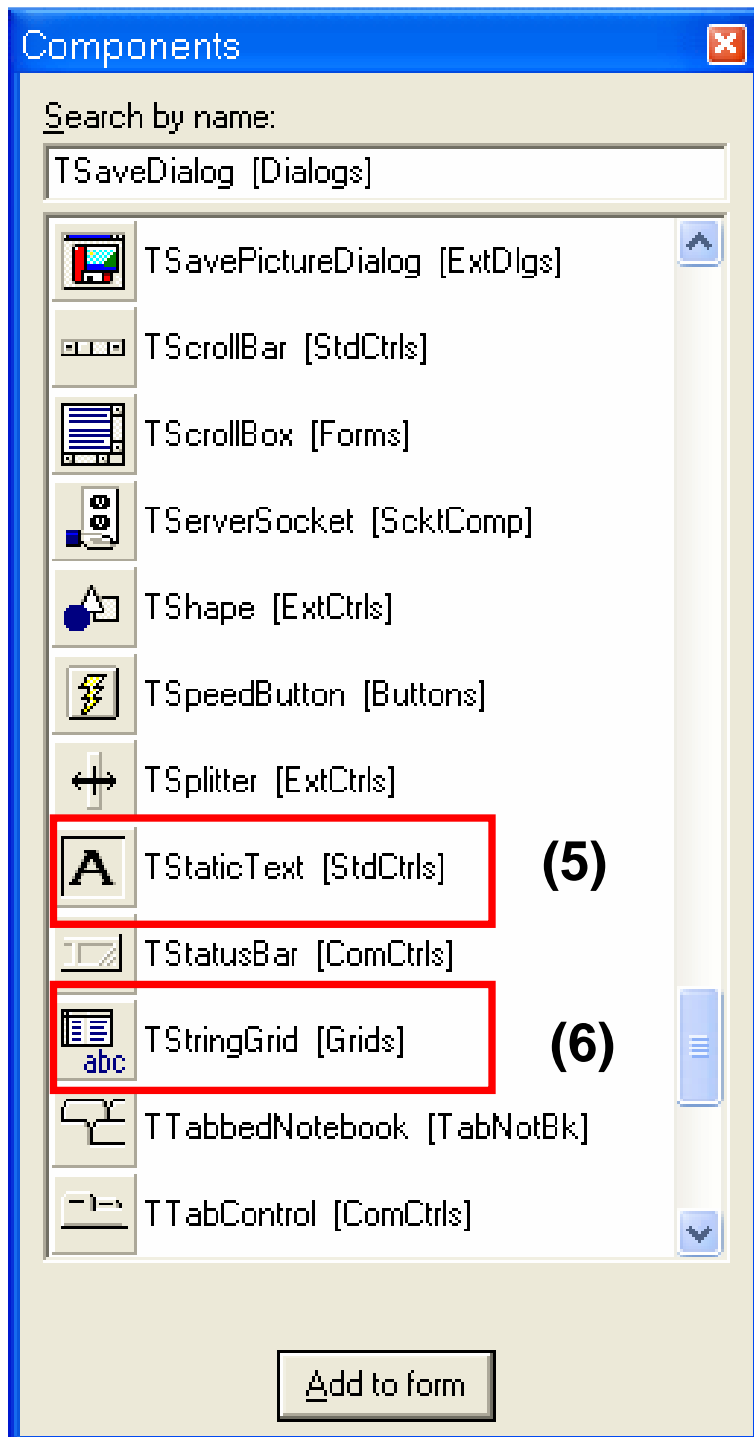
Components Search by name: TFindDialog [Dialogs]

- TFontDialog [Dialogs]
- TGroupBox [StdCtrls]
- THeader [ExtCtrls]
- THeaderControl [ComCtrls]
- THotKey [ComCtrls]
- TImage [ExtCtrls]
- TImageList [Controls]
- TLabel [StdCtrls] (3)**
- TLabelEdit [ExtCtrls]
- TListBox [StdCtrls]
- TListView [ComCtrls]
- TMainMenu [Menus] (4)**

Add to form

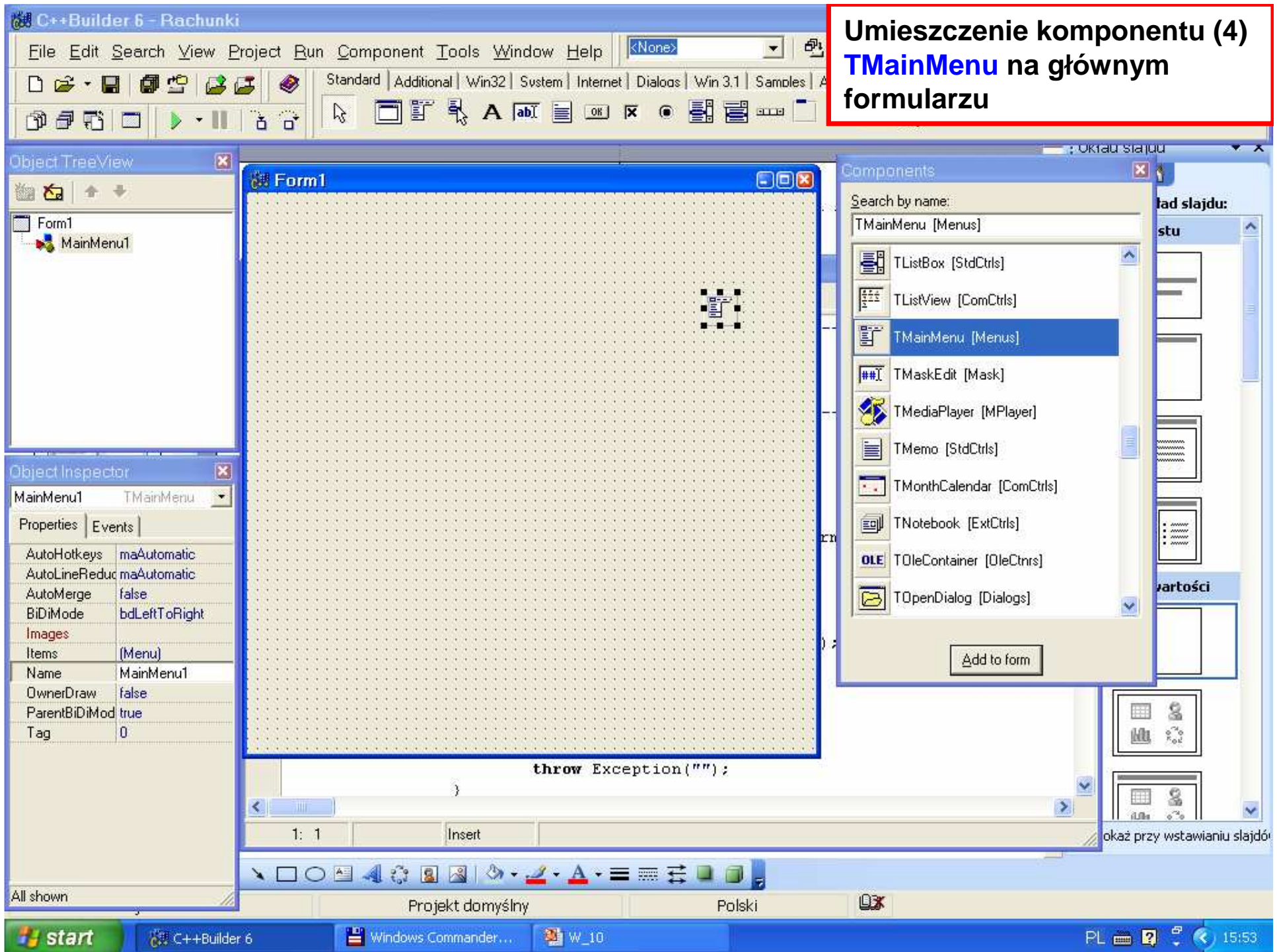
Wybrane komponenty w aplikacji
(2) Pole do wprowadzania danych
(3) Etykieta
(4) Menu głównego formularza



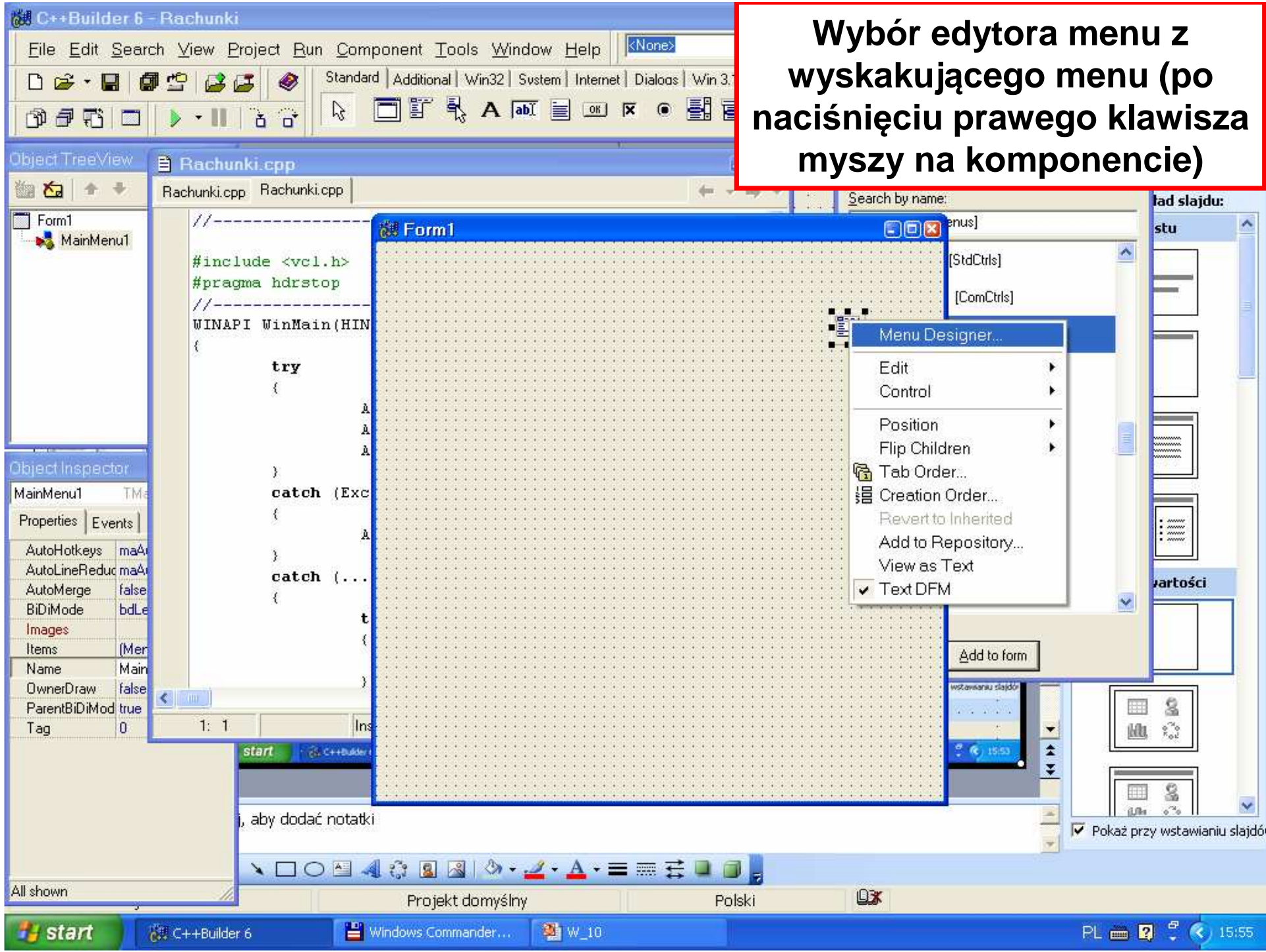


Wybrane komponenty w aplikacji
(5) Pole tekstowe
(6) Dwuwymiarowa tablica

**Umieszczenie komponentu (4)
TMainMenu na głównym
formularzu**



Wybór edytora menu z wyskakującego menu (po naciśnięciu prawego klawisza myszy na komponencie)



Wstawianie elementów menu głównego – tworzenie listy typu Menu Bar

The screenshot displays the C++Builder 6 IDE interface. The main window shows a form titled 'Form1' with a menu bar 'Form1->MainMenu1' being edited. A context menu is open over the menu bar, showing options like 'Insert', 'Delete', 'Create Submenu', 'Select Menu...', 'Save As Template...', 'Insert From Template...', 'Delete Templates...', and 'Insert From Resource...'. The 'Insert' option is selected. The 'Components' palette on the right shows various components, with 'Menu' selected. The 'Object Inspector' on the left shows the properties of the selected component. The 'Object TreeView' on the top left shows the project structure. The 'Code Editor' in the center shows the source code for 'Rachunki.cpp'. The 'Object Inspector' shows the following properties:

Property	Value
Action	
AutoCheck	false
AutoHotkeys	maP
AutoLineReduction	maP
Bitmap	(None)
Break	mbN
Caption	
Checked	false
Default	false
Enabled	true
GroupIndex	0
HelpContext	0
Hint	
ImageIndex	-1
Name	
RadioItem	false
ShortCut	(None)

The 'Code Editor' shows the following code:

```
//-----  
#include <vcl.h>  
#pragma hdrstop  
//-----  
WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)  
{  
    try  
    {  
        Application->CreateForm(TForm, &Form1);  
        Application->Form1->MainMenu1->Create();  
    }  
    catch (Exception &E)  
    {  
        ShowMessage(E.Message);  
    }  
    catch (...)  
    {  
        ShowMessage("Unknown error");  
    }  
    Application->Run();  
}
```

Wstawianie elementów menu głównego – tworzenie elementu „Pliki” w liście typu Menu Bar

The screenshot shows the C++Builder 6 IDE interface. The main window displays the 'Form1' design view, with a sub-window 'Form1->MainMenu1' open. The 'Object Inspector' is visible, showing the properties of the selected menu item. The 'Caption' property is set to 'Pliki'. The 'Component Trays' on the right show the 'Menu' component selected. The 'Object TreeView' on the left shows the project structure with 'Form1' and 'MainMenu1'.

Object Inspector Properties:

Property	Value
Action	
AutoCheck	false
AutoHotkeys	maParent
AutoLineReduction	maParent
Bitmap	(None)
Break	mbNone
Caption	Pliki
Checked	false
Default	false
Enabled	true
GroupIndex	0
HelpContext	0
Hint	
ImageIndex	-1
Name	
RadioItem	false
Shortcut	(None)
All shown	

Component Trays:

- Menu [Menus]
- Mask [Mask]
- MPlayer [MPlayer]
- StdCtrls
- Calendar [ComCtrls]
- ExtCtrls
- DlgCtrls
- Dialogs [Dialogs]

Object TreeView:

- Form1
 - MainMenu1

**Wstawianie listy rozwijanej
do elementu „Pliki” z listy
Menu Bar – znak & pozwala
wyróżnić wybraną literę w
pozycji listy**

The screenshot displays the C++Builder 6 IDE interface. The main window shows a form titled 'Form1' with a menu bar containing a single item 'Pliki'. The 'Object Inspector' is open, showing the properties of the selected 'Pliki' menu item. The 'Caption' property is set to 'Dtworz &plik', where the ampersand (&) is used to highlight the letter 'P' in the menu. The 'Object TreeView' on the left shows the project structure, including 'Form1', 'MainMenu1', and 'Pliki (Pliki)'. The 'Component Palette' on the right shows various controls, including 'Menu [Menus]'. The status bar at the bottom indicates 'Slajd 28 z 29' and 'Projekt domyślny'.

Property	Value
Action	
AutoCheck	false
AutoHotkeys	maParent
AutoLineReduction	maParent
Bitmap	(None)
Break	mbNone
Caption	Dtworz &plik
Checked	false
Default	false
Enabled	true
GroupIndex	0
HelpContext	0
Hint	
ImageIndex	-1
Name	
RadioItem	false
Shortcut	(None)
All shown	

Wstawianie listy rozwijanej do elementu „Pliki” z listy Menu Bar – znak – oznacza dodanie poziomej linii do listy

The screenshot shows the C++Builder 6 IDE interface. The main window displays a form with a menu bar containing 'Pliki' and 'Zapisz plik'. The 'Object Inspector' is open, showing the properties of a selected 'TMenuItem'. The 'Component Palette' is also open, showing various components like [Menus], [Mask], [MPlayer], etc. The 'Object TreeView' shows the project structure, including 'Rachunki.cpp' and 'Form1'. The 'Form1->MainMenu1' window is open, showing the menu items. The 'Object Inspector' shows the following properties for the selected 'TMenuItem':

Property	Value
Action	
AutoCheck	false
AutoHotkeys	maParent
AutoLineReduction	maParent
Bitmap	(None)
Break	mbNone
Caption	
Checked	false
Default	false
Enabled	true
GroupIndex	0
HelpContext	0
Hint	
ImageIndex	-1
Name	
RadioItem	false
Shortcut	(None)

The 'Component Palette' shows the following components:

- [StdCtrls]
- [ComCtrls]
- [Menus]
- [Mask]
- [MPlayer]
- [StdCtrls]
- [Calendar [ComCtrls]
- [ExtCtrls]
- [DlgCtrls]
- [Dialogs]

The 'Object TreeView' shows the project structure:

- Rachunki.cpp
- Form1
 - MainMenu1
 - Pliki {Plik1}
 - Otworz plik
 - Zapisz plik

Wstawianie listy rozwijanej do elementu „Pliki” z listy Menu Bar

The screenshot displays the C++Builder 6 IDE interface. The main window shows a form titled 'Form1' with a menu bar containing a 'Pliki' menu item. A context menu is open over the 'Pliki' item, showing 'Otworz plik' and 'Zapisz plik'. The 'Object Inspector' is open, showing the properties of the selected 'TMenuItem' object. The 'Properties' tab is active, and the 'Caption' property is set to 'Koniec'. The 'Object TreeView' on the left shows the project structure, including 'MainMenu1' and 'Pliki (Plik1)'. The 'Components' palette on the right is open, showing various components like 'Menu', 'MenuItem', 'MenuItemSeparator', etc. The 'Object TreeView' shows the following structure:

```
form1
├── MainMenu1
│   ├── Pliki (Plik1)
│   │   ├── Otworz &plik
│   │   └── &Zapisz plik
│   └── ...
└── ...
```

The 'Object Inspector' shows the following properties for the selected 'TMenuItem' object:

Property	Value
Action	
AutoCheck	false
AutoHotkeys	maParent
AutoLineReduction	maParent
Bitmap	(None)
Break	mbNone
Caption	Koniec
Checked	false
Default	false
Enabled	true
GroupIndex	0
HelpContext	0
Hint	
ImageIndex	-1
Name	
RadioItem	false
Shortcut	(None)
All shown	

The 'Components' palette shows the following components:

- Menu
- MenuItem
- MenuItemSeparator
- MenuItemImage
- MenuItemImageList
- MenuItemImageList2
- MenuItemImageList3
- MenuItemImageList4
- MenuItemImageList5
- MenuItemImageList6
- MenuItemImageList7
- MenuItemImageList8
- MenuItemImageList9
- MenuItemImageList10
- MenuItemImageList11
- MenuItemImageList12
- MenuItemImageList13
- MenuItemImageList14
- MenuItemImageList15
- MenuItemImageList16
- MenuItemImageList17
- MenuItemImageList18
- MenuItemImageList19
- MenuItemImageList20
- MenuItemImageList21
- MenuItemImageList22
- MenuItemImageList23
- MenuItemImageList24
- MenuItemImageList25
- MenuItemImageList26
- MenuItemImageList27
- MenuItemImageList28
- MenuItemImageList29
- MenuItemImageList30
- MenuItemImageList31
- MenuItemImageList32
- MenuItemImageList33
- MenuItemImageList34
- MenuItemImageList35
- MenuItemImageList36
- MenuItemImageList37
- MenuItemImageList38
- MenuItemImageList39
- MenuItemImageList40
- MenuItemImageList41
- MenuItemImageList42
- MenuItemImageList43
- MenuItemImageList44
- MenuItemImageList45
- MenuItemImageList46
- MenuItemImageList47
- MenuItemImageList48
- MenuItemImageList49
- MenuItemImageList50
- MenuItemImageList51
- MenuItemImageList52
- MenuItemImageList53
- MenuItemImageList54
- MenuItemImageList55
- MenuItemImageList56
- MenuItemImageList57
- MenuItemImageList58
- MenuItemImageList59
- MenuItemImageList60
- MenuItemImageList61
- MenuItemImageList62
- MenuItemImageList63
- MenuItemImageList64
- MenuItemImageList65
- MenuItemImageList66
- MenuItemImageList67
- MenuItemImageList68
- MenuItemImageList69
- MenuItemImageList70
- MenuItemImageList71
- MenuItemImageList72
- MenuItemImageList73
- MenuItemImageList74
- MenuItemImageList75
- MenuItemImageList76
- MenuItemImageList77
- MenuItemImageList78
- MenuItemImageList79
- MenuItemImageList80
- MenuItemImageList81
- MenuItemImageList82
- MenuItemImageList83
- MenuItemImageList84
- MenuItemImageList85
- MenuItemImageList86
- MenuItemImageList87
- MenuItemImageList88
- MenuItemImageList89
- MenuItemImageList90
- MenuItemImageList91
- MenuItemImageList92
- MenuItemImageList93
- MenuItemImageList94
- MenuItemImageList95
- MenuItemImageList96
- MenuItemImageList97
- MenuItemImageList98
- MenuItemImageList99
- MenuItemImageList100

Wstawiona lista rozwijana do elementu „Pliki” z listy Menu Bar

The screenshot displays the C++Builder 6 IDE interface. The main window shows a form titled "Sporządzanie rachunków" with a menu bar containing "Pliki", "Wstawianie", "Formatowanie", "Wstawianie slajdów", and "Wstawianie wartości". The "Pliki" menu is expanded, showing a dropdown list with three items: "Otworz plik", "Zapisz plik", and "Koniec". The "Object Inspector" is open on the left, showing the properties of the form. The "Object TreeView" on the far left shows the project structure, including "Form1" and "MainMenu1". The "Components" palette on the right is also visible, showing various standard components like "StdCtrls" and "ComCtrls". The status bar at the bottom indicates "Slajd 31 z 32", "Projekt domyślny", and "Polski".

```
RachunekApp.cpp
RachunekApp.h | Rachunki.cpp |
//-----
#ifndef RachunekAppH
#define RachunekAppH
//-----
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
//-----
class TForm1 : public TForm
{
    __published:      // IDE-managed Components
        TMainMenu *MainMenu1;
        TMenuItem *Plik1;
        TMenuItem *Otworzplik1;
        TMenuItem *Zapiszplik1;
        TMenuItem *N1;
        TMenuItem *Koniec1;
private:             // User declarations
public:              // User declarations
        __fastcall TForm1(TComponent* Owner);
};
//-----
extern PACKAGE TForm1 *Form1;
//-----
#endif
```

Atrybuty komponentu **TMainMenu** wstawione automatycznie do pliku nagłówkowego formularza głównego



Sporządzanie rachunków



Pliki Dane Wyświetl O programie

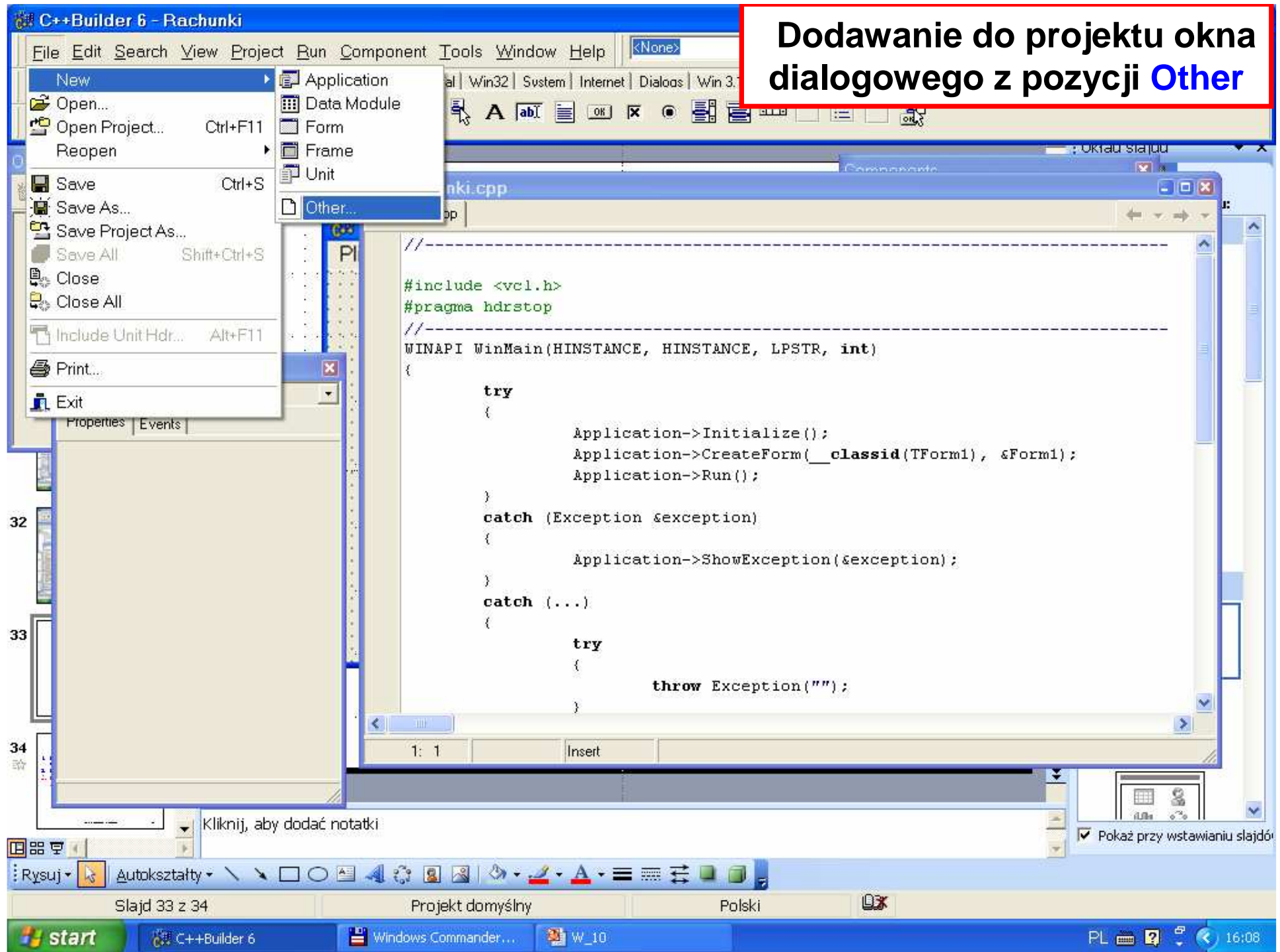


Gotowe Menu aplikacji

Budowa aplikacji z graficznym interfejsem użytkownika - GUI (Graphic User Interface)

- 1. Udostępnianie wszystkich prywatnych atrybutów do prezentacji, wprowadzenie standardu nazewnictwa plików – nazwy plików aplikacji poprzedzone literą T**
- 2. Budowa głównego formularza GUI**
- 3. Budowa okienek dialogowych do wprowadzania danych**

Dodawanie do projektu okna dialogowego z pozycji **Other**

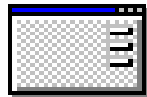


New Items

New | Rachunki | Forms | Dialogs | Projects



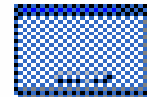
Dialog with
Help (Ho...



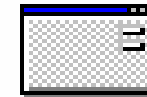
Dialog with
Help (Vertical)



Password
Dialog



Standard
Dialog
(Horizontal)



Standard
Dialog ...

**Dodawanie do projektu
standardowego okna
dialogowego **Standard
Dialogs** z zakładki **Dialogs****

Copy Inherit Use

OK

Cancel

Help

Dodawanie do projektu okna dialogowego **OKBottomDlg** reprezentowanego przez plik **Unit1.cpp**

The screenshot displays the C++Builder 6 IDE interface. The main window shows a form with a grid and a dialog box titled "Dodawanie produktów" (Adding products) with "OK" and "Cancel" buttons. The Object TreeView on the left shows a tree structure for "OKBottomDlg" containing "Bevel1", "CancelBtn", and "OKBtn". The Object Inspector on the bottom left shows the properties for "TOKBottomDlg", including "Caption" set to "Dodawanie produktów". The code editor in the foreground shows the implementation of the dialog class in "Unit1.cpp":

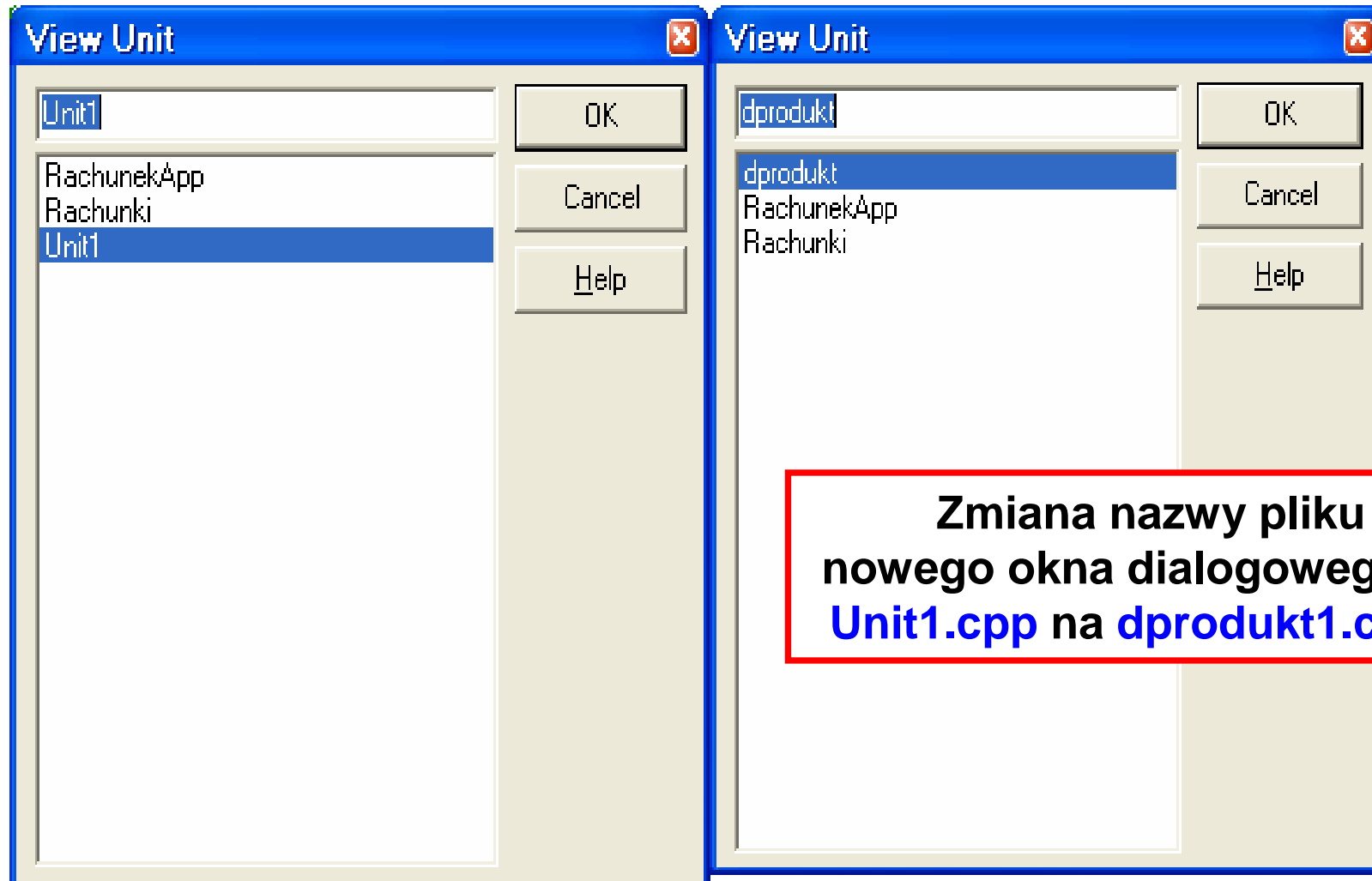
```
//-----  
#include <vcl.h>  
#pragma hdrstop  
  
#include "Unit1.h"  
//-----  
#pragma resource "*.dfm"  
TOKBottomDlg *OKBottomDlg;  
//-----  
__fastcall TOKBottomDlg::TOKBottomDlg(TComponent* AOwner)  
    : TForm(AOwner)  
{  
}  
//-----
```

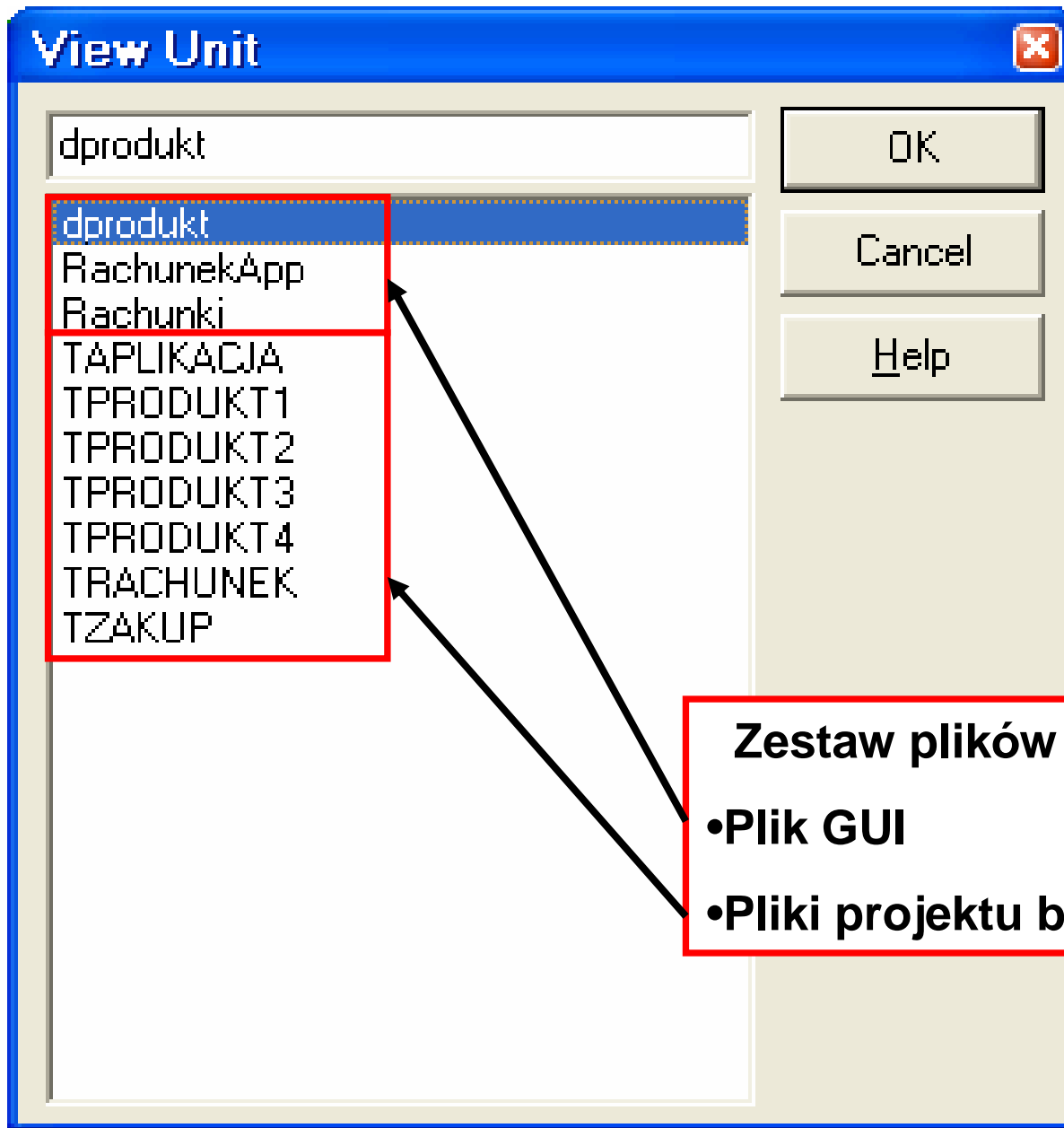
The status bar at the bottom indicates "Slajd 37 z 37" (Slide 37 of 37), "Projekt domyślny" (Default project), and "Polski" (Polish). The Windows taskbar at the very bottom shows the Start button, C++Builder 6, Windows Commander, and W_10, with the system clock at 16:13.

Widok modułów

Przed Save As

Po Save As





C++Builder 6 - Rachunki

File Edit Search View Project Run Component Tools Window Help

Standard Additional Win32 System Internet Dialogs Win 3.1 Samples ActiveX

Dodawanie produktow

dprodukt.cpp

```

//
#ifdef dproduktH
#define dproduktH
//
#include <vc1\System.hpp>
#include <vc1\Windows.hpp>
#include <vc1\SysUtils.hpp>
#include <vc1\Classes.hpp>
#include <vc1\Graphics.hpp>
#include <vc1\StdCtrls.hpp>
#include <vc1\Forms.hpp>
#include <vc1\Controls.hpp>
#include <vc1\Buttons.hpp>
#include <vc1\ExtCtrls.hpp>
//
class TOKBottomDlg : public TForm
{
public:
    TButton *OKBtn;
    TButton *CancelBtn;
    TBevel *Bevel1;
private:
public:
    virtual __fastcall TOKBottomDlg(TComponent* AOwner);
};
extern PACKAGE TOKBottomDlg *OKBottomDlg;
#endif

```

Object Inspector

OKBottomDlg TOKBottomDlg	
Properties	
Action	
ActiveControl	
Align	alNone
AlphaBlend	false
AlphaBlendVal	255
⊕ Anchors	[akLeft,akTop]
AutoScroll	false
AutoSize	false
BiDiMode	bdLeftToRight
⊕ BorderIcons	[biSystemMenu,biMinimize]
BorderStyle	bsDialog
BorderWidth	0
Caption	Dodawanie produktow
ClientHeight	204
ClientWidth	313
Color	<input type="checkbox"/> clBtnFace
⊕ Constraints	(TSizeConstraints)
All shown	

Podstawowe elementy standardowego okna dialogowego

notatki

Slajd 38 z 38 Projekt domyślny Polski

start C++Builder 6 Windows Commander... W_10 PL 16:18

```
dprodukt.cpp
Rachunki.cpp dprodukt.cpp

//-----
#include <vcl.h>
#pragma hdrstop

#include "dprodukt.h"
//-----
#pragma resource "*.dfm"
TOKBottomDlg *OKBottomDlg;
//-----
__fastcall TOKBottomDlg::TOKBottomDlg(TComponent* AOwner)
    : TForm(AOwner)
{
}
//-----
```

3: 16 Insert

Utworzony automatycznie plik modułowy **dprodukt1.cpp** okna dialogowego – po zmianie nazwy

The image shows a screenshot of a C++ IDE window titled "dprodukt.cpp". The window has a tab bar with "RachunekApp.h", "dprodukt.h", and "Rachunki.cpp". The main editor area displays the following code:

```
//-----  
#ifndef dproduktH  
#define dproduktH  
//-----  
#include <vcl\System.hpp>  
#include <vcl\Windows.hpp>  
#include <vcl\SysUtils.hpp>  
#include <vcl\Classes.hpp>  
#include <vcl\Graphics.hpp>  
#include <vcl\StdCtrls.hpp>  
#include <vcl\Forms.hpp>  
#include <vcl\Controls.hpp>  
#include <vcl\Buttons.hpp>  
#include <vcl\ExtCtrls.hpp>  
//-----  
class TOKBottomDlg : public TForm  
{  
    __published:  
        TButton *OKBtn;  
        TButton *CancelBtn;  
        TBevel *Bevel1;  
private:  
public:  
        virtual __fastcall TOKBottomDlg(TComponent* AOwner);  
};  
//-----  
extern PACKAGE TOKBottomDlg *OKBottomDlg;  
//-----  
#endif
```

A red-bordered callout box on the right side of the window contains the following text:

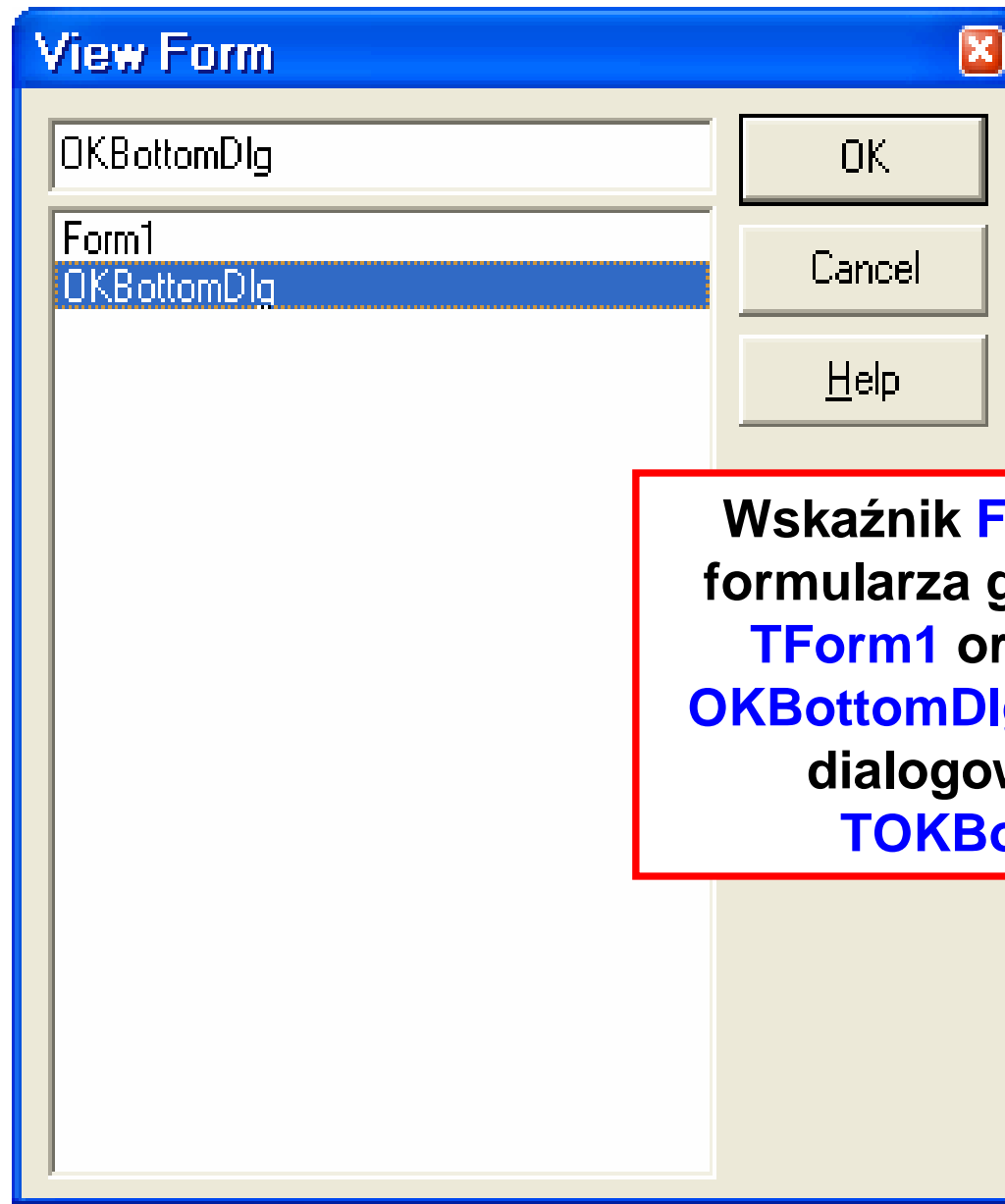
Utworzony automatycznie
plik nagłówekowy **dprodukt1.h**
standardowego okna
dialogowego - po zmianie
nazwy

At the bottom of the IDE window, the status bar shows "1: 1" and "Insert".

```
Rachunki.cpp
RachunekApp.h  dprodukt.h  Rachunki.cpp

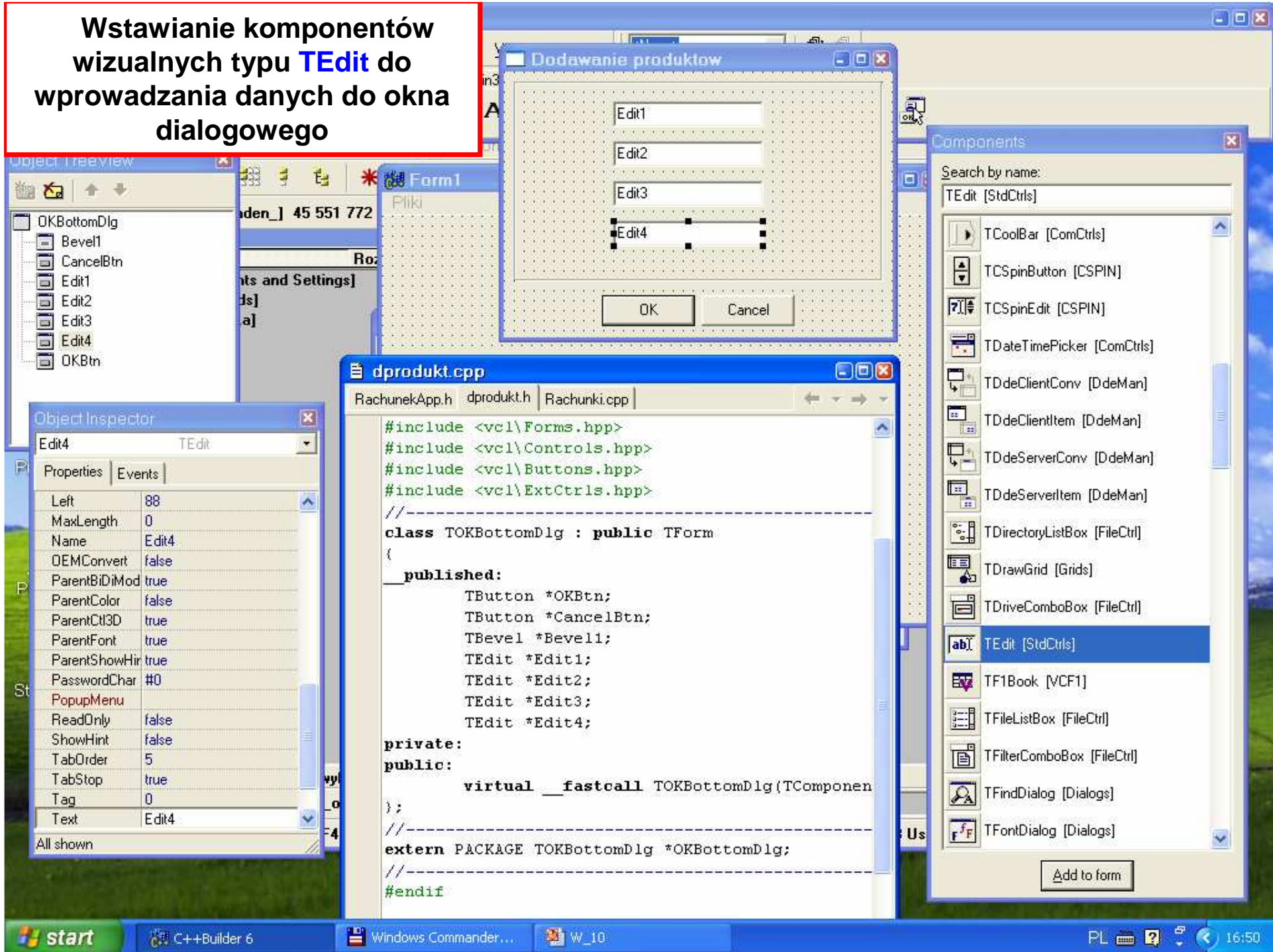
//-----
#include <vc1.h>
#pragma hdrstop
//-----
USEFORM("RachunekApp.cpp", Form1);
USEFORM("dprodukt.cpp", OKBottomDlg);
//-----
WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR,
{
    try
    {
        Application->Initialize();
        Application->CreateForm(__classid(TForm1), &Form1);
        Application->CreateForm(__classid(TOKBottomDlg), &OKBottomDlg);
        Application->Run();
    }
    catch (Exception &exception)
    {
        Application->ShowException(&exception);
    }
    catch (...)
    {
        try
        {
            throw Exception("");
        }
        catch (Exception &exception)
        {
            Application->ShowException(&exception);
        }
    }
    return 0;
}
```

Główny plik GUI z
automatycznie dołączonym
obiektem formularza
głównego typu **TForm1** oraz
obiektem okna dialogowego
typu **TOKBottomDlg**



Wskaźnik **Form1** obiektu formularza głównego typu **TForm1** oraz wskaźnik **OKBottomDlg** obiektu okna dialogowego typu **TOKBottomDlg**

Wstawianie komponentów wizualnych typu TEdit do wprowadzania danych do okna dialogowego



Wstawianie komponentów wizualnych typu TLabel jako etykiet pól edycyjnych do okna dialogowego

The screenshot displays the Delphi IDE interface with several windows open:

- Object Treeview:** Shows the component hierarchy for the form, including OKBottomDlg, Bevel1, CancelBtn, Edit1, Edit2, Edit3, Edit4, Label1, Label2, Label3, and Label4.
- Object Inspector:** Shows the properties for the selected TLabel component (Label4), such as Align, Alignment, Anchors, AutoSize, BiDiMode, Caption, Color, Constraints, Cursor, DragCursor, DragKind, DragMode, Enabled, FocusControl, Font, Height, and HelpContext.
- Code Editor (dprodukt.cpp):** Shows the C++ code for the TOKBottomDlg class, including header files and member variables for buttons, bevels, edit boxes, and labels.
- Components Palette:** Shows a list of available components, with TLabel [StdCtrls] selected.
- Dialog Box (Dodawanie produktow):** Shows a visual representation of the dialog box with four edit boxes (Edit1-4) and their corresponding labels (Label1-4).

```
#include <vc1\Forms.hpp>
#include <vc1\Controls.hpp>
#include <vc1\Buttons.hpp>
#include <vc1\ExtCtrls.hpp>
//-----
class TOKBottomDlg : public TForm
{
    published:
        TButton *OKBtn;
        TButton *CancelBtn;
        TBevel *Bevel1;
        TEdit *Edit1;
        TEdit *Edit2;
        TEdit *Edit3;
        TEdit *Edit4;
        TLabel *Label1;
        TLabel *Label2;
        TLabel *Label3;
        TLabel *Label4;
    private:
    public:
        virtual __fastcall TOKBottomDlg (TComponent
```

Dodawanie produktów

Nazwa

Cena

Podatek

Promocja

OK Cancel

**Gotowy
projekt okna
dialogowego**

Dodawanie produktów X

Nazwa	<input type="text" value="1"/>
Cena	<input type="text" value="1"/>
Podatek	<input type="text" value="1"/>
Promocja	<input type="text" value="1"/>

Uruchomione okno dialogowe

```
dprodukt.cpp
RachunekApp.cpp  dprodukt.h

#ifndef dproduktH
#define dproduktH
//-----
#include <vc1\System.hpp>
#include <vc1\Windows.hpp>
#include <vc1\SysUtils.hpp>
#include <vc1\Classes.hpp>
#include <vc1\Graphics.hpp>
#include <vc1\StdCtrls.hpp>
#include <vc1\Forms.hpp>
#include <vc1\Controls.hpp>
#include <vc1\Buttons.hpp>
#include <vc1\ExtCtrls.hpp>
#include "RachunekApp.h"

//-----
class TOKBottomDlg : public TForm
{
    __published:
        TButton *OKBtn;
        TButton *CancelBtn;
        TBevel *Bevel1;
        TEdit *Edit1;
        TEdit *Edit2;
        TEdit *Edit3;
        TEdit *Edit4;
        TLabel *Label1;
        TLabel *Label2;
        TLabel *Label3;
        TLabel *Label4;
    private:
    public: string tab[5]; //recznie - tablica na atrybuty lancuchowe produktu
        bool __fastcall Execute(); //recznie - obsluga okienka dialogowego
        virtual __fastcall TOKBottomDlg(TComponent* AOwner);
};
//-----
extern PACKAGE TOKBottomDlg *OKBottomDlg; //deklaracja globalnego wskaznika
```

Zawartość pliku nagłówkowego okna dialogowego po wykonaniu projektu

Definicja metody **Execute()** do wprowadzania danych w oknie dialogowym

```
#include <vcl.h>
#pragma hdrstop
#include "dprodukt.h"
//-----
#pragma resource "*.dfm"
TOKBottomDlg *OKBottomDlg; //wskaznik globalny okna dialogowego
//-----
__fastcall TOKBottomDlg::TOKBottomDlg(TComponent* AOwner): TForm(AOwner)
( )
//-----
bool __fastcall TOKBottomDlg::Execute() //recznie
(
    Edit1->Text=""; //recznie
    Edit2->Text="";
    Edit3->Text="";
    Edit4->Text="";
    if (ShowModal() ==mrOk) //przejscie do trybu modalnego, który konczy sie
        //nacisnieciem klawisza OK lub Cancel;
    (
        //pobranie atrybutow produktu z pol tekstowych wejsciowych
        tab[1]=Edit1->Text.c_str(); //nazwa
        tab[2]=Edit2->Text.c_str(); //cena netto
        tab[3]=Edit3->Text.c_str(); //podatek
        tab[4]=Edit4->Text.c_str(); //promocja
        //oraz wyznaczenie typu produktu w tab[0] w zaleznosci, czy wprowadzono
        //podatek oraz promocje
        if(tab[3]!="")
            if(tab[4]!="")
                tab[0]="4"; //jest promocja i podatek - typ TProdukt4
            else tab[0]="2"; //jest podatek - typ TProdukt2
        else
            if(tab[4]!="")
            (
                tab[0]="3"; //jest promocja - typ TProdukt3
                tab[3]=tab[4]; //korekta przekazania promocji, jesli brak podatku
            )
            else tab[0]="1"; //brak podatku i promocji - typ TProdukt1
        return true;}
    else
        return false;
}
```

#include "dprodukt.h"
The status bar at the bottom shows '14: 26', 'Modified', and 'Insert'."/>

```
// -----  
#ifndef RachunekApp1H  
#define RachunekApp1H  
// -----  
#include <Classes.hpp>  
#include <Controls.hpp>  
#include <StdCtrls.hpp>  
#include <Forms.hpp>  
#include <Menus.hpp>  
#include <Dialogs.hpp>  
#include <ExtCtrls.hpp>  
#include <Buttons.hpp>  
#include "TAPLIKACJA.h"  
#include "dprodukt.h"
```

Zawartość pliku nagłówkowego formularza głównego – dołączenie pliku nagłówkowego aplikacji z danymi oraz pliku nagłówkowego okna dialogowego – część pierwsza

```
C:\Settings\dydaktyka\Programowanie_obiektowe\w_10_2\RachunekApp.cpp
RachunekApp1.cpp  RachunekApp.h

#include <Dialogs.hpp>
#include <ExtCtrls.hpp>
#include <Buttons.hpp>
#include "TAplikacja.h"
#include "dprodukt.h",
//-----
class TForm1 : public TForm
{
    __published:      // IDE-managed Components
        TMainMenu *MainMenu1;
        TMenuItem *Plik1;
        TMenuItem *Dane1;
        TMenuItem *Wyswietl1;
        TMenuItem *Oprogramie1;
        TMenuItem *Dodajprodukt1;
        TMenuItem *Dodajrachunek1;
        TMenuItem *Dodajzakup1;
        TMenuItem *Wyswietlprodukty2;
        TMenuItem *Wyswietlrachunek2;
        TMenuItem *N1;
        TMenuItem *Koniec1;
        TMenuItem *oplik1;
        TMenuItem *zplik1;
        void __fastcall Dodajprodukt1Click(TObject *Sender);
private:             // User declarations
public:              // User declarations
    TAplikacja aplikacja; //połączenie z obiektem uzytkownika
    __fastcall TForm1(TComponent* Owner);
};
//-----
extern PACKAGE TForm1 *Form1; //deklaracja globalnego wskaznika
//-----
#endif
```

Zawartość pliku nagłówkowego okna formularza głównego po wykonaniu projektu – cd.

Połączenie GUI z obiektami aplikacji za pomocą obiektu hermetyzującego aplikacja typu **TAplikacja**

The image shows a screenshot of the C++Builder 6 IDE. The main window displays the source code for `RachunekApp.cpp`. The code defines a `TForm1` class that inherits from `TForm`. It lists several menu items in the `published` section, including `*Dodajprodukt1`. A method `void __fastcall Dodajprodukt1Click(TObject *Sender);` is also visible. The `Object TreeView` on the left shows the form structure with `MainMenu1`. The `Object Inspector` at the bottom left shows properties for `Form1`. A dialog box titled "Sporządzanie rachunków" is open, showing a menu with the option "Dodaj produkt" selected. A red box highlights a text block on the right side of the image, which explains the connection between the menu item and the code method.

```
class TForm1 : public TForm
(
    published:    // IDE-managed Components
    TMainMenu *MainMenu;
    TMenuItem *Plik1;
    TMenuItem *Dane1;
    TMenuItem *Wyswietl1;
    TMenuItem *Oprogramie1;
    TMenuItem *Dodajprodukt1;
    TMenuItem *Dodajrachunek1;
    TMenuItem *Dodajzakup1;
    TMenuItem *Wyswietlprodukty2;
    TMenuItem *Wyswietlrachunek2;
    TMenuItem *N1;
    TMenuItem *Koniec1;
    TMenuItem *oplik1;
    TMenuItem *zplik1;
    void __fastcall Dodajprodukt1Click(TObject *Sender);
);
```

Po kliknięciu na „Dodaj produkt” można przejść do pisania kodu automatycznie dołączonej metody `Dodajprodukt1Click` obsługującej dodawanie produktów w pliku `RachunekApp.cpp` głównego formularza aplikacji

definicja metody pobierającej dane z okna dialogowego **OKBottomDlg** i wywołanie metody **Wstaw_produkt** obiektu aplikacja

```
//automatycznie
void __fastcall TForm1::Dodajprodukt1Click(TObject *Sender)
{ int wynik; //recznie
  if (OKBottomDlg->Execute())
  { if (OKBottomDlg->tab[1] != "" && OKBottomDlg->tab[2] != "")
    { wynik=aplikacja.Wstaw_produkt(OKBottomDlg->tab);
      Application->MessageBox(napisy[wynik], "");
    }
  }
  else
    Application->MessageBox("Nie podano nazwy lub ceny produktu", "");
}
else
  Application->MessageBox("Anulowano dodanie produktu", "");
}
```

RachunekApp.cpp

dprodukt.cpp drachunek.cpp RachunekApp.cpp

```
#include <vcl.h>
#pragma hdrstop
#include "RachunekApp.h"
//-----
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1; //wskaznik globalny okna formularza
char* napisy[]={"Brak miejsca w kolekcji",
               "Proba wstawienia tej samej danej",
               "Ustawiono dane",
               "Brak pamieci",
               "Brak rachunku",
               "Brak produktu",
               "Wybrany produkt kupiono wczesniej - powiekszo jego ilosc"};
//-----
__fastcall TForm1::TForm1(TComponent* Owner) //automatycznie
    : TForm(Owner)
{
}
```

Zawartość pliku
modułowego okna
formularza głównego
– wstawienie tablicy
napisy z
komunikatami

5: 1

Modified

Insert

Dodawanie produktów

Nazwa	1
Cena	1
Podatek	1
Promocja	1

OK Cancel

Sporządzenie rachunków

- Dodaj produkt
- Dodaj rachunek
- Dodaj zakup

Układy zawartości

Wstawiono dane

OK

Uruchomienie aplikacji z oknem dialogowym do wprowadzania danych – przypadek poprawnego wprowadzania danych

Microsoft PowerPoint - [W_10]

Pliki Edycja Widok Wstaw Format Narzędzia Pokaż slajdów

56 57 58 59 60 61

Kliknij, aby dodać notatki

Slajd 59 z 71 Projekt domyślny Polski

start C++Builder 6 Windows Commander ... W_10 Rachunki PL 18:26

The image shows a screenshot of a Microsoft PowerPoint presentation with several overlapping windows:

- Microsoft PowerPoint - [W_10]**: The main application window with a menu bar (Plik, Edycja, Widok, Wstaw, Format, Narzędzia, Pokaż slajdy) and a toolbar. The slide thumbnail pane on the left shows slides 56 through 61.
- Sporządzenie rachunków**: A menu window with options: Dodaj produkt, Dodaj rachunek, and Dodaj zakup.
- Dodawanie produktów**: A dialog box with four input fields, each containing the number '1':
 - Nazwa: 1
 - Cena: 1
 - Podatek: 1
 - Promocja: 1Buttons for OK and Cancel are at the bottom.
- Proba wstawienia tej samej danej**: An error message dialog box with an OK button.
- Uruchomienie aplikacji z oknem dialogowym do wprowadzania danych – przypadek ponownego wprowadzenia tej samej danej**: A text box with a red border explaining the error.

The Windows taskbar at the bottom shows the Start button, taskbar icons for C++Builder 6, Windows Commander..., W_10, and Rachunki, and the system tray with the date 18:26.

The image shows a screenshot of a Windows XP desktop environment. In the background, there is a Microsoft PowerPoint window titled "Microsoft PowerPoint - [W_10]" with a menu bar including "Plik", "Edycja", "Widok", "Wstaw", "Format", "Narzędzia", "Pokaż slajdów", and "Okno". Below the menu bar is a toolbar with various icons. A slide thumbnail pane on the left shows slides numbered 56, 57, 58, and 59. A "Sporządzanie rachunków" (Account Management) application window is open, displaying a menu with options: "Dodaj produkt", "Dodaj rachunek", and "Dodaj zakup".

In the foreground, a dialog box titled "Dodawanie produktów" (Adding products) is displayed. It contains four input fields: "Nazwa" (Name) with the value "1", "Cena" (Price) which is empty, "Podatek" (Tax) which is empty, and "Promocja" (Promotion) with the value "1". At the bottom of the dialog are "OK" and "Cancel" buttons.

Below the "Dodawanie produktów" dialog, an error message dialog box is shown with the text "Nie podano nazwy lub ceny produktu" (Name or price of product not provided) and an "OK" button.

A red-bordered text box in the bottom-left corner contains the following text:

Uruchomienie aplikacji z oknem dialogowym do wprowadzania danych – przypadek nie poprawnego wprowadzania danych

The taskbar at the bottom shows the Start button, several open applications including "C++Builder 6", "Windows Commander...", "W_10", and "Rachunki", and the system tray with the date "18:26" and language "PL".

Budowa aplikacji z graficznym interfejsem użytkownika - GUI (Graphic User Interface)

- 1. Udostępnianie wszystkich prywatnych atrybutów do prezentacji, wprowadzenie standardu nazewnictwa plików – nazwy plików aplikacji poprzedzone literą T**
- 2. Budowa głównego formularza GUI**
- 3. Budowa okienek dialogowych do wprowadzania danych**
- 4. Budowa okienek dialogowych do wyświetlania danych**

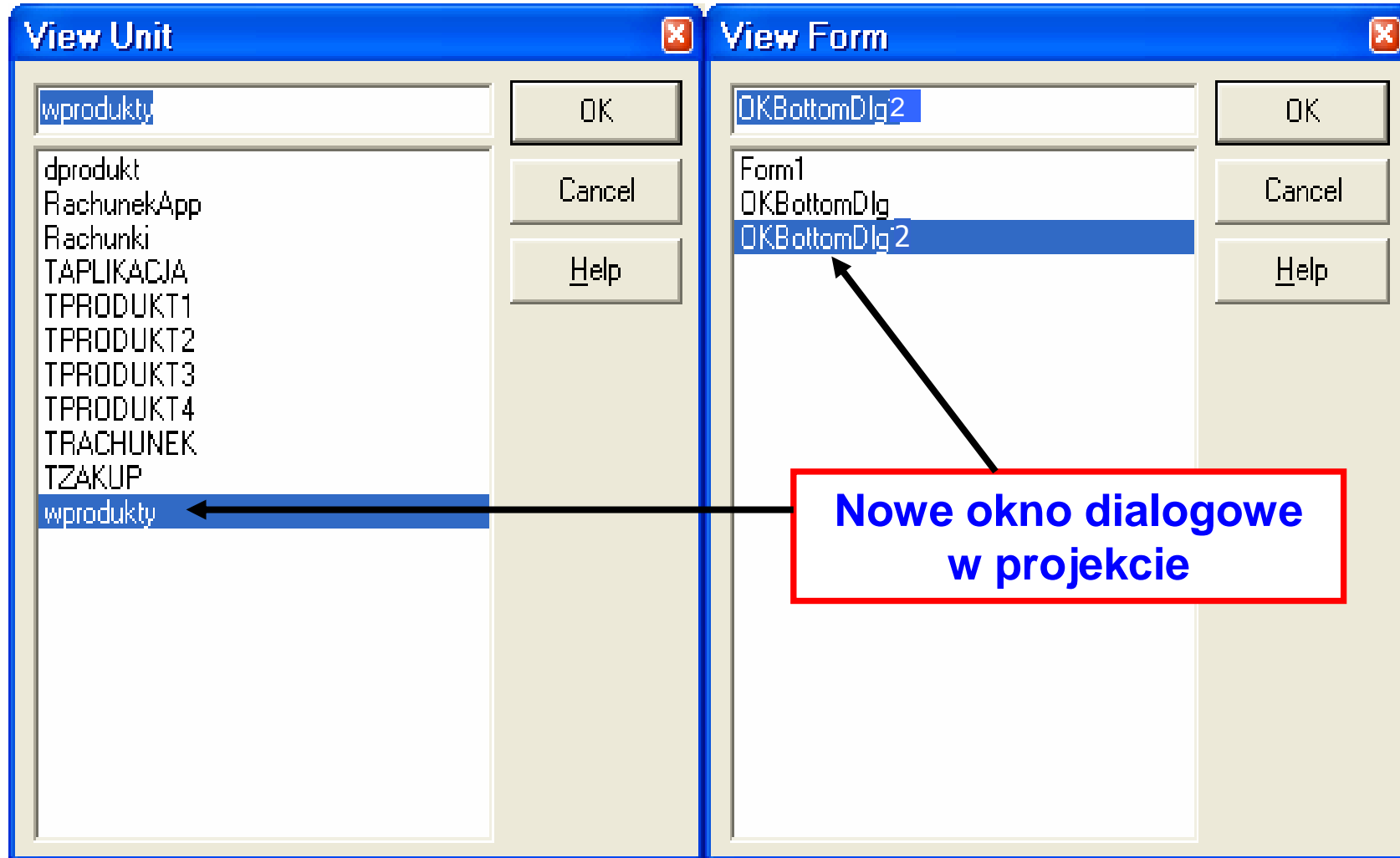
Dodanie nowego standardowego okna dialogowego do projektu, nadanie plikom nazwy **wprodukty.h** i **wprodukty.cpp** i wstawienie komponentu **TStringGrid**

The screenshot displays the C++Builder 6 IDE interface. At the top, the menu bar includes File, Edit, Search, View, and Project. The main workspace shows a form titled 'Form1' with a code editor containing the following content:

```
wprodukty.cpp  
RachunekApp.cpp | dprodukt.h | wprodukty.cpp  
//  
//
```

A dialog window titled 'Wyświetlenie produktów' is overlaid on the form. It contains a TStringGrid component with 4 columns and 4 rows. The grid is currently empty. Below the grid are 'OK' and 'Cancel' buttons. The Object Inspector on the left shows the properties of the 'StringGrid1' component, including 'Caption' (wyświetlenie produktów), 'ColCount' (4), and 'RowCount' (4). The Components palette on the right shows the 'TStringGrid [Grids]' component selected. The Windows taskbar at the bottom shows the Start button, C++Builder 6, Windows Commander, and W_10, along with the system tray showing the time as 17:33.

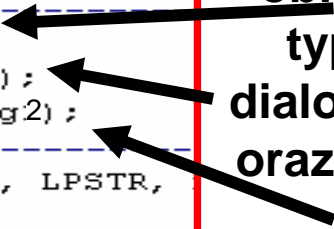
Dodanie nowego okna dialogowego do projektu



```
Rachunki.cpp
RachunekApp.h  Rachunki.cpp  dprodukt.h  wprodukty.cpp

//-----
#include <vc1.h>
#pragma hdrstop
//-----
USEFORM("RachunekApp.cpp", Form1);
USEFORM("dprodukt.cpp", OKBottomDlg);
USEFORM("wprodukty.cpp", OKBottomDlg2);
//-----
WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR,
(
    try
    (
        Application->Initialize();
        Application->CreateForm(__classid(TForm1), &Form1);
        Application->CreateForm(__classid(TOKBottomDlg), &OKBottomDlg);
        Application->CreateForm(__classid(TOKBottomDlg2), &OKBottomDlg2);
        Application->Run();
    )
    catch (Exception &exception)
    (
        Application->ShowException(&exception);
    )
    catch (...)
    (
        try
        (
            throw Exception("");
        )
        catch (Exception &exception)
        (
            Application->ShowException(&exception);
        )
    )
    return 0;
)
//-----
```

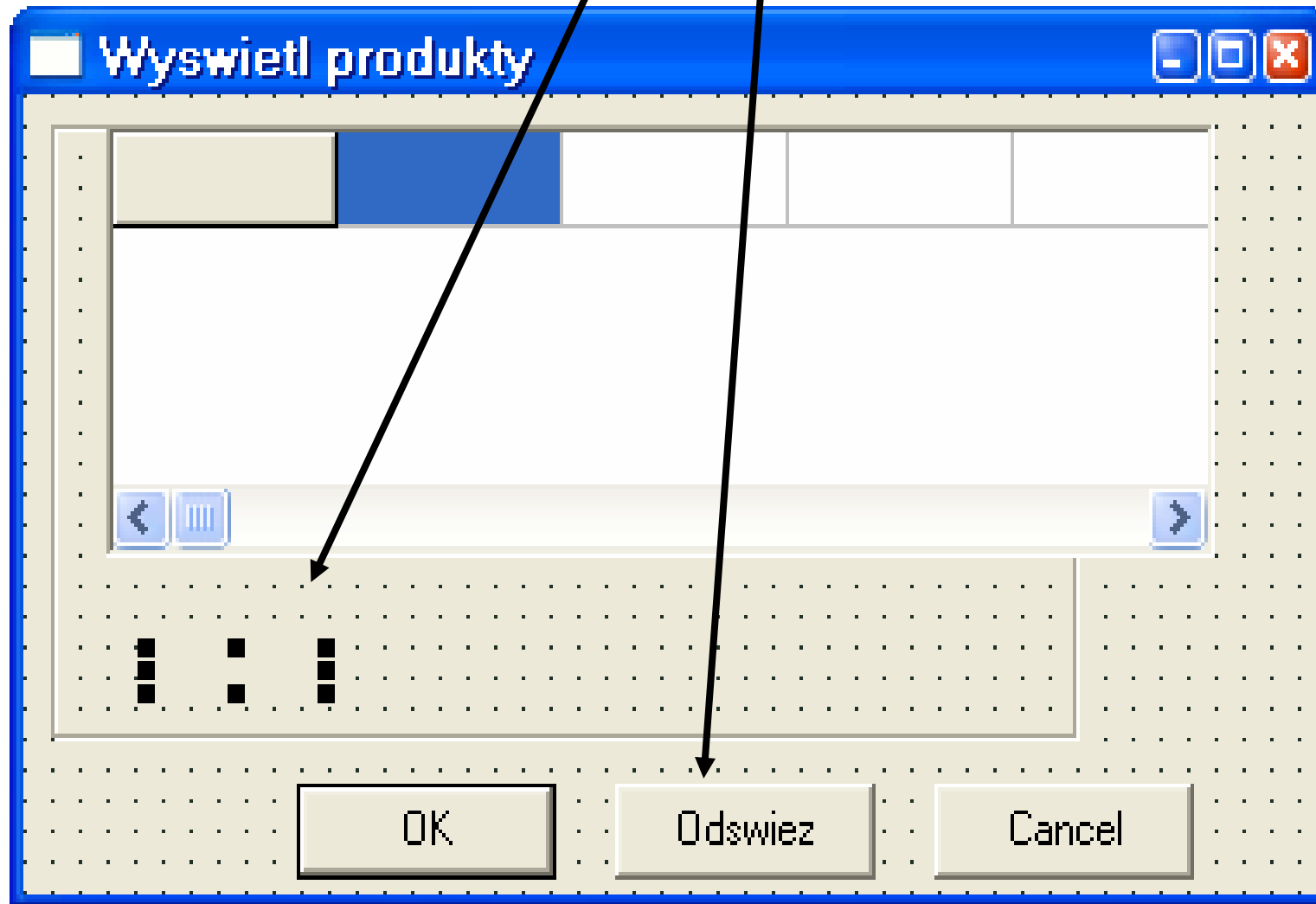
Główny plik GUI z
automatycznie dołączonym
obiektem formularza głównego
typu **TForm1**, obiektem okna
dialogowego typu **TOKBottomDlg**
oraz obiektem okna dialogowego
typu **TOKBottomDlg2**



```
wprodukty.cpp
RachunekApp.h | Rachunki.cpp | dprodukt.h | wprodukty.h
//-----
#ifndef wproduktyH
#define wproduktyH
//-----
#include <vc1\System.hpp>
#include <vc1\Windows.hpp>
#include <vc1\SysUtils.hpp>
#include <vc1\Classes.hpp>
#include <vc1\Graphics.hpp>
#include <vc1\StdCtrls.hpp>
#include <vc1\Forms.hpp>
#include <vc1\Controls.hpp>
#include <vc1\Buttons.hpp>
#include <vc1\ExtCtrls.hpp>
#include <Grids.hpp>
//-----
class TOKBottomDlg2 : public TForm
(
    __published:
        TButton *OKBtn;
        TButton *CancelBtn;
        TBevel *Bevel1;
        TStringGrid *StringGrid1;
private:
public:
    virtual __fastcall TOKBottomDlg2(TComponent* AOwner);
};
//-----
extern PACKAGE TOKBottomDlg2 *OKBottomDlg2;
//-----
#endif
```

Plik nagłówekowy standardowego okna dialogowego po wstawieniu komponentu **TStringGrid**

Wstawienie przycisku typu **TButton** do odświeżania zawartości okna oraz pola komunikatów typu **TStaticText**



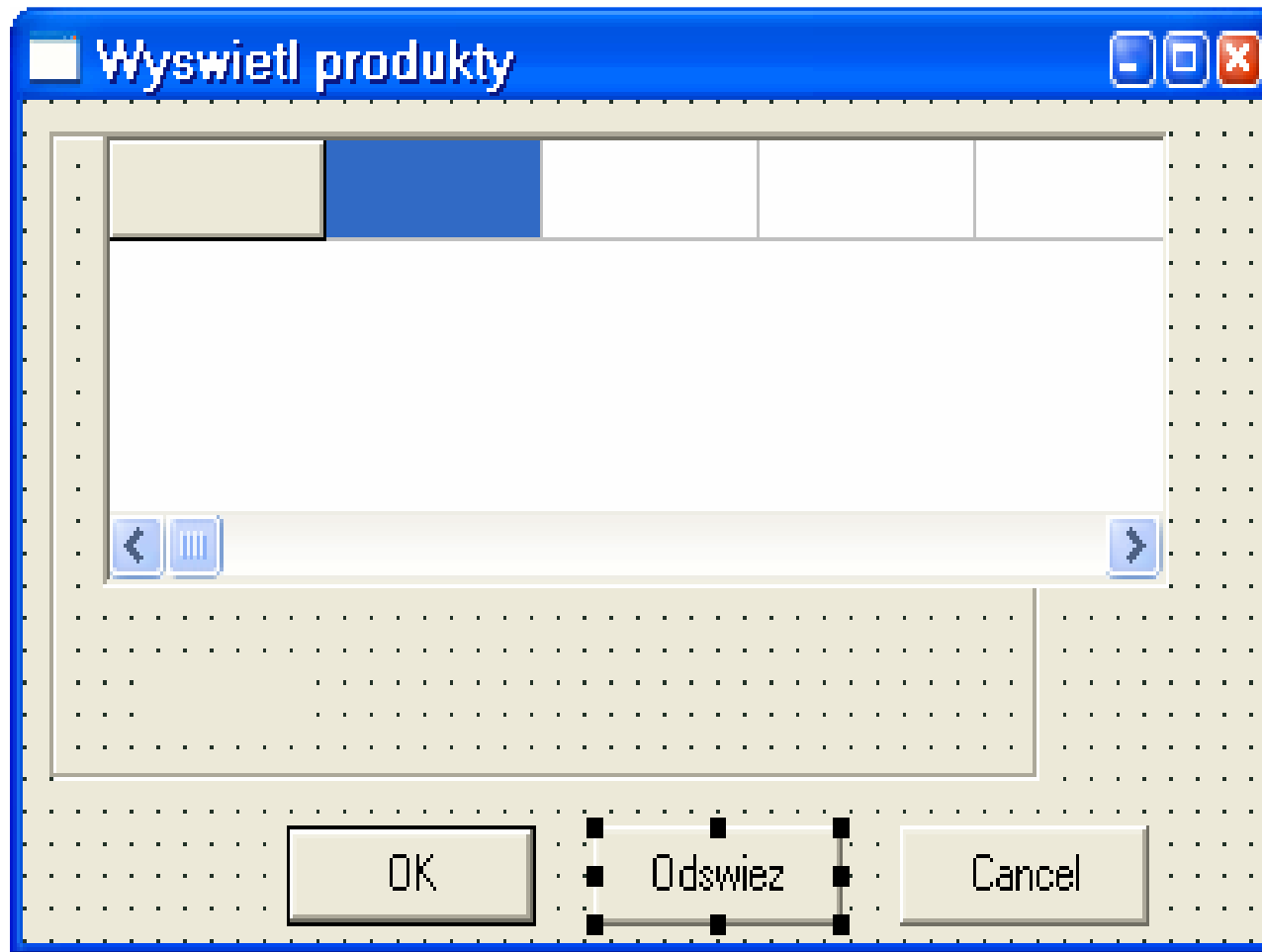
```
wprodukty.cpp
RachunekApp.cpp  wprodukty.h  dprodukt.cpp

#include <vc1\Forms.hpp>
#include <vc1\Controls.hpp>
#include <vc1\Buttons.hpp>
#include <vc1\ExtCtrls.hpp>
#include <Grids.hpp>
#include "RachunekApp.h"
//-----
class TOKBottomDlg2 : public TForm
{
  __published:
    TButton *OKBtn;
    TButton *CancelBtn;
    TBevel *Bevel1;
    TStringGrid *StringGrid1;
    TButton *Button1;
    TLabel *Label1;
    void __fastcall StringGrid1Click(TObject *Sender);
    void __fastcall Button1Click(TObject *Sender);
private: // ręcznie = obliczanie ceny netto na podstawie ceny brutto
string __fastcall cena_netto4(string a, string b, string c); // oraz podatku i promocji
string __fastcall cena_netto3(string a, string b); // oraz promocji
string __fastcall cena_netto2(string a, string b); // oraz podatku
public:
    bool wybrany; // ręcznie - zmienna na wybrany rowek oprócz pierwszego
    string tab[5]; // ręcznie - tablica na atrybuty lancuchowe produktu
    bool __fastcall Execute(); // ręcznie - obsługa okienka dialogowego
    virtual __fastcall TOKBottomDlg2(TComponent* AOwner);
};
//-----
extern PACKAGE TOKBottomDlg2 *OKBottomDlg2; // deklaracja globalnego wskaźnika
//-----
#endif
```

Plik nagłówkowy standardowego okna dialogowego po wstawieniu komponentu **TStringGrid**, przycisku typu **TButton**, pola komunikatów **TStaticText** oraz metody **Execute**, atrybutów **wybrany** oraz **tab**

```
#include <vcl.h>
#pragma hdrstop
#include "wprodukty.h"
//-----
#pragma resource "*.dfm"
TOKBottomDlg2 *OKBottomDlg2; //wskaznik globalny okna dialogowego
//-----
__fastcall TOKBottomDlg2::TOKBottomDlg2(TComponent* AOwner) //automatycznie
: TForm(AOwner)
{
    StringGrid1->Cells[0][0]="L.p."; //utworzenie jednorazowo
    StringGrid1->Cells[1][0]="Nazwa"; //nagłówka tabeli w konstruktorze
    StringGrid1->Cells[2][0]="Cena"; //okna dialogowego
    StringGrid1->Cells[3][0]="Podatek";
    StringGrid1->Cells[4][0]="Promocja";
    StringGrid1->Visible=true; }
//-----
bool __fastcall TOKBottomDlg2::Execute() //ręcznie
{
    //wyswietlanie tabeli
    ShowModal(); //przejscie do trybu modalnego, który konczy sie
    return true; //nacisnieciem klawisza OK lub Cancel;
}
```

Plik modułowy standardowego okna dialogowego – definicja metody **Execute()** wyświetlającej zawartość komponentu typu **TStringGrid** oraz konstruktor wstawiający nagłówki do tabeli typu **TStringGrid**



Po kliknięciu na przycisk typu `TButton` z napisem „Odswiez” przechodzi się do pliku źródłowego okienka dialogowego `wprodukty.cpp`, gdzie wygenerowała się pusta metoda do obsługi tego klawisza

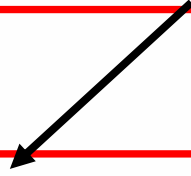
```
void __fastcall TOKBottomDlg2::Button1Click(TObject *Sender)
```

```
wprodukty.cpp
RachunekApp.cpp  wprodukty.h  dprodukt.cpp

#include <vc1\Forms.hpp>
#include <vc1\Controls.hpp>
#include <vc1\Buttons.hpp>
#include <vc1\ExtCtrls.hpp>
#include <Grids.hpp>
#include "RachunekApp.h"

//-----
class TOKBottomDlg2 : public TForm
{
  __published:
    TButton *OKBtn;
    TButton *CancelBtn;
    TBevel *Bevel1;
    TStringGrid *StringGrid1;
    TButton *Button1;
    TLabel *Label1;
    void __fastcall StringGrid1Click(TObject *Sender);
    void __fastcall Button1Click(TObject *Sender);
private: // ręcznie = obliczanie ceny netto na podstawie ceny brutto
  string __fastcall cena_netto4(string a, string b, string c); // oraz podatku i promocji
  string __fastcall cena_netto3(string a, string b); // oraz promocji
  string __fastcall cena_netto2(string a, string b); // oraz podatku
public:
  bool wybrany; // ręcznie - zmienna na wybrany rowek oprócz pierwszego
  string tab[5]; // ręcznie - tablica na atrybuty lancuchowe produktu
  bool __fastcall Execute(); // ręcznie - obsługa okienka dialogowego
  virtual __fastcall TOKBottomDlg2(TComponent* AOwner);
};
//-----
extern PACKAGE TOKBottomDlg2 *OKBottomDlg2; // deklaracja globalnego wskaźnika
//-----
#endif
```

Metoda wywoływana podczas działania metody **Execute()**



void __fastcall Button1Click(TObject *Sender);

bool __fastcall Execute(); // ręcznie - obsługa okienka dialogowego


```
void __fastcall TOKBottomDlg2::Button1Click(TObject *Sender) //automatycznie
{
    //recznie
    TKol2<TProdukt1> produkty=Form1->aplikacja.Podaj_produkty();
    int i=0; TProdukt1* produkt;
    char lan[10];
    produkty.Zeruj(); //odczytywanie kolekcji od początku
    while (produkty.Koniec()==0) //odswiezanie zawartosci komorek tabeli
    { produkt=produkty.Podaj_nast();
      i++; //zmienna do liczenia rowkow tabeli
      TStringGrid1->RowCount=i+1; //dodanie nowego rowka do tabeli
      TStringGrid1->Cells[0][i]=i; //numer pozycji w rachunku
      TStringGrid1->Cells[1][i]=produkt->Podaj_nazwe().c_str();
      TStringGrid1->Cells[2][i]=gcvt(produkt->Podaj_cene(),3,lan);
      float pom=produkt->Podaj_podatek();
      if (pom==-1) //jesli brak podatku wyswietla sie pusty lancuch
        TStringGrid1->Cells[3][i]="";
      else
        TStringGrid1->Cells[3][i]=gcvt(produkt->Podaj_podatek(),3,lan);
      pom=produkt->Podaj_promocje();
      if (pom==-1) //jesli brak promocji wyswietla sie pusty lancuch
        TStringGrid1->Cells[4][i]="";
      else
        TStringGrid1->Cells[4][i]=gcvt(produkt->Podaj_promocje(),3,lan);
    }
    TStringGrid1->Visible=true;
}
```

```
#include <Dialogs.hpp>
#include <ExtCtrls.hpp>
#include <Buttons.hpp>
#include "TAplikacja.h"
#include "dprodukt.h"
#include "wprodukty.h"
//-----
class TForm1 : public TForm
{
    __published:      // IDE-managed Components
        TMainMenu *MainMenu1;
        TMenuItem *Plik1;
        TMenuItem *Dane1;
        TMenuItem *Wyświetl1;
        TMenuItem *Oprogramie1;
        TMenuItem *Dodajprodukt1;
        TMenuItem *Dodajrachunek1;
        TMenuItem *Dodajzakup1;
        TMenuItem *Wyświetlprodukty2;
        TMenuItem *Wyświetlrachunek2;
        TMenuItem *N1;
        TMenuItem *Koniec1;
        TMenuItem *oplik1;
        TMenuItem *zplik1;
        void __fastcall Dodajprodukt1Click(TObject *Sender);
        void __fastcall Wyświetlprodukty2Click(TObject *Sender);
private:      // User declarations
public:      // User declarations
        TAplikacja aplikacja; //połączenie z obiektem uzytkownika
        __fastcall TForm1(TComponent* Owner);
};
//-----
extern PACKAGE TForm1 *Form1; //deklaracja globalnego wskaznika
```

Plik nagłówkowy formularza głównego po wstawieniu okna dialogowego do wyświetlania produktów – wstawienie automatycznie metody **Wyświetlprodukty2Click** wywołującej to okno



```
void __fastcall TForm1::Wyswietlprodukty2Click(TObject *Sender) //automatycznie
{
    OKBottomDlg2->Execute(); //recznie
}
```

Po kliknięciu na „Wyswietl produkty” można przejść do pisania kodu metody obsługującej wyświetlanie produktów w pliku [RachunekApp.cpp](#) głównego formularza aplikacji

Dodawanie produktów

Nazwa

Cena

Podatek

Promocja

OK

Cancel

Uruchomienie aplikacji

Sporządzanie rachunkow

Pliki Dane Wyszwiel O programie

Wyszwiel produkty

Wyszwiel rachunki

Wyszwiel produkty

L.p.	Nazwa	Cena	Podatek	Promocja
------	-------	------	---------	----------



OK

Odswiez

Cancel

Wyszwiel produkty

L.p.	Nazwa	Cena	Podatek	Promocja
1	1	1	1	1



OK

Odswiez

Cancel

```
wprodukty.cpp
RachunekApp.cpp  wprodukty.h  dprodukt.cpp

#include <vc1\Forms.hpp>
#include <vc1\Controls.hpp>
#include <vc1\Buttons.hpp>
#include <vc1\ExtCtrls.hpp>
#include <Grids.hpp>
#include "RachunekApp.h"

//-----
class TOKBottomDlg2 : public TForm
{
  __published:
    TButton *OKBtn;
    TButton *CancelBtn;
    TBevel *Bevel1;
    TStringGrid *StringGrid1;
    TButton *Button1;
    TLabel *Label1;
    void __fastcall StringGrid1Click(TObject *Sender);
    void __fastcall Button1Click(TObject *Sender);
private: // ręcznie = obliczanie ceny netto na podstawie ceny brutto
  string __fastcall cena_netto4(string a, string b, string c); // oraz podatku i promocji
  string __fastcall cena_netto3(string a, string b); // oraz promocji
  string __fastcall cena_netto2(string a, string b); // oraz podatku
public:
  bool wybrany; // ręcznie - zmienna na wybrany rowek oprócz pierwszego
  string tab[5]; // ręcznie - tablica na atrybuty lancuchowe produktu
  bool __fastcall Execute(); // ręcznie - obsługa okienka dialogowego
  virtual __fastcall TOKBottomDlg2(TComponent* AOwner);
};
//-----
extern PACKAGE TOKBottomDlg2 *OKBottomDlg2; // deklaracja globalnego wskaźnika
//-----
#endif
```


Plik nagłówkowy standardowego okna dialogowego do wyświetlania produktów – deklaracja metody **StringGrid1Click** do obsługi wyboru wiersza z danymi w komponencie typu **TStringGrid** za pomocą klikania myszą na wybranym wierszu. Metoda ta jest wywoływana podczas działania metody **Execute()**

void __fastcall StringGrid1Click(TObject *Sender);




void __fastcall Button1Click(TObject *Sender);

bool __fastcall Execute(); // ręcznie - obsługa okienka dialogowego

virtual __fastcall TOKBottomDlg2(TComponent* AOwner);

Wyswietl produkty 

L.p.	Nazwa	Cena	Podatek	Promocja
1	1	1	1	1
2	2	2		

```
wprodukty.cpp
RachunekApp.cpp wprodukty.cpp dprodukt.cpp
void __fastcall TOKBottomDlg2::StringGrid1Click(TObject *Sender) //automatycznie
{
    char pom[10]; //ręcznie
    TStringList* lista;
    wybrany=true;
    if (StringGrid1->Row==0) //kliknieto na rowek bez danych
    {
        wybrany=false; //czyli nie wybrano danych
        return; //koniec wprowadzania danych o produkcji
    }
    lista= StringGrid1->Rows[StringGrid1->Row];
    for (int i=1; i<=4;i++)
        tab[i]=lista->Strings[i].c_str(); //pobranie z rowka atrybutow produktu
    //korekta zawartosci tab[2] reprezentujacego cene brutto na cene netto
    //oraz wyznaczenie typu produktu w tab[0] w zaleznosci, czy wprowadzono
    //podatek oraz promocje
    if (tab[3] != "")
    {
        if (tab[4] != "")
        {
            tab[0]="4"; //jest promocja i podatek = typ TProdukt4
            tab[2]=cena_netto4(tab[2],tab[3],tab[4]); //obliczanie ceny netto
        }
        else
        {
            tab[0]="2"; //jest promocja = typ TProdukt2
            tab[2]=cena_netto2(tab[2],tab[3]); //obliczanie ceny netto
        }
    }
    else
    {
        if (tab[4] != "")
        {
            tab[0]="3"; //jest promocja = typ TProdukt3
            tab[2]=cena_netto3(tab[2],tab[4]); //obliczanie ceny netto
            tab[3]=tab[4]; //korekta przekazania promocji, gdy brak podatku
        }
        else
        {
            tab[0]="1"; //brak promocji i podatku - jest TProdukt1
        }
    }
}
```

89: 26 Modified Insert

```
//ręcznie
```

```
• string __fastcall TOKBottomDlg2::cena_netto4(string a, string b, string c)
• { float x,y,z; char lan[10];
•   x=atof(a.c_str()); y=atof(b.c_str()); z=atof(c.c_str()); //cena netto
•   x=(x*100)/(100+y-z); //na podstawie ceny brutto x, podatku y i promocji z
•   return gcvt(x,3,lan);
• }

• string __fastcall TOKBottomDlg2::cena_netto3(string a, string b) //ręcznie
• { float x,y; char lan[10];
•   x=atof(a.c_str()); y=atof(b.c_str());
•   x=(x*100)/(100-y); //cena netto na podstawie ceny brutto x i promocji y
•   return gcvt(x,3,lan);
• }

• string __fastcall TOKBottomDlg2::cena_netto2(string a, string b) //ręcznie
• { float x,y; char lan[10];
•   x=atof(a.c_str()); y=atof(b.c_str());
•   x=(x*100)/(100+y); //cena netto na podstawie ceny brutto x i podatku y
•   return gcvt(x,3,lan);
• }
```


Budowa aplikacji z graficznym interfejsem użytkownika - GUI (Graphic User Interface)

- 1. Udostępnianie wszystkich prywatnych atrybutów do prezentacji, wprowadzenie standardu nazewnictwa plików – nazwy plików aplikacji poprzedzone literą T**
- 2. Budowa głównego formularza GUI**
- 3. Budowa okienek dialogowych do wprowadzania danych**
- 4. Budowa okienek dialogowych do wyświetlania danych**
- 5. Aplikacja do sporządzania rachunków -
uzupełnienie**

```
Rachunki.cpp
RachunekApp.h  Rachunki.cpp

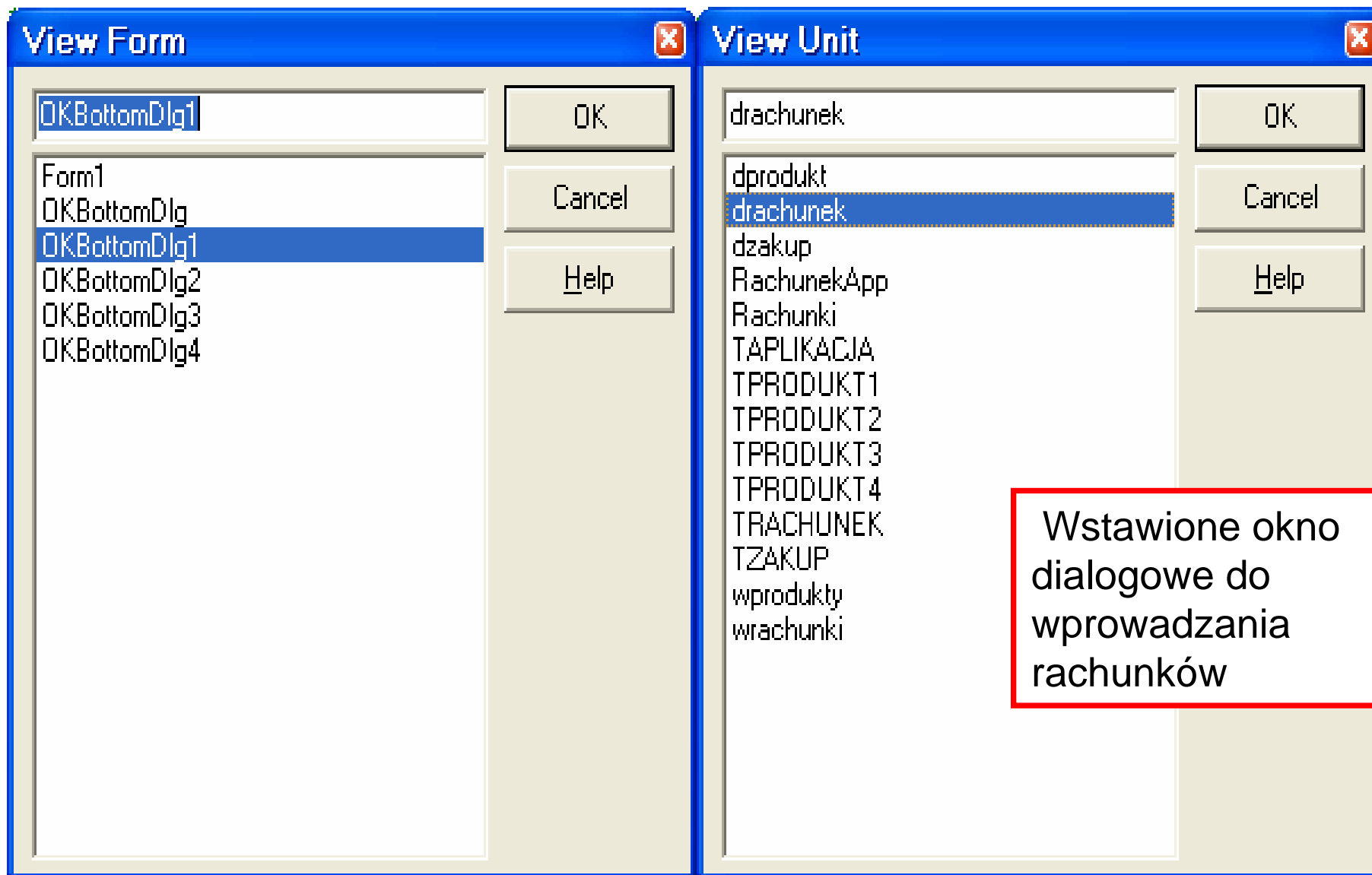
#include <vc1.h>
#pragma hdrstop
//-----
USEFORM("RachunekApp.cpp", Form1);
USEFORM("dprodukt.cpp", OKBottomDlg);
USEFORM("drachunek.cpp", OKBottomDlg1);
USEFORM("wprodukty.cpp", OKBottomDlg2);
USEFORM("wrachunki.cpp", OKBottomDlg3);
USEFORM("dzakup.cpp", OKBottomDlg4);
//-----
WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)
{
    try
    {
        Application->Initialize();
        Application->CreateForm(__classid(TForm1), &Form1);
        Application->CreateForm(__classid(TOKBottomDlg), &OKBottomDlg);
        Application->CreateForm(__classid(TOKBottomDlg1), &OKBottomDlg1);
        Application->CreateForm(__classid(TOKBottomDlg2), &OKBottomDlg2);
        Application->CreateForm(__classid(TOKBottomDlg3), &OKBottomDlg3);
        Application->CreateForm(__classid(TOKBottomDlg4), &OKBottomDlg4);
        Application->Run();
    }
    catch (Exception &exception)
    {
        Application->ShowException(&exception);
    }
    catch (...)
    {
        try
        {
            throw Exception("");
        }
        catch (Exception &exception)
        {
            Application->ShowException(&exception);
        }
    }
    return 0;
}
```

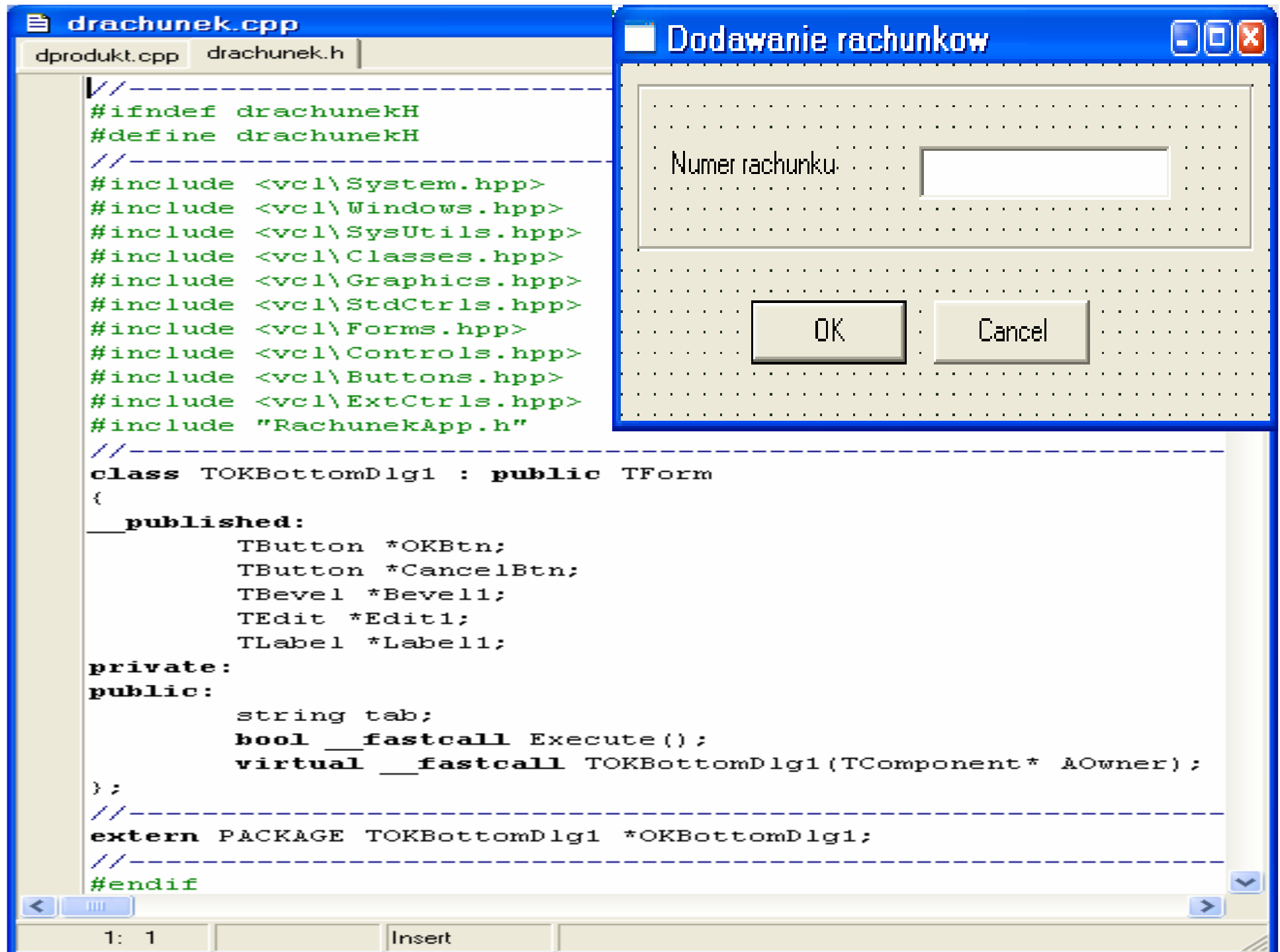
Główny plik GUI z
automatycznie dołączonym
obiektem formularza głównego
typu **TForm1** oraz obiektami
okienek dialogowych

```
//-----  
#ifndef RachunekAppH  
#define RachunekAppH  
//-----  
#include <Classes.hpp>  
#include <Controls.hpp>  
#include <StdCtrls.hpp>  
#include <Forms.hpp>  
#include <Menus.hpp>  
#include <Dialogs.hpp>  
#include <ExtCtrls.hpp>  
#include <Buttons.hpp>  
#include "T Aplikacja.h"  
#include "dprodukt.h"  
#include "drachunek.h"  
#include "dzakup.h";  
#include "wprodukty.h"  
#include "wrachunki.h"
```

```
class TForm1 : public TForm
{
    published:          // IDE-managed Components
        TMainMenu *MainMenu1;
        TMenuItem *Plik1;
        TMenuItem *Dane1;
        TMenuItem *Wyswietl1;
        TMenuItem *Oprogramie1;
        TMenuItem *Dodajprodukt1;
        TMenuItem *Dodajrachunek1;
        TMenuItem *Dodajzakup1;
        TMenuItem *Wyswietlprodukty2;
        TMenuItem *Wyswietlrachunek2;
        TMenuItem *N1;
        TMenuItem *Koniec1;
        TMenuItem *oplik1;
        TMenuItem *zplik1;
        void __fastcall Dodajprodukt1Click(TObject *Sender);
        void __fastcall Wyswietlprodukty2Click(TObject *Sender);
        void __fastcall Dodajrachunek1Click(TObject *Sender);
        void __fastcall Wyswietlrachunek2Click(TObject *Sender);
        void __fastcall Dodajzakup1Click(TObject *Sender);
        void __fastcall Oprogramie1Click(TObject *Sender);
private:           // User declarations
public:           // User declarations
        TApplikacja aplikacja; //połączenie z obiektem uzytkownika
        __fastcall TForm1(TComponent* Owner);
};
//-----
extern PACKAGE TForm1 *Form1; //deklaracja globalnego wskaznika
//-----
#endif
```

Plik nagłówkowy formularza z metodami obsługującymi klikania w pozycje list rozwijanych menu formularza





```
#include <vcl.h>
#pragma hdrstop

#include "drachunek.h"
//-----
#pragma resource "*.dfm"
TOKBottomDlg1 *OKBottomDlg1; //wskaznik globalny okna dialogowego
//-----
__fastcall TOKBottomDlg1::TOKBottomDlg1(TComponent* AOwner)
    : TForm(AOwner)
{
}
//-----
bool __fastcall TOKBottomDlg1::Execute()
{
    //przejscie do trybu modalnego, ktory konczy sie
    if (ShowModal() == mrOk) //nacisnieciem klawisza OK lub Cancel;
    {
        tab=Edit1->Text.c_str();
        return true;}
    else
        return false;
}
```

Obsługa wprowadzania numeru rachunku w oknie dialogowym za pomocą metody [Execute\(\)](#)

RachunekApp.cpp

dprodukt.cpp drachunek.cpp RachunekApp.cpp

```
//automatycznie
void __fastcall TForm1::Dodajrachunek1Click(TObject *Sender)
{ int wynik; //recznie
  if(OKBottomDlg1->Execute())
  { if(OKBottomDlg1->tab!="")
    { wynik=aplikacja.Wstaw_rachunek(atoi((OKBottomDlg1->tab).c_str()));
      Application->MessageBox(napisy[wynik], "");
    }
  else
    Application->MessageBox("Nie podano numeru rachunku", "");
  }
  else
    Application->MessageBox("Anulowano dodanie rachunku", "");
}
```

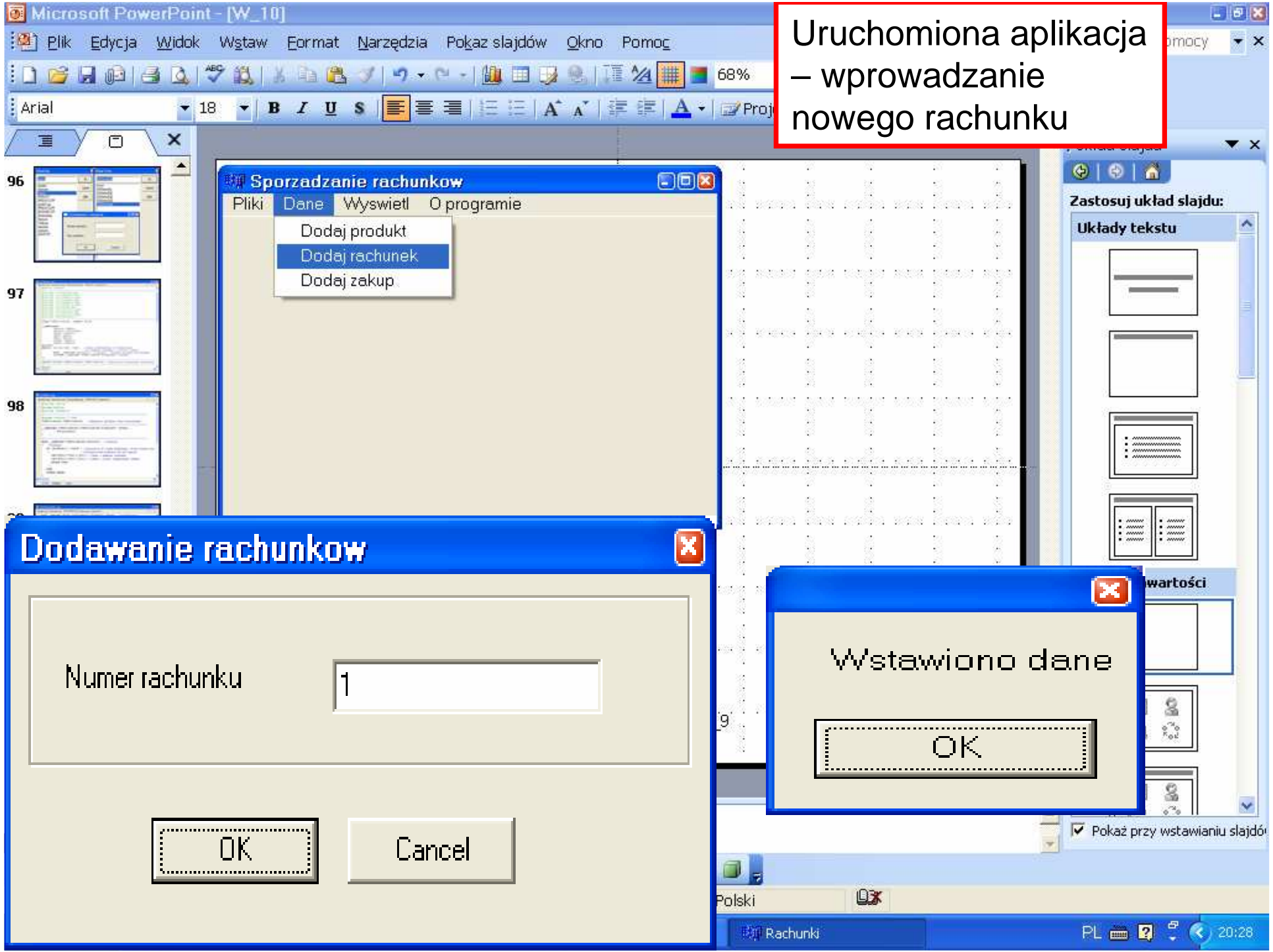
Metoda **Dodajrachunek1Click** w formularzu głównym pobierająca numery nowego rachunku z okna dialogowego za pomocą metody **Execute()**

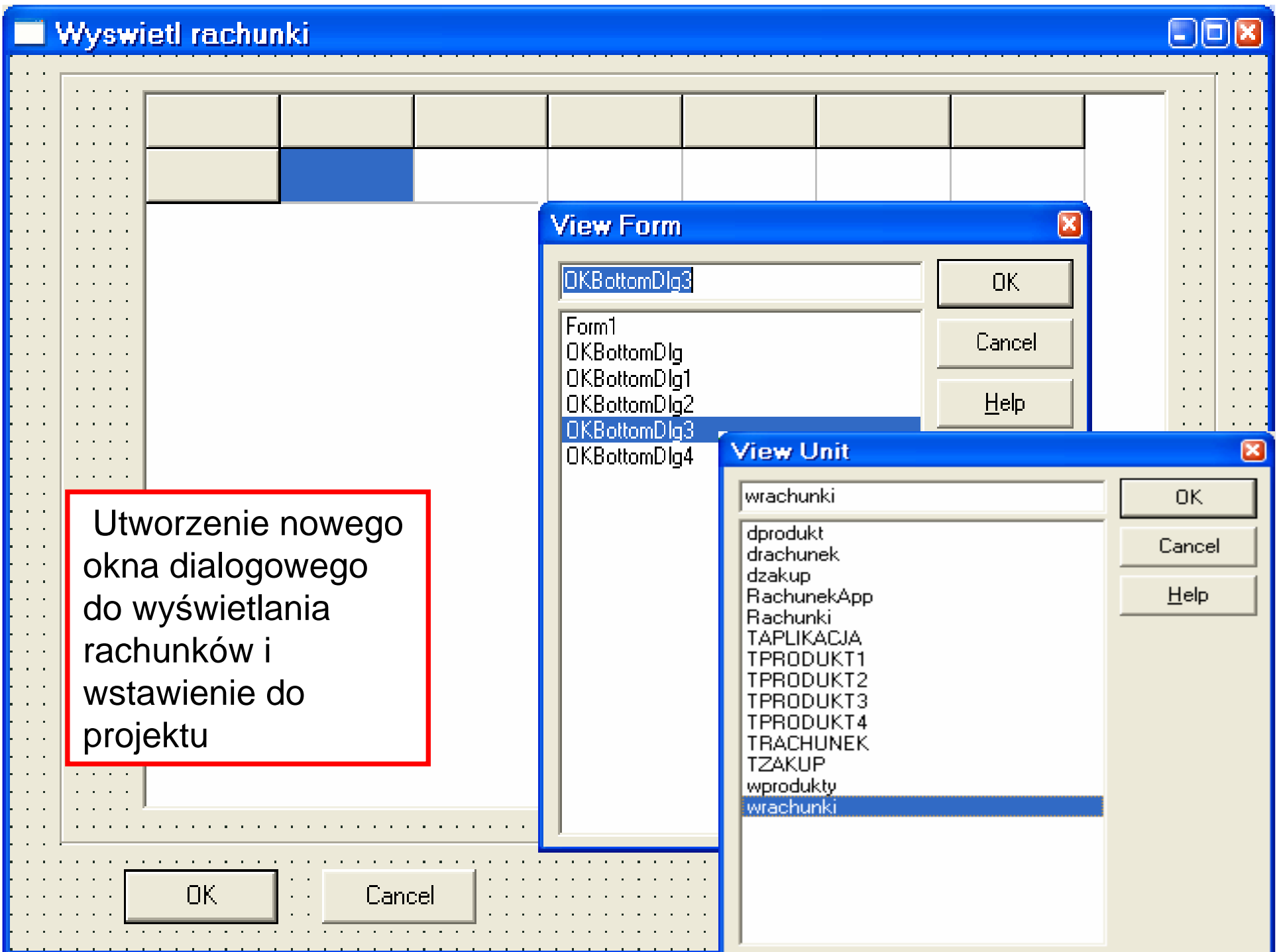
46: 73

Modified

Insert

Uruchomiona aplikacja – wprowadzanie nowego rachunku





```
#ifndef wrachunkiH
#define wrachunkiH

//-----
#include <vc1\System.hpp>
#include <vc1\Windows.hpp>
#include <vc1\SysUtils.hpp>
#include <vc1\Classes.hpp>
#include <vc1\Graphics.hpp>
#include <vc1\StdCtrls.hpp>
#include <vc1\Forms.hpp>
#include <vc1\Controls.hpp>
#include <vc1\Buttons.hpp>
#include <vc1\ExtCtrls.hpp>
#include <Grids.hpp>
#include "RachunekApp.h"

//-----
class TOKBottomDlg3 : public TForm
{
    __published:
        TButton *OKBtn;
        TButton *CancelBtn;
        TBevel *Bevel1;
        TStringGrid *StringGrid1;
private: //recznie -wypełnianie pustymi lancuchami nie wykorzystanych komorek
    void __fastcall TOKBottomDlg3::zeruj(int i, int j);
public: bool __fastcall Execute(); //recznie - obsługa okienka dialogowego
    virtual __fastcall TOKBottomDlg3(TComponent* AOwner);
};
```

Plik nagłówkowy
okna dialogowego
do wyświetlania
rachunków

```
bool _fastcall TOKBottomDlg3::Execute()           //ręcznie
{TKol2<TRachunek> rachunki=Form1->aplikacja.Podaj_rachunki();
  int i=0,j;
  TRachunek* rachunek;
  TZakup* zakup; TProdukt1* produkt;
  char tab[10];
  rachunki.Zeruj(); //przejscie w kolekcji rachunkow na poczatek
  while (rachunki.Koniec()==0)
  { j=0;      //zmienna do numerowania pozycji rachunku
    i++;     //zmienna do liczenia rowkow tabeli
    StringGrid1->RowCount=i+1; //dodanie nowego rowka
    rachunek=rachunki.Podaj_nast();
    StringGrid1->Cells[0][i]="Nr rach: ";
    StringGrid1->Cells[1][i]=itoa(rachunek->Podaj_numer(),tab,10);
    zeruj(i,2); //wypełnianie pustymi lancuchami komorek nie wykorzystanych
    i++;
    StringGrid1->RowCount=i+1; //dodanie nowego rowka
    StringGrid1->Cells[0][i]="L.p.";
    StringGrid1->Cells[1][i]="Nazwa";
    StringGrid1->Cells[2][i]="Cena";
    StringGrid1->Cells[3][i]="Podatek";
    StringGrid1->Cells[4][i]="Promocja";
    StringGrid1->Cells[5][i]="Ilosc";
    StringGrid1->Cells[6][i]="Wartosc";
    TKol2<TZakup> zakupy=rachunek->Podaj_zakupy();
    zakupy.Zeruj(); //przejscie w kolekcji zakupow wybranego rachunku na poczatek
```

Definicja metody `Execute()` do wypełniania komponentu typu `TStringGrid` zawartością kolekcji `rachunki`

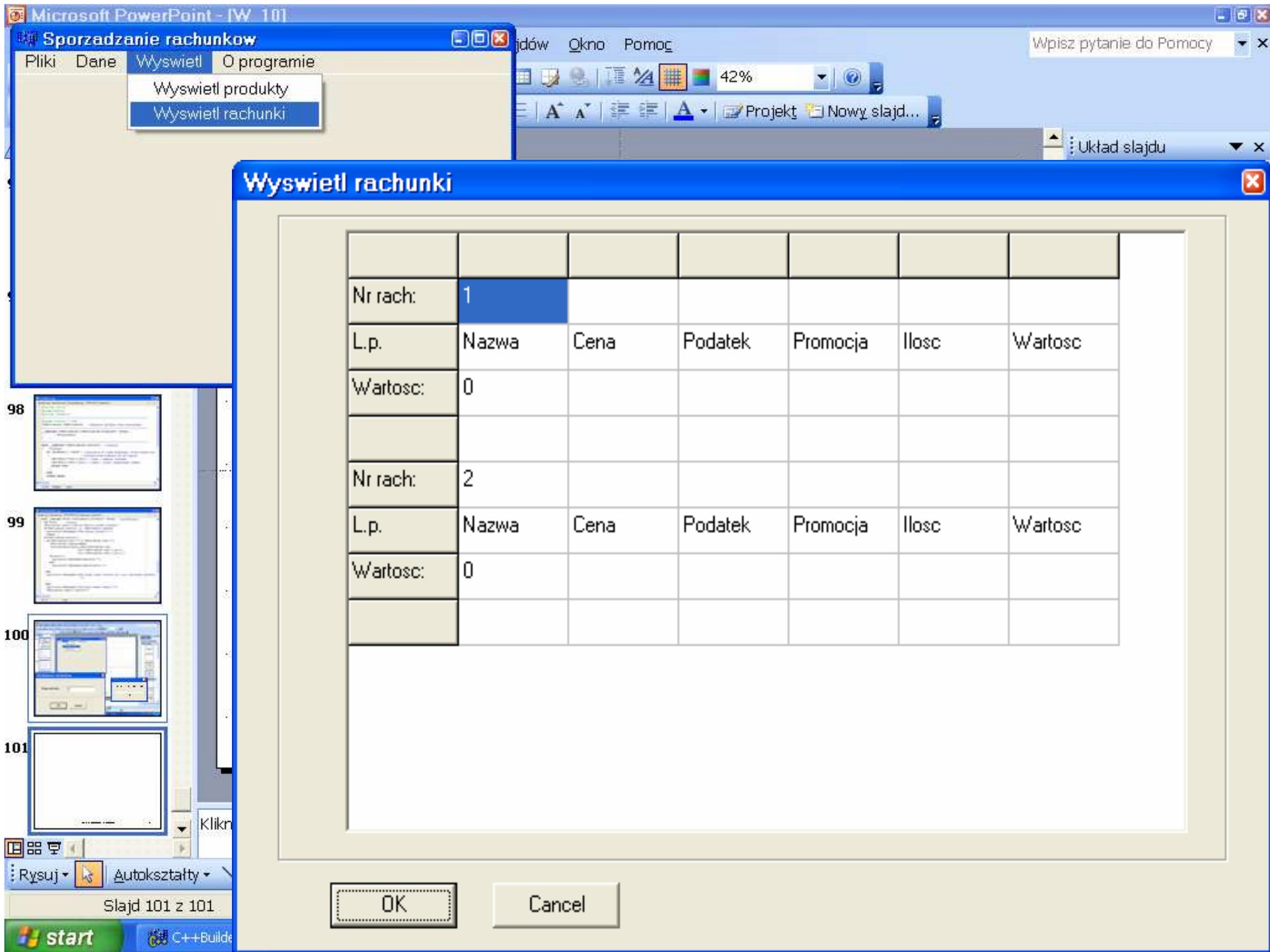
```
while (zakupy.Koniec () == 0)
{
    zakup = zakupy.Podaj_nast ();
    produkt = zakup->Podaj_produkt ();
    i++;           //numer rowka
    StringGrid1->RowCount = i+1; //dodanie nowego rowka
    j++;           //numer pozycji rachunku
    StringGrid1->Cells[0][i] = j;
    StringGrid1->Cells[1][i] = produkt->Podaj_nazwe ().c_str ();
    StringGrid1->Cells[2][i] = gcvt (produkt->Podaj_cene (), 3, tab);
    float pom = produkt->Podaj_podatek ();
    if (pom == -1) //jesli brak podatku wyswietla sie pusty lancuch
        StringGrid1->Cells[3][i] = "";
    else
        StringGrid1->Cells[3][i] = gcvt (produkt->Podaj_podatek (), 3, tab);
    pom = produkt->Podaj_promocje ();
    if (pom == -1) //jesli brak promocji wyswietla sie pusty lancuch
        StringGrid1->Cells[4][i] = "";
    else
        StringGrid1->Cells[4][i] = gcvt (produkt->Podaj_promocje (), 3, tab);
    StringGrid1->Cells[5][i] = itoa (zakup->Podaj_ilosc (), tab, 10);
    StringGrid1->Cells[6][i] = gcvt (zakup->Podaj_wartosc (), 3, tab);
}
```

```
wrachunki.cpp  
dprodukt.cpp drachunek.cpp RachunekApp.cpp wrachunki.cpp  
StringGrid1->RowCount=i+1; //dodanie nowego rowka  
StringGrid1->Cells[0][i]="Wartosc: ";  
StringGrid1->Cells[1][i]= gcvt(rachunek->Podaj_wartosc(),3,tab);  
zeruj(i,2); //wypełnianie pustymi lancuchami komorek nie wykorzystanych  
i++;  
StringGrid1->RowCount=i+1; //dodanie nowego rowka  
zeruj(i,0); //i wypełnienie pustymi lancuchami  
}  
if (rachunki.Pusta()==0)  
    StringGrid1->Visible=true; //jesli sa dane w tabeli, bedzie wyswietlana  
else  
    StringGrid1->Visible=false; // w przeciwnym wypadku jest ukryta  
ShowModal(); //przejscie do trybu modalnego, który konczy sie nacisnieciem  
return true; //klawisza OK lub Cancel;  
}
```

83: 18 Modified Insert

```
#include <vcl.h>
#pragma hdrstop
#include "wrachunki.h"
//-----
#pragma resource "*.dfm"
TOKBottomDlg3 *OKBottomDlg3; //wskaznik globalny okna dialogowego
//-----
__fastcall TOKBottomDlg3::TOKBottomDlg3 (TComponent* AOwner)
    : TForm(AOwner)
{
}
//-----
void __fastcall TOKBottomDlg3::zeruj(int i, int j) //ręcznie
{
    // wypelnianie pustymi lancuchami komorek nie wykorzystanych
    for (; j<=6;j++)
        StringGrid1->Cells[j][i]="";
}
```

```
void __fastcall TForm1::Wyswietlrachunek2Click(TObject *Sender) //automatycznie
{
    OKBottomDlg3->Execute(); //recznie
}
```

VI Wstawienie do projektu okna dialogowego do wstawiania zakupów

dzakup

OK

OKBottomDlg4

OK

dprodukt
drachunek

Cancel

Form1

Cancel

dzakup

Help

OKBottomDlg

Help

RachunekApp

OKBottomDlg1

Rachunki

OKBottomDlg2

TAPLIKACJA

OKBottomDlg3

TPRODUKT1

OKBottomDlg4

TPRODUKT2

TPRODUKT3

TPRODUKT4

TRACHUNEK

TZAKUP

wprodukty

wrachunki

Dodawanie zakupow

Numer rachunku

Ilosc produktu

OK

Cancel

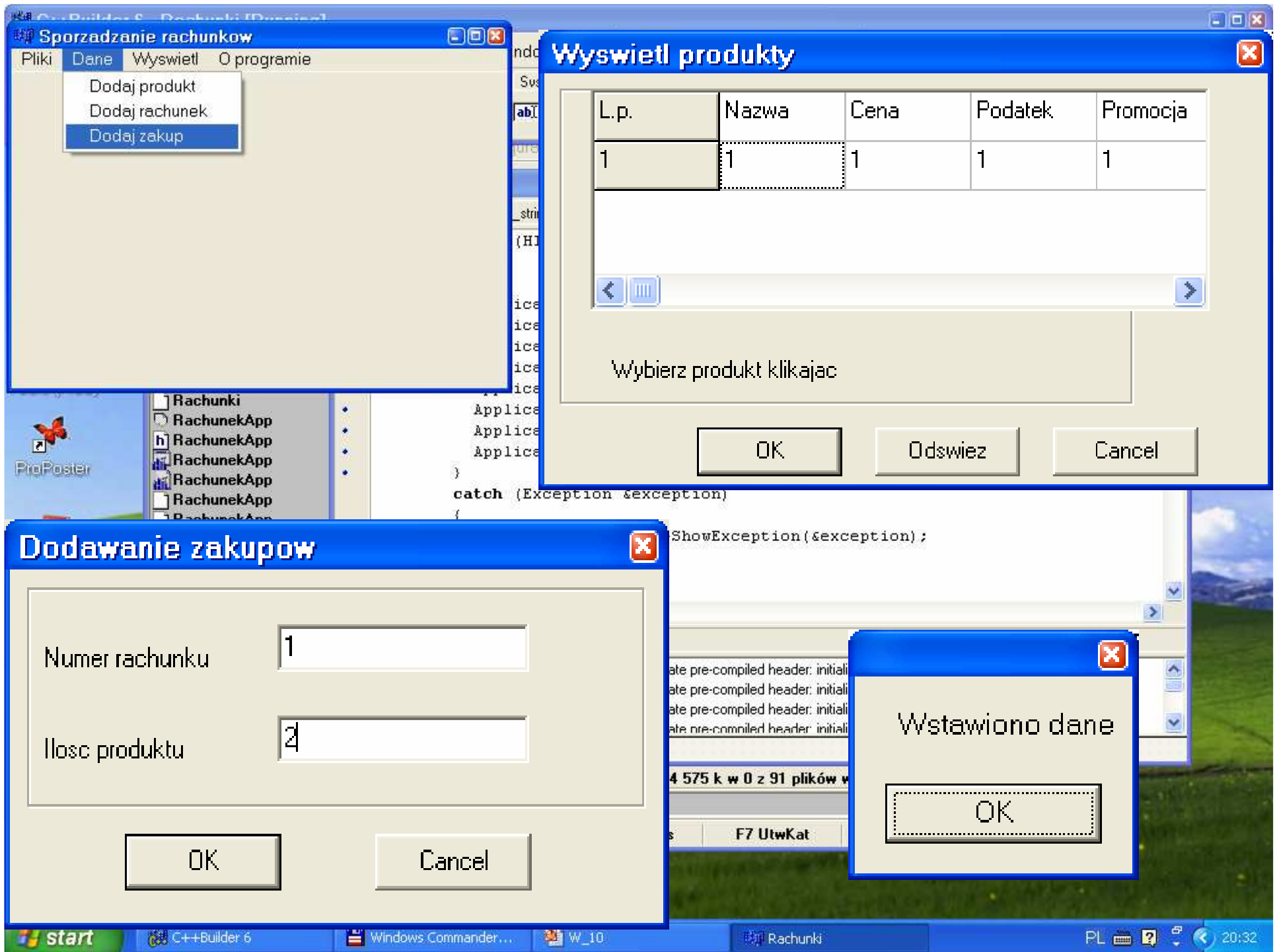
```
#define dzakupH
//-----
#include <vcl\System.hpp>
#include <vcl\Windows.hpp>
#include <vcl\SysUtils.hpp>
#include <vcl\Classes.hpp>
#include <vcl\Graphics.hpp>
#include <vcl\StdCtrls.hpp>
#include <vcl\Forms.hpp>
#include <vcl\Controls.hpp>
#include <vcl\Buttons.hpp>
#include <vcl\ExtCtrls.hpp>
#include "RachunekApp.h"
//-----
class TOKBottomDlg4 : public TForm
{
    __published:
        TButton *OKBtn;
        TButton *CancelBtn;
        TBevel *Bevel1;
        TEdit *Edit1;
        TEdit *Edit2;
        TLabel *Label1;
        TLabel *Label2;
    private:
    public: string tab1, tab2; //dane przekazywane do formularza
        //z numerem rachunku i iloscia towaru
        bool __fastcall Execute(); //recznie - obsluga okienka dialogowego
        virtual __fastcall TOKBottomDlg4(TComponent* AOwner);
};
//-----
extern PACKAGE TOKBottomDlg4 *OKBottomDlg4; //deklaracja globalnego wskaznika
//-----
#endif
```

```
#include <vcl.h>
#pragma hdrstop
#include "dzakup.h"
//-----
#pragma resource "*.dfm"
TOKBottomDlg4 *OKBottomDlg4; //wskaznik globalny okna dialogowego
//-----
__fastcall TOKBottomDlg4::TOKBottomDlg4(TComponent* AOwner)
    : TForm(AOwner)
{
}
//-----
bool __fastcall TOKBottomDlg4::Execute() //recznie
{
    //recznie
    if (ShowModal() ==mrOk ) //przejscie do trybu modalnego, ktory konczy sie
        { //nacisnieciem klawisza OK lub Cancel;
            tab1=Edit1->Text.c_str(); //dane o numerze rachunku
            tab2=Edit2->Text.c_str(); //dane o ilosci zakupionego towaru
            return true;
        }
    else
        return false;
}
```

**Definicja metody `Execute()`
pobierającej numer rachunku oraz
ilość zakupionego produktu**

```
• void __fastcall TForm1::Dodajzakup1Click(TObject *Sender) //automatycznie
• { int wynik; //recznie
• OKBottomDlg2->Label1->Caption="Wybierz produkt klikajac";
• if (OKBottomDlg2->Execute() && !OKBottomDlg2->wybrany)
• { Application->MessageBox("Nie wybrano produktu", "");
• return; }
• if (OKBottomDlg4->Execute()
• { if (OKBottomDlg4->tab1!="" && OKBottomDlg4->tab2!="")
• { OKBottomDlg2->wybrany=false;
• wynik=aplikacja.Wstaw_zakup(OKBottomDlg2->tab,
• atoi((OKBottomDlg4->tab2).c_str()),
• atoi((OKBottomDlg4->tab1).c_str()));
•
• if (wynik==1)
• Application->MessageBox(napisy[6], "");
• else
• Application->MessageBox(napisy[wynik], "");
•
• }
• else
• Application->MessageBox("Nie podano numeru rachunku lub ilosci produktu",
• "");
•
• }
• else
• Application->MessageBox("Anulowano dodanie zakupu", "");
• OKBottomDlg2->Label1->Caption="";
•
• }
```

- Definicja metody po kliknięciu na „Dodaj zakup”, która pobiera kliknięty wiersz z produktem z okien dialogowych:
- wyświetlający produkty
- wstawiający numer rachunku i ilość produktu



Wyświetl rachunki



Nr rach:	1					
L.p.	Nazwa	Cena	Podatek	Promocja	Ilość	Wartość
1	1	1	1	1	2	2
Wartość:	2					
Nr rach:	2					
L.p.	Nazwa	Cena	Podatek	Promocja	Ilość	Wartość
Wartość:	0					

OK

Cancel