

IDŹ DO

PRZYKŁADOWY ROZDZIAŁ



SPIIS TREŚCI

KATALOG KSIĄŻEK

KATALOG ONLINE

ZAMÓW DRUKOWANY KATALOG

TWÓJ KOSZYK

DODAJ DO KOSZYKA

CENNIK I INFORMACJE

ZAMÓW INFORMACJE
O NOWOŚCIACH

ZAMÓW CENNIK

CZYTEL尼亚

FRAGMENTY KSIĄŻEK ONLINE

AutoCAD. Automatyzacja zadań grafiki za pomocą Delphi

Autorzy: Wojciech Czyżycki, Edward Lisowski

ISBN: 83-7197-682-8

Format: B5, stron: 254

Zawiera CD-ROM



Obecnie dokumentacja techniczna jest używana i przetwarzana najczęściej w formie elektronicznej. Do zapisu konstrukcji stosuje się różnorodne programy CAD. Możliwości wyboru jednego spośród nich są duże, jednakże dla osiągnięcia wydajnej pracy programy te trzeba odpowiednio dostosować do określonych potrzeb. Jednym ze sposobów takiego dostosowania jest opracowanie specjalistycznych aplikacji pozwalających na automatyzację procesu projektowania, np. w AutoCAD-zie. Program ten ma wbudowany interfejs automatyzacji OLE, który pozwala oddziaływać w sposób programowy na proces tworzenia rysunku.

Książka powstała w wyniku opracowania przez autorów (na potrzeby własne i innych użytkowników) aplikacji do wspomagania projektowania w AutoCAD-zie. Jest ona adresowana do inżynierów, studentów i innych użytkowników tego programu. Książka zawiera wiele kodów, które można bezpośrednio uruchomić i wykorzystać w swojej pracy projektowej. Użycie większości kodów znajdujących się na płycie CD wymaga jedynie przeprowadzenia instalacji. Użytkownik posiadający pewne umiejętności programowania w Delphi może je rozbudować lub adaptować do swoich celów.

Nawet zaawansowane aplikacje przedstawione w książce są łatwe do uruchomienia i bezpośredniego użycia. Bardzo przydatne dla projektanta mogą być programy do automatycznego rysowania połączeń wpustowych w połączeniu z automatycznymi obliczeniami, wspomagające rysowanie oznaczeń połączeń spawanych czy automatyzujące generowanie elementów połączeń śrubowych.

Książka adresowana jest do inżynierów, studentów i innych użytkowników posiadających wiedzę na temat programu AutoCAD. Użycie wielu programów załączonych na płycie CD wymaga jedynie przeprowadzenia instalacji. Praca zawiera kod wielu programów do bezpośredniego wykorzystania. Użytkownik posiadający pewne umiejętności programowania w Delphi może programy te rozbudować lub adaptować do swoich celów.

W podręczniku zamieszczono te fragmenty kodów programów, które wymagają wyjaśnień (komentarza), kompletny kod przedstawionych programów znajduje się na załączonej do pracy płycie CD.

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl



Spis treści

Przedmowa	5
Rozdział 1. Wstęp	7
Automatyzacja.....	7
Struktura klienta automatyzacji.....	8
Klasy i obiekty	9
Rozdział 2. Interfejs automatyzacji OLE programu AutoCAD 2000	11
Obiekty interfejsu automatyzacji OLE.....	13
Kolekcja Blocks	13
Obiekt Block	14
Obiekt ModelSpace	15
Nawiązanie połączenia z serwerem OLE i ustawienia początkowe	17
Translacja współrzędnych.....	18
Metody generowania podstawowych elementów rysunku.....	18
Wybrane metody modyfikacji obiektów	38
Wybrane składniki obiektu Document i kolekcji Documents	48
Kolekcja Dictionaries, obiekty Dictionary i XRecord	53
Kolekcja DimStyles i obiekt DimStyle	57
Kolekcja Layers i obiekt Layer	58
Kolekcja Linetypes i obiekt Linetype	60
Kolekcje Groups i SelectionSets oraz obiekty Group i SelectionSet.....	61
Kolekcja TextStyles i obiekt TextStyle.....	65
Kolekcja UserCoordinateSystems i obiekt UCS.....	68
Obiekt Utility	69
Ustalanie aktywnych właściwości obiektu Document.....	75
Paski menu i paski narzędziowe	76
Kolekcja MenuGroups i obiekt MenuGrup.....	76
Kolekcje MenuBar i PopUpMenu oraz obiekt PopUpMenuItem	76
Kolekcja Toolbars oraz obiekty Toolbar i ToolbarItem.....	78
Rozdział 3. Przykłady aplikacji	83
Przygotowanie dokumentu — dodawanie warstw i typów linii	83
Generowanie składników rysunkowych	86
Rysowanie otworów według zadanych współrzędnych.....	86
Umieszczanie składników na ustalonej warstwie	89
Bloki składników.....	92
Atrybuty bloków	97
Kreskowania.....	98
Działania na obiektach rysunku	103
Rysunki 3D.....	109

Linia śrubowa — krzywe sklepane	117
Generowanie powierzchni 3D	121
Rozdział 4. Zdarzenia	127
Dostęp do obsługi zdarzeń — biblioteki typów	127
Importowane biblioteki typów	127
Nawiązanie połączenia z obiektami interfejsu OLE przy użyciu modułu importowego	129
Obsługa zdarzeń programu AutoCAD	130
Rozdział 5. Aplikacje dla mechaników	139
Oznaczenia połączeń spawanych	139
Elementy połączeń śrubowych	157
Obliczenia i rysowanie połączenia wpustowego	179
Dobór i generowanie rysunków profili zamkniętych prostokątnych na podstawie obliczeń wytrzymałościowych	190
Przetwarzanie informacji zapisanych w tabelce rysunku	194
Automatyzacja rysowania schematów pneumatycznych i hydraulicznych	196
Literatura	201
Polskie Normy	202
Dodatek A Wykaz metod, zdarzeń i właściwości interfejsu automatyzacji OLE programu AutoCAD 2000	203
Dodatek B Programy na płycie CD	24
Skorowidz	251

Rozdział 5.

Aplikacje dla mechaników

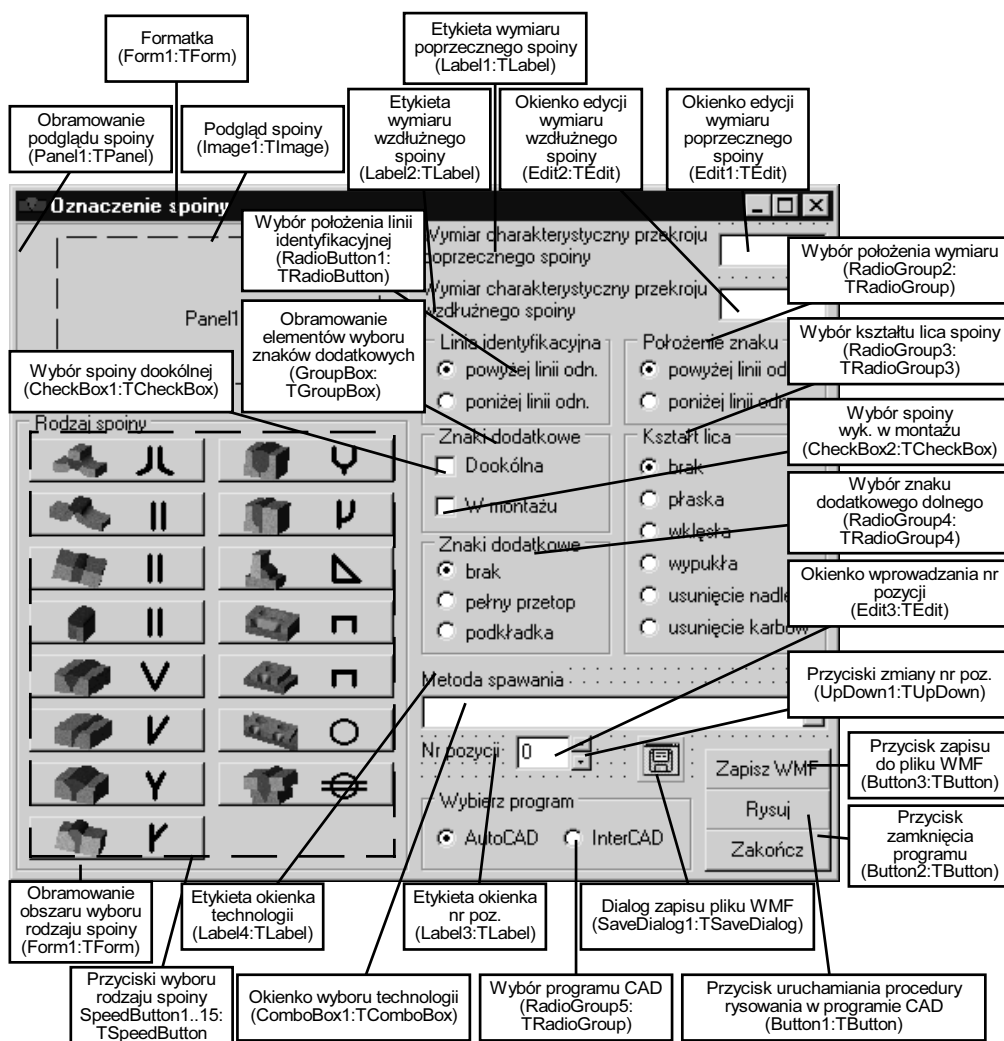
W tym rozdziale przedstawiono zaawansowane programy wykorzystujące techniki programowania obiektowego, korzystające z klas i dziedziczenia, posługujące się relacyjnymi bazami danych i językiem *SQL*. Podjęte zadania dotyczą automatyzacji:

- ♦ oznaczania i generowania symboli połączeń spawanych,
- ♦ połączeń śrubowych przy zastosowaniu dziedziczenia do opisu rodzin elementów,
- ♦ obliczenia i rysowania połączenia wpustowego,
- ♦ doboru i generowania rysunków profili zamkniętych prostokątnych na podstawie obliczeń wytrzymałościowych, wyznaczanie linii wpływowej ugięć dla dobranej belki,
- ♦ przetwarzania informacji zapisanych w tabelce rysunku z automatycznym przesłaniem ich do arkusza kalkulacyjnego MS Excel,
- ♦ rysowania schematów pneumatycznych i hydraulicznych.

Oznaczenia połączeń spawanych

Przy projektowaniu konstrukcji spawanych niezbędne jest określenie wielu parametrów dla złącza oraz narysowanie odpowiedniego symbolu na rysunku. W konstrukcji, w której występuje wiele różnego typu spoin, zautomatyzowanie procesu opisu złącza spawanego może znacznie ułatwić wykonanie dokumentacji rysunkowej. Symbol oznaczenia buduje się z różnych składników zdefiniowanych w przedmiotowej normie. Zautomatyzowanie opisu spoin zrealizowano programowo w ten sposób, że użytkownik w oknie dialogowym wybiera właściwości i parametry spoiny. W trakcie tej czynności w oknie pojawia się automatycznie rysunek symbolu spoiny z oznaczeniami, które użytkownik wprowadza. Po interaktywnym uzgodnieniu oznaczenia złącza spawanego skomponowany symbol jest generowany w programie AutoCAD we wskazanym miejscu na aktywnym rysunku. Symbole oznaczeń mogą być również wyeksportowane do pliku *wmf*. Program nazwano *PolSpaw*, formatkę tego programu przedstawiono na rysunku 5.1. Umieszczono na niej następujące komponenty:

- ♦ przyciski z ikonami *SpeedButton*,
- ♦ grupy przycisków *RadioGroup*,
- ♦ znaczniki *CheckBox*,
- ♦ okienka edycji *Edit*,
- ♦ przyciski zmiany wartości *UpDown*,



Rysunek 5.1. Formatka programu PolSpaw

- ♦ obramowania grup GroupBox,
- ♦ ramkę Panel,
- ♦ etykiety tekstowe Label,
- ♦ ilustrację Image,
- ♦ dialog zapisu SaveDialog,
- ♦ przyciski bez ikon Button.

Opracowano także wymienione niżej moduły:

- ♦ *OznaczFrm* do obsługi formatki i interaktywnego ustawiania właściwości spoiny,

- ♦ *SpoinaSpaw* do przechowywania ustawień spoiny i aktualizacji podglądu oznaczenia na formatce,
- ♦ *RysSpoina* do zgromadzenia procedur rysowania oznaczenia spoiny w programach CAD.

Komponentom zgromadzonym na formatce programu *PolSpaw* nadano odpowiednie ustawienia. Tabela 5.1 zawiera zestawienie tych ustawień, które są różne od ustawień domyślnych komponentów.

Tabela 5.1.

Ustawienia komponentów programu *PolSpaw*

Obiekt	Właściwości		Zdarzenia	
	Nazwa	Wartość	Nazwa	Metoda
Form1	Caption	Oznaczenie spoiny	OnCreate	FormCreate
	BorderStyle	bsSingle	OnDestroy	FormDestroy
	biMaximize	False		
	Autosize	False		
Panel1	BevelOuter	byLowered		
Image1	Align	alClient		
Label1	Caption	Wymiar charakterystyczny przekroju poprzecznego spoiny		
Label2	Caption	Wymiar charakterystyczny przekroju wzdłużnego spoiny		
Label3	Caption	Nr pozycji		
Label4	Caption	Metoda spawania		
Edit1	Text		OnChange	Edit1Change
	MaxLength	3		
Edit2	Text		OnChange	Edit2Change
	MaxLength	5		
Edit3	Text	0		
UpDown1	Associate	Edit3	OnChange	Edit3Change
	Max	999		
Button1	Caption	Rysuj	OnClick	Button1Click
Button2	Caption	Zakończ	OnClick	Button2Click

Tabela 5.1.

Ustawienia komponentów programu PolSpaw — ciąg dalszy

Obiekt	Właściwości		Zdarzenia	
	Nazwa	Wartość	Nazwa	Metoda
Button3	Caption	Zapisz WMF	OnClick	Button3Click
CheckBox1	Caption	Dookólna		
CheckBox2	Caption	W montażu		
ComboBox1	Text		OnChange	Edit3Change
	Style	csDropDownList		
GroupBox1	Caption	Znaki dodatkowe		
GroupBox2	Caption	Rodzaj spoiny		
RadioGroup1	Caption	Linia identyfikacyjna	OnClick	RadioGroup1Click
	Items.Strings	powyżej linii odn. poniżej linii odn.		
	ItemIndex	0		
RadioGroup2	Caption	Położenie znaku	OnClick	RadioGroup2Click
	Items.Strings	powyżej linii odn. poniżej linii odn.		
	ItemIndex	0		
RadioGroup3	Caption	Kształt lica	OnClick	RadioGroup3Click
	Items.Strings	brak płaska wklęsła wypukła usunięcie nadlewu usunięcie karbów		
	ItemIndex	0		
RadioGroup4	Caption	Znaki dodatkowe	OnClick	RadioGroup4Click
	Items.Strings	brak pełny przetop podkładka		
	ItemIndex	0		

Tabela 5.1.

Ustawienia komponentów programu PolSpaw — ciąg dalszy

Obiekt	Właściwości		Zdarzenia	
	Nazwa	Wartość	Nazwa	Metoda
RadioGroup5	Caption	Wybierz program	OnClick	RadioGroup5Click
	Items.Strings	AutoCAD interCAD		
	ItemIndex	0		
SaveDialog1	DefaultExt	wmf		
	Filter	*.wmf *.wmf		
SpeedButton1	Hint	Spoina brzeżna — przetop całkowity	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton2	Hint	Spoina brzeżna — przetop częściowy	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton3	Hint	Spoina I	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton4	Hint	Spoina grzbietowa	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton5	Hint	Spoina V	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton6	Hint	Spoina 1/2 V	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton7	Hint	Spoina Y	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton8	Hint	Spoina 1/2 Y	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton9	Hint	Spoina U	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton10	Hint	Spoina 1/2 U (J)	OnClick	SpeedButton1Click
	GroupIndex	1		

Tabela 5.1.

Ustawienia komponentów programu PolSpaw — ciąg dalszy

Obiekt	Właściwości		Zdarzenia	
	Nazwa	Wartość	Nazwa	Metoda
SpeedButton11	Hint	Spoina pachwinowa	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton12	Hint	Spoina otworowa podłużna	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton13	Hint	Spoina otworowa okrągła	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton14	Hint	Bezotworowa punktowa	OnClick	SpeedButton1Click
	GroupIndex	1		
SpeedButton15	Hint	Bezotworowa liniowa	OnClick	SpeedButton1Click
	GroupIndex	1		

Obsługa elementów formatki umieszczona jest w module *OznaczFrm*.

```

unit OznaczFrm;
interface
uses //moduły standardowe
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, ComCtrls, Registry, Buttons,
  SpoinySpaw, RysSpoina, ExtDlgs; //moduły użytkownika
const
  NazwaKlucza='Software\PGI PK\Spoiny';
type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    GroupBox1: TGroupBox;
    CheckBox1: TCheckBox;
    CheckBox2: TCheckBox;
    Panel1: TPanel;
    Image1: TImage;
    RadioGroup1: TRadioGroup;
    Label1: TLabel;
    Edit1: TEdit;
    Label2: TLabel;
    Edit2: TEdit;
    RadioGroup2: TRadioGroup;
    Button3: TButton;
    SaveDialog1: TSaveDialog;
    ComboBox1: TComboBox;
    Label3: TLabel;
    Label4: TLabel;
  end;

```

```

Edit3: TEdit;
UpDown1: TUpDown;
RadioGroup3: TRadioGroup;
RadioGroup4: TRadioGroup;
GroupBox2: TGroupBox;
SpeedButton1: TSpeedButton;
SpeedButton2: TSpeedButton;
SpeedButton3: TSpeedButton;
SpeedButton4: TSpeedButton;
SpeedButton5: TSpeedButton;
SpeedButton6: TSpeedButton;
SpeedButton7: TSpeedButton;
SpeedButton8: TSpeedButton;
SpeedButton9: TSpeedButton;
SpeedButton10: TSpeedButton;
SpeedButton11: TSpeedButton;
SpeedButton12: TSpeedButton;
SpeedButton13: TSpeedButton;
SpeedButton14: TSpeedButton;
SpeedButton15: TSpeedButton;
procedure Button2Click(Sender: TObject);
procedure CheckBox1Click(Sender: TObject);
procedure CheckBox2Click(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure RadioGroup1Click(Sender: TObject);
procedure Edit1Change(Sender: TObject);
procedure Edit2Change(Sender: TObject);
procedure RadioGroup2Click(Sender: TObject);
procedure Edit3Change(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure RadioGroup4Click(Sender: TObject);
procedure RadioGroup3Click(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure SpeedButton1Click(Sender: TObject);
procedure FormDestroy(Sender: TObject);
end;
var
  Form1: TForm1;
  Spoina: TSpoina;
  Reg: TRegistry;
implementation
uses PolaczAutoCAD;
{$R *.DFM}
procedure TForm1.Button2Click(Sender: TObject);
begin //zamknięcie programu
  Application.Terminate;
end;
procedure TForm1.CheckBox1Click(Sender: TObject);
begin //włączenie bądź wyłączenie znaku spoiny dookołnej
  Spoina.Dookolna:=CheckBox1.Checked;
end;
procedure TForm1.CheckBox2Click(Sender: TObject);
begin //włączenie bądź wyłączenie znaku spoiny wykonywanej w montażu
  Spoina.Wmontazu:=CheckBox2.Checked;
end;
procedure TForm1.FormCreate(Sender: TObject);
begin
  Spoina:=TSpoina.Create; //utworzenie i inicjalizacja obiektu Spoina
  Spoina.Width:=Image1.Width; //przypisanie obszaru do rysowania
  Spoina.Height:=Image1.Height;

```

```

Spoina.GdzieRysowac:=Image1.Canvas;
ComboBox1.Items.LoadFromFile('MetodySp.lst'); //wyp. listy metod spawania
try //odczytanie ostatnio wybranej spoiny
  Reg:=TRegistry.Create;
  Reg.RootKey:=HKEY_CURRENT_USER;
  Reg.OpenKey(NazwaKlucza,True);
  try
    (FindComponent(Reg.ReadString('Ostatni'))as TSpeedButton).Down:=True;
    SpeedButton1.OnClick(FindComponent(Reg.ReadString('Ostatni')) as
      TSpeedButton);
  except
    SpeedButton1.Down:=True;
    SpeedButton1.OnClick(SpeedButton1);
  end;
finally
  Reg.Free;
end;
Edit1.Text:='a'; Edit2.Text:='l'; //ustawienia początkowe komponentów
Edit1.OnChange(Form1); Edit2.OnChange(Form1);
end;
procedure TForm1.RadioGroup1Click(Sender: TObject);
begin //zmiana położenia linii identyfikacyjnej
  if RadioGroup1.ItemIndex=0 then
    Spoina.GdzieLinID:=True
  else
    Spoina.GdzieLinID:=False;
end;
procedure TForm1.Edit1Change(Sender: TObject);
begin //zmiana wartości wymiaru poprzecznego spoiny
  Spoina.ZnWymPop:=Edit1.Text;
end;
procedure TForm1.Edit2Change(Sender: TObject);
begin //zmiana wartości wymiaru wzdłużnego spoiny
  Spoina.ZnWymWzd:=Edit2.Text;
end;
procedure TForm1.RadioGroup2Click(Sender: TObject);
begin //ustawienie położenia znaku
  if RadioGroup2.ItemIndex=0 then
    Spoina.GdzieZnak:=True
  else
    Spoina.GdzieZnak:=False;
end;
procedure TForm1.Edit3Change(Sender: TObject);
begin //włączanie znaku rozwidlenia i ustawienie nr pozycji i technologii
  if (StrToInt(Edit3.Text)>0) or (ComboBox1.Text<>'') then
    Spoina.Rozwidlenie:=True;
  else
    Spoina.Rozwidlenie:=False;
  Spoina.NrPoz:=StrToInt(Edit3.Text);
  Spoina.Technologia:=Copy(ComboBox1.Text,1,Pos(' ',ComboBox1.Text)-1);
end;
procedure TForm1.Button3Click(Sender: TObject);
begin //zapis symbolu spoiny do pliku wmf
  if SaveDialog1.Execute then
    Spoina.SaveToFile(SaveDialog1.FileName);
end;
procedure TForm1.RadioGroup4Click(Sender: TObject);
begin //ustawienie znaku dodatkowego dolnego

```

```

    Spoina.ZnDodDol:=RadioGroup4.ItemIndex;
end;
procedure TForm1.RadioGroup3Click(Sender: TObject);
begin //ustawienie znaku dodatkowego górnego
    Spoina.ZnDodGor:=RadioGroup3.ItemIndex;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin //inicjowanie połączenia i rysowanie spoiny w programie CAD
    try
        RysSp:=TCADSpoina.Create;
        Application.Minimize;
        if (Not VarIsEmpty(AcadDoc)) then //rysowanie po nawiązaniu połączenia
            RysSp.RysujSpoina;
        finally
            RysSp.Free;
        end;
        Application.Restore;
    end;
procedure TForm1.SpeedButton1Click(Sender: TObject);
begin //podanie rodzaju spoiny na podstawie nazwy wciśniętego przycisku
    if Sender=SpeedButton1 then Spoina.Znak:=1;
    if Sender=SpeedButton2 then Spoina.Znak:=2;
    if Sender=SpeedButton3 then Spoina.Znak:=3;
    if Sender=SpeedButton4 then Spoina.Znak:=4;
    if Sender=SpeedButton5 then Spoina.Znak:=5;
    if Sender=SpeedButton6 then Spoina.Znak:=6;
    if Sender=SpeedButton7 then Spoina.Znak:=7;
    if Sender=SpeedButton8 then Spoina.Znak:=8;
    if Sender=SpeedButton9 then Spoina.Znak:=9;
    if Sender=SpeedButton10 then Spoina.Znak:=10;
    if Sender=SpeedButton11 then Spoina.Znak:=11;
    if Sender=SpeedButton12 then Spoina.Znak:=12;
    if Sender=SpeedButton13 then Spoina.Znak:=13;
    if Sender=SpeedButton14 then Spoina.Znak:=14;
    if Sender=SpeedButton15 then Spoina.Znak:=15;
    Spoina.RysSymbolWmf; //narysowanie symbolu spoiny
end;
procedure TForm1.FormDestroy(Sender: TObject);
var
    i: Integer;
begin
    try //zapisanie zmian do pliku ini
        Reg:=TRegistry.Create;
        Reg.RootKey:=HKEY_CURRENT_USER;
        Reg.OpenKey(NazwaKlucza,True);
        i:=0;
        //znalezienie wciśniętego przycisku i zapamiętanie
        while i < ComponentCount do begin
            if Components[i] is TSpeedButton then
                if (Components[i] as TSpeedButton).Down then begin
                    Reg.WriteString('Ostatni',Components[i].Name);
                    i:=ComponentCount;
                end;
            Inc(i);
        end;
    finally
        Reg.Free;
        Spoina.Free;
    end;
end;

```

```
end;  
end;  
initialization //ustawienie katalogu roboczego  
  SetCurrentDir(ExtractFilePath(Application.ExeName));  
end.
```

Ustawienia uproszczonego symbolu spoiny przechowywane są w polach klasy TSpoina zdefiniowanej w module *SpoinySpaw*. Klasa TSpoina zawiera również metody automatycznego rysowania podglądu symbolu spoiny. Sekcja interface modułu *SpoinySpaw* przedstawia się następująco:

```
unit SpoinySpaw;  
interface  
uses  
  graphics, extctrls, classes, controls;  
type  
  //definicja klasy TSpoina przechowującej ustawienia spoiny oraz  
  //zawierającej metody automatycznie rysujące symbol spoiny w oknie  
  //programu w czasie wybierania przez użytkownika rodzaju spoiny  
  TSpoina=class(TMetafile)  
  protected  
    WmfC:TMetafileCanvas;  
    FLinCl:TColor;  
    FBgCl:TColor;  
    FZnak: Integer;  
    FDookolna: Boolean;  
    FWmontazu: Boolean;  
    x0,y0,xs,ys,yLinID,xZn,yOdITex,h,yZnak,yZnakDod,FontSize: Integer;  
    FDP0: Integer;  
    FGdzielLinID: Boolean;  
    FZnWymPop: String;  
    FZnWymWzd: String;  
    FRozwidlenie: Boolean;  
    FGdzieZnak: Boolean;  
    Fy0Znaku: Integer;  
    FNrPoz: Integer;  
    FTechnologia: String;  
    FZnDodDol: Integer;  
    FZnDodGor: Integer;  
  //metody rysowania symbolu podczas doboru  
    procedure RysZnak;  
    procedure RysDookol;  
    procedure RysWmontazu;  
    procedure RysRozwidlenie;  
    procedure RysLinID;  
    procedure RysZnWymSp;  
    procedure RysNrPozTech;  
    procedure RysZnDodDol;  
    procedure RysZnDodGor;  
  //metody ustawiania właściwości  
    procedure SetZnak(const Value: Integer);  
    procedure SetDookolna(const Value: Boolean);  
    procedure SetWmontazu(const Value: Boolean);  
    procedure SetDPO(const Value: Integer);  
    procedure SetGdzielLinID(const Value: Boolean);  
    procedure SetZnWymPop(const Value: String);  
    procedure SetZnWymWzd(const Value: String);  
    procedure SetRozwidlenie(const Value: Boolean);
```

```

    procedure SetGdzieZnak(const Value: Boolean);
    procedure Sety0Znaku(const Value: Integer);
    procedure SetNrPoz(const Value: Integer);
    procedure SetTechnologia(const Value: String);
    procedure SetZnDodDol(const Value: Integer);
    procedure SetZnDodGor(const Value: Integer);
//procedury pomocnicze
    procedure ArcCR(xC,yC,r,kp,kk:Double);
    procedure PozycjaZnakow;
published
    GdzieRysowac:TCanvas;
    property Znak:Integer read FZnak write SetZnak;
    property Dookolna:Boolean read FDookolna write SetDookolna;
    property Wmontazu:Boolean read FWmontazu write SetWmontazu;
    property ZnDodDol:Integer read FZnDodDol write SetZnDodDol;
    property Rozwidlenie:Boolean read FRozwidlenie write SetRozwidlenie;
    property DPO:Integer read FDPO write SetDPO;
    property GdzieZnak:Boolean read FGdzieZnak write SetGdzieZnak;
    property y0Znaku:Integer read Fy0Znaku write Sety0Znaku;
    property GdzieLinID:Boolean read FGdzieLinID write SetGdzieLinID;
    property ZnWymPop:String read FZnWymPop write SetZnWymPop;
    property ZnWymWzd:String read FZnWymWzd write SetZnWymWzd;
    property NrPoz:Integer read FNrPoz write SetNrPoz;
    property Technologia:String read FTechnologia write SetTechnologia;
    property ZnDodGor:Integer read FZnDodGor write SetZnDodGor;
    procedure RysSymbolWmf;
public
    constructor Create;override;
end;
implementation
...
end.

```

Procedury rysowania symbolu spoiny w programie AutoCAD zgromadzone są w klasie TCADSpoina zdefiniowanej w module *RysSpoina*.

```

unit RysSpoina;
//procedury rysowania symbolu spoiny dla programów AutoCAD
//w tym module można dopisać procedury rysowania symboli dla innych
//programów CAD wyposażonych w interfejs OLE Automaion
interface
uses
    PolaczAutoCAD, AcadConst, WarstwyLinie;
type
//Klasa TCADSpoina zawiera metody rysowania symbolu spoiny w programach CAD
TCADSpoina=class
protected
    h,x0,y0,x0s,y0s,l,yLinID,yZnak,y0Znaku,yTech,yPoz,xZn,
        yZnakDod,Kat:Double;
    P,W,Kr,KrID,KrZnak:Integer;
    AcadObj:OLEVariant;
    procedure WyznaczPoloz;
    procedure RysSymbolAcad;
    procedure RysStrzałkaAcad;
    procedure RysZnakAcad;
    procedure RysZnakDodDolAcad;
    procedure RysZnakDodGorAcad;
public

```

```

        constructor Create;
        destructor Free;
    published
        procedure RysujSpoina;
    end;
var
    RysSp: TCADSpoina;
implementation
uses
    Dialogs, Math, SysUtils, Forms, OznacFrm;
const
    NazwaWarstwy = 'OLESpoiny';
constructor TCADSpoina.Create; //tworzenie obiektu i ustawienia początkowe
begin
    P:=2; W:=-1; h:=3.5; yLinID:=1/3*h;
    yZnak:=h; xZn:=2*h+0.71*h*Length(Spoina.ZnWymPop);
    yZnakDod:=yLinID;
    if Spoina.GdzieLinID then
        KrID:=1
    else
        KrID:=-1;
    if Spoina.GdzieZnak then
        KrZnak:=1
    else
        KrZnak:=-1;
    //linia identyfikacyjna i znak po przeciwnych stronach
    if Spoina.GdzieZnak=Spoina.GdzieLinID then begin
        y0Znaku:=KrZnak*(yLinID+0.1*h);
        yZnakDod:=0;
    end
    else begin
        y0Znaku:=KrZnak*0.1*h; //aby tekst nie zlewał się z linią
        yZnakDod:=KrID*yLinID;
    end;
    //położenie oznaczenia technologii wykonania
    if (Spoina.Technologia<>'') and (Spoina.NrPoz>0) then begin
        yTech:=-h; yPoz:=h;
    end
    else begin
        yTech:=0; yPoz:=0;
    end;
    //połączenie z programem
    if PolaczZAcad then begin //jeśli nawiązano połączenie
        ZapamietajUstawienia;
        DefiniujWarstwe;
        WczytajTypyLinii;
        try //sprawdzenie czy dodawana warstwa istnieje w dokumencie
            if not LayIstnieje(NazwaWarstwy) then //jeśli nie ma będzie dodana
                AcadDoc.Layers.Add(NazwaWarstwy);
            //ustawienie aktywnej warstwy
            AktywnyElement:=AcadDoc.Layers.Item(NazwaWarstwy);
            AcadDoc.ActiveLayer:=AktywnyElement;
            AcadDoc.ActiveLayer.Color:=acRed;
        except
            end;
    end;
end;
destructor TCADSpoina.Free;

```

```

begin //niszczenie obiektu i przywracanie ustawień
  if not VarIsEmpty(AcadDoc) then
    PrzywrocUstawienia; //jeśli połączenie z AutoCADem było nawiązane
    inherited Destroy;
  end;
procedure TCADSpoina.WyznaczPoloz;
begin //obliczenie położenia półki symbolu spoiny
  if x0>x0s then begin //kierunek półka oznaczenia w prawo
    Kr:=1; xZn:=xZn;
  end
  else begin //kierunek półka oznaczenia w lewo
    Kr:=-1; xZn:=Kr*(1-xZn);
  end;
  yZnak:=KrZnak*yZnak;
//wyznaczenie kąta linii wskazującej spoinę
  Kat:=arccos((x0-x0s)/sqrt(sqr(x0-x0s)+sqr(y0-y0s)));
  if y0s>y0 then //3 i 4 ćwiartka
    Kat:=2*Pi-Kat;
end;
procedure TCADSpoina.RysujSpoinę;
begin //wywołanie odpowiedniej procedury rysowania symbolu
  RysSymbolAcad;
end;
procedure TCADSpoina.RysSymbolAcad;
var
  WskazanyPkt:OLEVariant;
begin //rysowanie symbolu spoiny
  try //niektóre zmienne zostaną zdefiniowane z uwagi na szerokość znaków
    l:=4*h+h*(Length(Spoina.ZnWymPop)+Length(Spoina.ZnWymWzd));
    xZn:=2*h+h*Length(Spoina.ZnWymPop);
    WskazanyPkt:=VarArrayCreate([0,2],varDouble); //pobieranie punktów
    WskazanyPkt:=AcadDoc.Utility.GetPoint('Wskaż położenie strzałki');
    x0s:=WskazanyPkt[0]; y0s:=WskazanyPkt[1];
    WskazanyPkt:=AcadDoc.Utility.GetPoint('Wskaż położenie półki symbolu');
    x0:=WskazanyPkt[0]; y0:=WskazanyPkt[1];
    WyznaczPoloz;
    AcadDwg.AddLine(p2D(x0s,y0s),p2D(x0,y0));
    RysStrzałkaAcad; //rysowanie strzałki
    AcadDwg.AddLine(p2D(x0,y0),p2D(x0+Kr*1,y0)); //półka
  //linia identyfikacyjna
    AcadObj:=AcadDwg.AddLine(p2D(x0+Kr*1/2*h,y0+KrID*yLinID),
      p2D(x0+Kr*1,y0+KrID*yLinID));
    AcadObj.LineType:='HIDDEN';
    RysZnakAcad; //znak umowny spoiny
  //wymiary spoiny
    if KrZnak>0 then begin //znak pod półką
      AcadDwg.AddText(Spoina.ZnWymPop,p2D(x0+xZn-h*
        (Length(Spoina.ZnWymPop)+1.5),y0+1.1*y0Znaku),h);
      AcadDwg.AddText(Spoina.ZnWymWzd,p2D(x0+xZn+2*h,y0+1.1*y0Znaku),h);
    end
    else begin //znak pod półką
      AcadDwg.AddText(Spoina.ZnWymPop,p2D(x0+xZn-h*
        (Length(Spoina.ZnWymPop)+1.5),y0+1.1*y0Znaku-h),h);
      AcadDwg.AddText(Spoina.ZnWymWzd,p2D(x0+xZn+h*
        (Length(Spoina.ZnWymPop)-1.5),y0+1.1*y0Znaku-h),h);
    end;
    RysZnakDodDo1Acad; //znak dodatkowy dolny
    RysZnakDodGorAcad; //znak dodatkowy gorny
  end;
end;

```

```

if Spoina.Dookolna then //spoina dookołna
  AcadDwg.AddCircle(p2D(x0,y0),1/3*h);
if Spoina.WMontazu then begin //chorągiewka
  AcadDwg.AddLine(p2D(x0,y0),p2D(x0,y0+2*h));
  AcadDwg.AddLine(p2D(x0,y0+2*h),p2D(x0-Kr*h*cos(Pi/12),
    y0+2*h-h*sin(Pi/6)));
  AcadDwg.AddLine(p2D(x0,y0+h),p2D(x0-Kr*h*cos(Pi/12),
    y0+2*h-h*sin(Pi/6)));
end;
//rozwidlenie linii odniesienia
if (Spoina.Technologia<>'') or (Spoina.NrPoz>0) then begin
  AcadDwg.AddLine(p2D(x0+Kr*1,y0),p2D(x0+Kr*
    (1+h*cos(Pi/4)),y0+h*sin(Pi/4)));
  AcadDwg.AddLine(p2D(x0+Kr*1,y0),p2D(x0+Kr*
    (1+h*cos(Pi/4)),y0-h*sin(Pi/4)));
end;
if Kr>0 then begin //oznaczenie technologii wykonania
  if (Spoina.Technologia<>'') then
    AcadDwg.AddText(Spoina.Technologia,p2D(x0+Kr*(1+h),y0+yTech-h/2),h);
  if Spoina.NrPoz>0 then //nr pozycji
    AcadDwg.AddText(IntToStr(Spoina.NrPoz),p2D(x0+Kr*(1+h),
      y0+yPoz-h),1.4*h);
end
else begin
  if (Spoina.Technologia<>'') then
    AcadDwg.AddText(Spoina.Technologia,p2D(x0+Kr*
      (1+h*Length(Spoina.Technologia)),y0+yTech-h/2),h);
  if Spoina.NrPoz>0 then //nr pozycji
    AcadDwg.AddText(IntToStr(Spoina.NrPoz),p2D(x0+Kr*
      (1+h+h*Length(IntToStr(Spoina.NrPoz))),y0+yPoz-h),1.4*h);
end;
except
  Application.Restore;
  MessageDlg('Przed wskazaniem punktu wywołano polecenie,'+
    #13+'które spowodowało utratę połączenia z serwerem OLE'+#13+
    #13+'Wstawianie elementu należy powtórzyć.',mtError,[mbOK],0);
end;
end;
procedure TCADSpoina.RysStrzałkaAcad;
begin //rysowanie strzałki symbolu spoiny
  AcadDwg.AddLine(p2D(x0s,y0s),p2D(x0s+h/cos(Pi/24)*
    cos(Kat+Pi/24),y0s+h/cos(Pi/24)*sin(Kat+Pi/24)));
  AcadDwg.AddLine(p2D(x0s,y0s),p2D(x0s+h/cos(Pi/24)*
    cos(Kat-Pi/24),y0s+h/cos(Pi/24)*sin(Kat-Pi/24)));
  AcadDwg.AddLine(p2D(x0s+h/cos(Pi/24)*cos(Kat+Pi/24),
    y0s+h/cos(Pi/24)*sin(Kat+Pi/24),p2D(x0s+h/cos(Pi/24)*
    cos(Kat-Pi/24),y0s+h/cos(Pi/24)*sin(Kat-Pi/24)));
end;
procedure TCADSpoina.RysZnakAcad;
//rysowanie znaku umownego rodzaju spoiny
procedure ZnakSpI;
begin //procedura rysowania znaku spoiny I; ten znak występuje kilka razy
  AcadDwg.AddLine(p2D(x0+xZn-h/4,y0+yZnak+y0Znaku),
    p2D(x0+xZn-h/4,y0+y0Znaku));
  AcadDwg.AddLine(p2D(x0+xZn+h/4,y0+yZnak+y0Znaku),
    p2D(x0+xZn+h/4,y0+y0Znaku));
end;
procedure ZnakSpPod10tw;

```

```

begin //procedura rysowania znaku spoiny otworowej
  AcadDwg.AddLine(p2D(x0+xZn-h,y0+y0Znaku),
    p2D(x0+xZn-h,y0+y0Znaku+yZnak));
  AcadDwg.AddLine(p2D(x0+xZn-h,y0+y0Znaku+yZnak),
    p2D(x0+xZn+h,y0+y0Znaku+yZnak));
  AcadDwg.AddLine(p2D(x0+xZn+h,y0+y0Znaku+yZnak),
    p2D(x0+xZn+h,y0+y0Znaku));
  AcadDwg.AddLine(p2D(x0+xZn-h,y0+y0Znaku),p2D(x0+xZn+h,y0+y0Znaku));
end;
begin
case Spoina.Znak of
1:begin //Spoina brzeżna z krawędziami podwiniętymi przetopionymi całkow.
  AcadDwg.AddLine(p2D(x0+xZn-0.2*h,y0+yZnak+y0Znaku),
    p2D(x0+xZn-0.2*h,y0+0.5*yZnak+y0Znaku));
  AcadDwg.AddLine(p2D(x0+xZn+0.2*h,y0+yZnak+y0Znaku),
    p2D(x0+xZn+0.2*h,y0+0.5*yZnak+y0Znaku));
  if Spoina.GdzieZnak then begin //łuki w zależności od położenia znaku
    AcadDwg.AddArc(p2D(x0+xZn-0.7*h,y0+y0Znaku+0.5*yZnak),
      0.5*h,3/2*Pi,2*Pi);
    AcadDwg.AddArc(p2D(x0+xZn+0.7*h,y0+y0Znaku+0.5*yZnak),
      0.5*h,3/2*Pi);
  end
  else begin
    AcadDwg.AddArc(p2D(x0+xZn-0.7*h,y0+y0Znaku+0.5*yZnak),
      0.5*h,0,2*Pi);
    AcadDwg.AddArc(p2D(x0+xZn+0.7*h,y0+y0Znaku+0.5*yZnak),
      0.5*h,0,2*Pi);
  end
end;
2:ZnakSpI; //Spoina brzeżna z krawędz podwiniętymi przetopionymi część.
3:ZnakSpI; //Spoina I
4:ZnakSpI; //Spoina grzbietowa
5:begin //Spoina V
  AcadDwg.AddLine(p2D(x0+xZn-h/sqrt(3),y0+yZnak+y0Znaku),
    p2D(x0+xZn,y0+y0Znaku));
  AcadDwg.AddLine(p2D(x0+xZn,y0+y0Znaku),
    p2D(x0+xZn+h/sqrt(3),y0+yZnak+y0Znaku));
end;
6:begin //Spoina 1/2V
  AcadDwg.AddLine(p2D(x0+xZn-h/2,y0+yZnak+y0Znaku),
    p2D(x0+xZn-h/2,y0+y0Znaku));
  AcadDwg.AddLine(p2D(x0+xZn-h/2,y0+y0Znaku),
    p2D(x0+xZn-h/2+h*tan(Pi/4),y0+yZnak+y0Znaku));
end;
7:begin //Spoina Y
  AcadDwg.AddLine(p2D(x0+xZn,y0+1/3*yZnak+y0Znaku),
    p2D(x0+xZn,y0+y0Znaku));
  AcadDwg.AddLine(p2D(x0+xZn-sqrt(3),y0+yZnak+y0Znaku),
    p2D(x0+xZn,y0+y0Znaku+1/3*yZnak));
  AcadDwg.AddLine(p2D(x0+xZn,y0+y0Znaku+1/3*yZnak),
    p2D(x0+xZn+h/sqrt(3),y0+yZnak+y0Znaku));
end;
8:begin //Spoina 1/2Y
  AcadDwg.AddLine(p2D(x0+xZn-h/2,y0+yZnak+y0Znaku),
    p2D(x0+xZn-h/2,y0+y0Znaku));
  AcadDwg.AddLine(p2D(x0+xZn-h/2,y0+y0Znaku+1/3*yZnak),
    p2D(x0+xZn-h/2+h*tan(Pi/4),y0+yZnak+y0Znaku));
end;
9:begin //Spoina U

```

```

AcadDwg.AddLine(p2D(x0+xZn,y0+1/4*yZnak+y0Znaku),
  p2D(x0+xZn,y0+y0Znaku));
if Spoina.GdzieZnak then
  AcadDwg.AddArc(p2D(x0+xZn,y0+yZnak+y0Znaku),3/4*h,Pi,2*Pi)
else
  AcadDwg.AddArc(p2D(x0+xZn,y0+yZnak+y0Znaku),3/4*h,0,Pi);
end;
10:begin //Spoina 1/2U
  AcadDwg.AddLine(p2D(x0+xZn-3/8*h,y0+yZnak+y0Znaku),
    p2D(x0+xZn-3/8*h,y0+y0Znaku));
  if Spoina.GdzieZnak then
    AcadDwg.AddArc(p2D(x0+xZn-3/8*h,y0+yZnak+y0Znaku),
      3/4*h,3/2*Pi,2*Pi)
  else
    AcadDwg.AddArc(p2D(x0+xZn-3/8*h,y0+yZnak+y0Znaku),
      3/4*h,0,Pi/2);
  end;
11:begin //Spoina pachwinowa
  AcadDwg.AddLine(p2D(x0+xZn-h/2,y0+yZnak+y0Znaku),
    p2D(x0+xZn-h/2,y0+y0Znaku));
  AcadDwg.AddLine(p2D(x0+xZn-h/2,y0+y0Znaku),
    p2D(x0+xZn+h/2,y0+y0Znaku));
  AcadDwg.AddLine(p2D(x0+xZn+h/2,y0+y0Znaku),
    p2D(x0+xZn-h/2,y0+yZnak+y0Znaku));
  end;
12: ZnakSpPod10tw; //Spoina podłużna
13: ZnakSpPod10tw; //Spoina otworowa okrągła
14: //Spoina bezotworowa punktowa
  AcadDwg.AddCircle(p2D(x0+xZn,y0+yZnak/2+y0Znaku),h/2);
15:begin //Spoina bezotworowa liniowa
  AcadDwg.AddLine(p2D(x0+xZn-3/4*h,y0+4/6*yZnak+y0Znaku),
    p2D(x0+xZn+3/4*h,y0+4/6*yZnak+y0Znaku));
  AcadDwg.AddLine(p2D(x0+xZn-3/4*h,y0+2/6*yZnak+y0Znaku),
    p2D(x0+xZn+3/4*h,y0+2/6*yZnak+y0Znaku));
  AcadDwg.AddCircle(p2D(x0+xZn,y0+yZnak/2+y0Znaku),h/2);
  end;
end;
end;
procedure TCADSpoina.RysZnakDodDo1Acad;
begin //rysowanie znaku dodatkowego dolnego
case Spoina.ZnDodDo1 of
0:; //brak znaku
1: begin //pełny przetop
  AcadDwg.AddLine(p2D(x0+xZn-h*cos(Pi/6),y0+yZnakDod),
    p2D(x0+xZn+h*cos(Pi/6),y0+yZnakDod));
  if Spoina.GdzieZnak then
    AcadDwg.AddArc(p2D(x0+xZn,y0+yZnakDod+yZnak/2),h,7/6*Pi,11/6*Pi)
  else
    AcadDwg.AddArc(p2D(x0+xZn,y0+yZnakDod+yZnak/2),h,Pi/6,5/6*Pi);
  end;
2: begin //podkładka
  AcadDwg.AddLine(p2D(x0+xZn-h/2,y0+yZnakDod),
    p2D(x0+xZn+h/2,y0+yZnakDod));
  AcadDwg.AddLine(p2D(x0+xZn+h/2,y0+yZnakDod),
    p2D(x0+xZn+h/2,y0+yZnakDod-yZnak/2));
  AcadDwg.AddLine(p2D(x0+xZn+h/2,y0+yZnakDod-yZnak/2),
    p2D(x0+xZn-h/2,y0+yZnakDod-yZnak/2));
  AcadDwg.AddLine(p2D(x0+xZn-h/2,y0+yZnakDod-yZnak/2),
    p2D(x0+xZn-h/2,y0+yZnakDod));
end;

```

```

        p2D(x0+xZn-h/2,y0+yZnakDod));
    end;
end;
end;
procedure TCADSpoina.RysZnakDodGorAcad;
begin //rysowanie znaku dodatkowego górnego
if Spoina.Znak<>11 then //dla spoin innych niż pachwinowa
case Spoina.ZnDodGor of
0:: //brak
1:begin //płaska
    AcadDwg.AddLine(p2D(x0+xZn-h/2,y0+y0Znaku+1.2*yZnak),
        p2D(x0+xZn+h/2,y0+y0Znaku+1.2*yZnak));
    end;
2:begin //wkłęsza
    if Spoina.GdzieZnak then
        AcadDwg.AddArc(p2D(x0+xZn,y0+y0Znaku+7/4*yZnak),h/2,5/4*Pi,7/4*Pi)
    else
        AcadDwg.AddArc(p2D(x0+xZn,y0+y0Znaku+7/4*yZnak),h/2,Pi/4,3/4*Pi);
    end;
3:begin //wypukła
    if Spoina.GdzieZnak then
        AcadDwg.AddArc(p2D(x0+xZn,y0+y0Znaku+3/4*yZnak),h/2,Pi/4,3/4*Pi)
    else
        AcadDwg.AddArc(p2D(x0+xZn,y0+y0Znaku+3/4*yZnak),h/2,
            5/4*Pi,7/4*Pi);
    end;
4:begin //usunięcie nadlewu spoiny równo z powierzchnią brzegu
    AcadDwg.AddLine(p2D(x0+xZn-h/2,y0+y0Znaku+5/4*yZnak),
        p2D(x0+xZn+h/2,y0+y0Znaku+5/4*yZnak));
    AcadDwg.AddCircle(p2D(x0+xZn,y0+y0Znaku+5/4*yZnak+yZnak/4),h/4);
    end;
5:begin //usunięcie podtopień i karbów "kotwica"
    AcadDwg.AddLine(p2D(x0+xZn,y0+y0Znaku+9/4*yZnak),
        p2D(x0+xZn,y0+y0Znaku+6/4*yZnak));
    if Spoina.GdzieZnak then begin
        AcadDwg.AddArc(p2D(x0+xZn+1/4*h,y0+y0Znaku+6/4*yZnak),h/4,Pi,2*Pi);
        AcadDwg.AddArc(p2D(x0+xZn-1/4*h,y0+y0Znaku+6/4*yZnak),h/4,Pi,2*Pi);
    end
    else begin
        AcadDwg.AddArc(p2D(x0+xZn+1/4*h,y0+y0Znaku+6/4*yZnak),h/4,0,Pi);
        AcadDwg.AddArc(p2D(x0+xZn-1/4*h,y0+y0Znaku+6/4*yZnak),h/4,0,Pi);
    end;
end;
end;
end
else //dla spoiny pachwinowej z uwagi na inne umiejscowienie znaku
case Spoina.ZnDodGor of
0:: //brak
1:begin //płaska
    AcadDwg.AddLine(p2D(x0+xZn-(h/2-h/2*cos(Pi/4)),y0+y0Znaku
        +3/2*yZnak*sin(Pi/4)),p2D(x0+xZn-(h/2-3/2*h*cos(Pi/4)),
        y0+y0Znaku+yZnak/2*sin(Pi/4)));
    end;
2:begin //wkłęsza
    if Spoina.GdzieZnak then
        AcadDwg.AddArc(p2D(x0+xZn-h/2+3/2*h*cos(Pi/4),
            y0+y0Znaku+3/2*yZnak*sin(Pi/4)),h/2,Pi,3/2*Pi)
    else
        AcadDwg.AddArc(p2D(x0+xZn-h/2+3/2*h*cos(Pi/4),

```


Elementy połączeń śrubowych

Przy projektowaniu urządzeń często stosuje się połączenia gwintowe. Istnieje wiele ich odmian przystosowanych do różnych zastosowań. Korzystna jest automatyzacja wykonywania rysunków i doboru tych elementów. Mając na uwadze określoną firmę czy też projektowanie konkretnych urządzeń, można programowo przygotować selekcję elementów, preferując ich unifikację. Przedstawiony w tym punkcie przykład dotyczy automatycznego generowania rysunków elementów połączeń śrubowych wraz z atrybutami przechowującymi informacje o ich oznaczeniach.

Przy budowie programu *PolaczGwint* do opisu elementów połączeń śrubowych zastosowano podejście obiektowe z wykorzystaniem *dziedziczenia* i *metod abstrakcyjnych*. Dzięki tym technologiom uzyskano większe możliwości programu w zakresie kompletacji połączenia.

Przedstawiony przykład dotyczy elementów połączeń śrubowych, takich jak:

- ♦ śruby z łbem sześciokątnym z gwintem na całej długości lub częściowym opisanych w normach [33], [32],
- ♦ śruby z łbem walcowym opisane w normie [34],
- ♦ nakrętki zwykłe i zmniejszone opisane w normach [35], [36],
- ♦ podkładki okrągłe zgrubne i dokładne opisane w normach [38], [39],
- ♦ podkładki sprężyste zwykłe i lekkie opisane w normach [40], [41],
- ♦ gwinty zewnętrzne i wewnętrzne opisane w normie [37].

Przedstawiony program może być rozszerzany o inne elementy i modyfikowany do współpracy z różnymi programami CAD.

Program do połączeń śrubowych nazwano *PolaczGwint* i zbudowano go z następujących modułów: *PGwintFrm*, *RysSruby*, *RysNakr*, *RysGwint*, *RysPodkl*, które opracowano do rysowania danego typu elementu.

Formatkę programu *PolaczGwint* przedstawiono na rysunku 5.3. Umieszczono na niej komponent *PageControl*, w którym zdefiniowano odpowiednio cztery zakładki dla wyróżnienia grup elementów. Rysunek 5.3 przedstawia zakładkę dla elementów typu śruba i zgromadzone odpowiednie komponenty, jak *DBGrid*, *RadioGroup*, itp., służące do wyselekcjonowania elementu o określonych wymiarach. Dla pozostałych elementów przygotowano analogiczne zakładki i umieszczono na nich odpowiednie komponenty.

W dolnej części formatki poza obszarem komponentu *PageControl* umieszczono komponenty *RadioGroup* oraz *ComboBox*, *CheckBox* i *Button* wspólne dla wszystkich zakładek, pozwalające na wybór sposobu rysowania elementu.

Wymiary i inne dane elementów zgromadzono w plikach relacyjnych baz danych z rozszerzeniem *db* (Paradox), natomiast dostęp do tych danych zrealizowano za pomocą języka SQL obsługiwanego przez komponent *Query*.

Tabela 5.2.

Ustawienia komponentów umieszczonych na zakładce Śruby formatki programu PolaczGwint

— ciąg dalszy

Obiekt	Właściwości		Zdarzenia	
	Nazwa	Wartość	Nazwa	Metoda
RadioGroup2	Caption	Rodzaj gwintu	OnClick	RadioGroup3Click
	Items.Strings	&zwykły &drobnozwojowy		
	ItemIndex	0		
RadioGroup3	Caption	Rodzaj śruby	OnClick	RadioGroup4Click
	Items.Strings	Łeb &sześciokątny &Cała gwintowana Łeb &walcowy		
	ItemIndex	0		
RadioGroup4	Caption	Klasa dokł.	OnClick	RadioGroup5Click
	Items.Strings	I II III		
	ItemIndex	0		
	Columns	3		
DBGrid1	Columns[0].FieldName	Nazwa	OnClick	DBGrid1CellClick
	DataSource	DataSource1		
	dgEditing	False		
	dgColumn- Resize	False		
Query1				
Query2				
DataSource1	DataSet	Query1		

Obsługę formatki programu *PolaczGwint* i dostęp do zgromadzonych danych zrealizowano w module *PGwintFrm*. Procedury do wykonywania rysunków śrub, nakrętek, gwintów i podkładek umieszczono odpowiednio w modułach: *RysŚruby*, *RysNakr*, *RysGwint* i *RysPodkl*. Przedstawione zostaną przykładowe moduły *PolaczGwint* i *RysŚruby*. Moduły dotyczące rysowania pozostałych elementów zbudowano podobnie jak moduł *RysŚruby*; dostępne są one na płycie CD załączonej do książki.

Kod modułu *PGwintFrm*:

```

unit PGwintyFrm;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ComCtrls, StdCtrls, Grids, DBGrids, Db, DBTables, ExtCtrls, Registry,
  RysSrubby, RysPodkl, RysNakr, RysGwint;
const
  NazwaKlucza='Software\PGI PK\Gwinty';
type
  TForm1 = class(TForm)
    CheckBox1: TCheckBox;
    RadioGroup10: TRadioGroup;
    Label3: TLabel;
    ComboBox2: TComboBox;
    RadioGroup11: TRadioGroup;
    RadioGroup12: TRadioGroup;
    RadioGroup13: TRadioGroup;
    Image1: TImage;
    Panel1: TPanel;
    Panel2: TPanel;
    Image2: TImage;
    Panel3: TPanel;
    Image3: TImage;
    Panel4: TPanel;
    Image4: TImage;
    procedure RadioGroup9Click(Sender: TObject);
    procedure RadioGroup5Click(Sender: TObject);
    procedure RadioGroup12Click(Sender: TObject);
    procedure DBGrid3CellClick(Column: TColumn);
    procedure RadioGroup13Click(Sender: TObject);
    procedure DBGrid2CellClick(Column: TColumn);
    procedure FormDestroy(Sender: TObject);
  published
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    TabSheet3: TTabSheet;
    TabSheet4: TTabSheet;
    DataSource1: TDataSource;
    Query1: TQuery;
    Query2: TQuery;
    DBGrid1: TDBGrid;
    Button1: TButton;
    RadioGroup2: TRadioGroup;
    RadioGroup1: TRadioGroup;
    DBGrid2: TDBGrid;
    Label1: TLabel;
    DBGrid3: TDBGrid;
    DBGrid4: TDBGrid;
    RadioGroup3: TRadioGroup;
    Label2: TLabel;
    ComboBox1: TComboBox;
    Label4: TLabel;
    ComboBox3: TComboBox;
    Label6: TLabel;
    RadioGroup9: TRadioGroup;
  end;

```

```

RadioGroup6: TRadioGroup;
RadioGroup5: TRadioGroup;
ComboBox6: TComboBox;
Label8: TLabel;
Label10: TLabel;
ComboBox8: TComboBox;
RadioGroup7: TRadioGroup;
RadioGroup8: TRadioGroup;
Label11: TLabel;
Label12: TLabel;
RadioGroup4: TRadioGroup;
procedure PageControl1Change(Sender: TObject);
procedure RadioGroup2Click(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure RadioGroup7Click(Sender: TObject);
procedure RadioGroup6Click(Sender: TObject);
procedure RadioGroup8Click(Sender: TObject);
procedure RadioGroup3Click(Sender: TObject);
procedure DBGrid1CellClick(Column: TColumn);
procedure RadioGroup4Click(Sender: TObject);
procedure Button1Click(Sender: TObject);
private
  CurrentRow:String; //zapamięt. aktualne oznaczenie gwintu
  CurrentRowPod:String; //zapamięt. aktualne oznaczenie gwintu bez skoku
  procedure UstawCurrentRow;
  procedure Rys82101; //śruby
  procedure Rys82105;
  procedure Rys82302;
  procedure Rys82144; //nakrętki
  procedure Rys82165;
  procedure RysW02013; //gwinty wewnętrzne
  procedure RysZ02013; //gwinty zewnętrzne
  procedure Rys82005; //podkładki
  procedure Rys82006;
  procedure Rys82008;
public
end;
var
  Form1:TForm1;
  NazwaTabeli,OldCurrDir:String;
  Reg:TRegistry;
implementation
{$R *.DFM}
function MyStrToFloat(Str:String):Double; //konwersja łańcuchów na liczby
//funkcja pomocnicza do konwersji liczby w nawiasach w na liczbę
//(liczba)->liczba, w razie niepowodzenia zwraca zero i komunikat
begin
  if Pos(',',Str)<>0 then
    Str:=Copy(Str,Pos(',',Str)+1,Pos(')',Str)-2);
  try
    Result:=StrToFloat(Str);
  except
    ShowMessage(Format('Konwersja łańcucha %s nie powiodła się',[Str]));
    Result:=0;
  end;
end;
procedure TForm1.UstawCurrentRow;

```

```
begin
  if (Query1.Active) and (Query1.FieldName('Nazwa').AsString<>'') then
  begin
    CurrentRow:=Query1.FieldName('Nazwa').AsString;
    if Pos('x',CurrentRow)<>0 then //zapamiętuje nazwę gwintu dla podkładek
      CurrentRowPod:=Copy(CurrentRow,1,Pos('x',CurrentRow)-1) //np. M12x1.5
    else
      CurrentRowPod:=CurrentRow; //np. M12
    if (Pos('x',CurrentRow)<>0) and (Pos('(',CurrentRow)<>0) then
      CurrentRowPod:=CurrentRowPod+')'; //dla średnic niezależnych
    end;
  end;
end;
procedure TForm1.FormCreate(Sender: TObject);
//ustawienie elementów zakładek
begin
  try //odczytanie z rejestru ostatnio wybranego widoku
    Reg:=TRegistry.Create;
    Reg.RootKey:=HKEY_CURRENT_USER;
    Reg.OpenKey(NazwaKlucza,True);
    try
      RadioGroup1.ItemIndex:=Reg.ReadInteger('Ostatni');
    except
      RadioGroup1.ItemIndex:=0; //w razie błędu odczytu
    end;
  finally
    Reg.Free;
  end;
  CurrentRow:='M10'; //program domyślnie ustawia kursor w tab. na gw. M10
  PageControl1.OnChange(Form1);
end;
procedure TForm1.PageControl1Change(Sender: TObject);
//aktualizuje dane na stronie wybranej za pomocą zakładki
begin
  Query1.Active:=False;
  Query1.SQL.Clear; //usuwanie istniejących zapytań
  Query1.SQL.Add('select * from ' + NazwaTabeli); //dodanie zapytania
  case PageControl1.ActivePage.TabIndex of
    0: RadioGroup3.OnClick(PageControl1);
    1: RadioGroup5.OnClick(PageControl1);
    2: RadioGroup6.OnClick(PageControl1);
    3: RadioGroup7.OnClick(PageControl1);
  end;
  //wyłączenie wyboru powłoki dla gwintów
  if PageControl1.ActivePage.TabIndex=2 then
    ComboBox2.Enabled:=False
  else
    ComboBox2.Enabled:=True;
  end;
end;
procedure TForm1.RadioGroup6Click(Sender: TObject);
begin
  try
    RadioGroup9.ItemIndex:=RadioGroup6.ItemIndex;
    RadioGroup2.ItemIndex:=RadioGroup6.ItemIndex;
    Query1.DisableControls; //wyłączenie wyświetlania wyników wyszukiwania
    Query1.Active:=False;
    if RadioGroup6.ItemIndex=0 then //ustalenie treści kwerendy SQL
      Query1.SQL[0]:=Format('select * from %s where %s not like %s',
        ['DB\Gwinty_02013','Nazwa','"%x%"'])
    end;
  end;
end;
```

```

else
    Query1.SQL[0]:=Format('select * from %s where %s like %s',
        ['DB\Gwinty_02013"', 'Nazwa', '"%x%"']);
    Query1.Active:=True;
    Query1.Locate('Nazwa', CurrentRow, []); //ustawienie zapamiętanej pozycji
finally
    Query1.EnableControls; //włączenie wyświetlania wyników wyszukiwania
end;
RadioGroup13.OnClick(RadioGroup6);
DBGrid3.OnCellClick(DBGrid3.Columns[0]);
end;
procedure TForm1.RadioGroup7Click(Sender: TObject);
//dynamiczne wypełnianie pozycji w komponencie RadioGroup
begin
    RadioGroup8.Items.Clear;
    if RadioGroup7.ItemIndex=0 then begin
        RadioGroup8.Items.Add('&Zgrubne');
        RadioGroup8.Items.Add('&Dokładne');
    end
    else begin
        RadioGroup8.Items.Add('&Zwykłe');
        RadioGroup8.Items.Add('&Lekkie');
    end;
    RadioGroup8.ItemIndex:=0;
    RadioGroup8.OnClick(RadioGroup7);
end;
procedure TForm1.RadioGroup8Click(Sender: TObject);
begin
    try
        UstawCurrentRow;
        Query1.DisableControls;
        Query1.Active:=False;
        //podkładki okrągłe zgrubne
        if (RadioGroup7.ItemIndex=0) and (RadioGroup8.ItemIndex=0) then begin
            RadioGroup10.Visible:=False;
            RadioGroup11.Visible:=False;
            Query1.SQL[0]:='select * from "DB\Podk10krZ_82005"';
            Image4.Picture.LoadFromFile('rys\82005.bmp');
        end;
        //okrągłe dokładne
        if (RadioGroup7.ItemIndex=0) and (RadioGroup8.ItemIndex=1) then begin
            RadioGroup10.Visible:=True;
            RadioGroup11.Visible:=True;
            if RadioGroup10.ItemIndex=0 then //normalne
                Query1.SQL[0]:='select * from "DB\Podk10krD_82006"';
            else //zmniejszone
                Query1.SQL[0]:='select * from "DB\Podk10krD_82006" where '+'
                    ' DZZ is not null';
                Image4.Picture.LoadFromFile('rys\82006.bmp');
            end;
        end;
        //sprężyste
        if RadioGroup7.ItemIndex=1 then begin
            RadioGroup10.Visible:=False;
            RadioGroup11.Visible:=False;
            if RadioGroup8.ItemIndex=0 then
                Query1.SQL[0]:='select * from "DB\Podk1SprZ_82008"';
            else
                Query1.SQL[0]:='select * from "DB\Podk1SprL_82008"';

```

```
        Image4.Picture.LoadFromFile('rys\82008.bmp');
    end;
    Query1.Active:=True;
    Query1.Locate('Nazwa',CurrentRowPod,[]);
finally
    Query1.EnableControls;
end;
end;
procedure TForm1.RadioGroup2Click(Sender: TObject);
var
    DMin, DMax:Double;
begin
    try
        //przełączenie wszystkich zakładek na odpowiedni rodzaj gwintu
        RadioGroup9.ItemIndex:=RadioGroup2.ItemIndex;
        RadioGroup6.ItemIndex:=RadioGroup2.ItemIndex;
        Query1.DisableControls;
        Query1.Active:=False;
        //zawężenie wyboru ze względu na klasę dokładności śrub
        case RadioGroup4.ItemIndex of
            0: begin
                DMin:=1.6; DMax:=160;
            end;
            1: begin
                DMin:=3; DMax:=39;
            end;
            2: begin
                DMin:=5; DMax:=52;
            end
        else begin //w razie problemu przyjmuję się III klasę
            DMin:=5; DMax:=52;
        end
    end;
    if RadioGroup2.ItemIndex=0 then
        Query1.SQL[0]:=Format('select * from %s where %s not like %s' +
            ' and D>=%g and D<=%g;',
            [NazwaTabeli,'Nazwa','"%x%"',DMin,DMax]);
    else
        Query1.SQL[0]:=Format('select * from %s where %s like %s',
            [NazwaTabeli,'Nazwa','"%x%"']);
    Query1.Active:=True;
    Query1.Locate('Nazwa',CurrentRow,[]);
finally
    Query1.EnableControls;
end;
end;
procedure TForm1.RadioGroup3Click(Sender: TObject);
begin
    //wybieranie nazwy tabeli z danymi dla wskazanego rodzaju śrub
    case RadioGroup3.ItemIndex of
        0: NazwaTabeli:='DB\Sruby_82101';
        1: NazwaTabeli:='DB\Sruby_82105';
        2: NazwaTabeli:='DB\Sruby_82302';
    end;
    //wczytywanie widoków śrub
    case RadioGroup3.ItemIndex of
        0: Image1.Picture.LoadFromFile('rys\82101.bmp');
        1: Image1.Picture.LoadFromFile('rys\82105.bmp');
        2: Image1.Picture.LoadFromFile('rys\82302.bmp');
```

```

end;
RadioGroup2.OnClick(RadioGroup3);
RadioGroup4.OnClick(RadioGroup3);
DBGrid1.OnCellClick(DBGrid1.Columns[0]);
end;
procedure TForm1.RadioGroup9Click(Sender: TObject);
begin //wybiera nazwę tabeli z danymi dla wskazanego rodzaju nakrętek
  try
    //przełączenie wszystkich zakładek na odpowiedni rodzaj gwintu
    RadioGroup2.ItemIndex:=RadioGroup9.ItemIndex;
    RadioGroup6.ItemIndex:=RadioGroup9.ItemIndex;
    Query1.DisableControls;
    Query1.Active:=False;
    //wykonanie A,B gwint zwykły
    if (RadioGroup9.ItemIndex=0) and ((RadioGroup12.ItemIndex=0)
      or (RadioGroup12.ItemIndex=1)) then begin
      Query1.SQL[0]:=Format('select * from %s where %s not like %s',
        [NazwaTabeli,'Nazwa','"%x%"']);
    end;
    //wykonanie A,B gwint drobnzwojowy
    if (RadioGroup9.ItemIndex=1) and ((RadioGroup12.ItemIndex=0)
      or (RadioGroup12.ItemIndex=1)) then begin
      Query1.SQL[0]:=Format('select * from %s where %s like %s',
        [NazwaTabeli,'Nazwa','"%x%"']);
    end;
    //wykonanie C gwint zwykły
    if (RadioGroup9.ItemIndex=0) and (RadioGroup12.ItemIndex=2) then begin
      Query1.SQL[0]:=Format('select * from %s where %s not like %s'+
        ' and %s is not null',[NazwaTabeli,'Nazwa','"%x%"','w1C']);
    end;
    //wykonanie C gwint drobnzwojowy
    if (RadioGroup9.ItemIndex=1) and (RadioGroup12.ItemIndex=2) then begin
      ShowMessage('Nakrętki zgrubne wykonywane są tylko z '+
        ' gwintem zwykłym.');
      RadioGroup9.ItemIndex:=0;
    end;
    Query1.Active:=True;
    Query1.Locate('Nazwa',CurrentRow,[]);
  finally
    Query1.EnableControls;
  end;
end;
procedure TForm1.RadioGroup4Click(Sender: TObject);
//ustalanie wytrzymałości śruby w zależności od dokładności wykonania
begin
  if RadioGroup4.ItemIndex=0 then begin
    ComboBox3.Items.Clear;
    ComboBox3.Items.CommaText:=('5.6,5.8,6.8,8.8,10.9,12.9');
  end;
  if RadioGroup4.ItemIndex=1 then begin
    ComboBox3.Items.Clear;
    ComboBox3.Items.CommaText:=('5.6,5.8,8.8,10.9,12.9');
  end;
  if RadioGroup4.ItemIndex=2 then begin
    ComboBox3.Items.Clear;
    ComboBox3.Items.CommaText:=('3.6,4.8,5.8');
    if RadioGroup2.ItemIndex=1 then begin
      ShowMessage('Śruby zgrubne wykonywane są tylko z gwintem zwykłym.');
    end;
  end;

```

```

        RadioGroup2.ItemIndex:=0;
    end
end;
if TComboBox(Sender).Name<>'ComboBox3' then
    ComboBox3.Text:=ComboBox3.Items[0];
    RadioGroup2.OnClick(RadioGroup4);
end;
procedure TForm1.DBGrid1CellClick(Column: TColumn);
//przepisywanie dopuszczalnych dla wskazanej śruby wartości długości
//z szeregu długości zapisanego w tablicy bazy danych do ComboBox
//pokazującą długość elementu
var
    i: Integer;
begin
    if (ComboBox3.Text<>'4.6') and (ComboBox3.Text<>'5.6')
        and (ComboBox3.Text<>'8.8') and (Query1.FieldByName('D').AsFloat>52)
        and (Form1.PageControl1.ActivePage.TabIndex=0) then begin
        ComboBox3.Items.Clear;
        ComboBox3.Items.CommaText:=('4.6,5.6,8.8');
        ShowMessage('Dla d>52 dopuszczalna wytrzymałość to: 4.6,5.6,8.8');
        end;
//wyszycie odpowiedniego zakresu z szeregu długości śrub
        Query2.Active:=False; Query2.SQL.Clear;
        case RadioGroup3.ItemIndex of //dla śrub z łbem walcowym szer. II
            2: Query2.SQL.Add(Format('select Dlugosc from %s'+
                ' where WartDlug<=%g and WartDlug>=%g',
                [ "DB\II_SzerDlugSrub", Query1.FieldByName('Lmax').AsFloat,
                Query1.FieldByName('Lmin').AsFloat ]));
            else //dla pozostałych śrub szereg I
                Query2.SQL.Add(Format('select Dlugosc from %s'+
                    ' where WartDlug<=%g and WartDlug>=%g',
                    [ "DB\I_SzerDlugSrub", Query1.FieldByName('Lmax').AsFloat,
                    Query1.FieldByName('Lmin').AsFloat ]));
            end;
        Query2.Active:=True; ComboBox1.Items.Clear; Query2.First;
        for i:=0 to Query2.RecordCount-1 do begin
            ComboBox1.Items.Add(Query2.FieldByName('Dlugosc').AsString);
            Query2.Next;
        end;
//wyswietlenie środkowej wartości z listy długości
        ComboBox1.Text:=ComboBox1.Items[Round(ComboBox1.Items.Count/2)];
        UstawCurrentRow;
    end;
procedure TForm1.DBGrid2CellClick(Column: TColumn);
begin
    UstawCurrentRow;
end;
procedure TForm1.RadioGroup13Click(Sender: TObject);
begin //wczytywanie widoków gwintów
    case RadioGroup13.ItemIndex of
        0: Image3.Picture.LoadFromFile('rys\W02013.bmp');
        1: Image3.Picture.LoadFromFile('rys\Z02013.bmp');
    end;
end;
procedure TForm1.DBGrid3CellClick(Column: TColumn);
var
    i: Integer;
begin

```

```

UstawCurrentRow;
Query2.Active:=False; Query2.SQL.Clear;
Query2.SQL.Add('select Dlugosc from "DB\I_SzerDlugSrub"');
Query2.Active:=True; ComboBox8.Items.Clear; Query2.First;
for i:=0 to Query2.RecordCount-1 do begin
  ComboBox8.Items.Add(Query2.FieldByName('Dlugosc').AsString);
  Query2.Next;
end;
ComboBox8.Text:=ComboBox8.Items[Round(ComboBox8.Items.Count/2)]
end;
procedure TForm1.RadioGroup5Click(Sender: TObject);
begin //wybór rodzaju nakrętki
  case RadioGroup5.ItemIndex of
    0: NazwaTabeli:="DB\Nakr_82144";
    1: NazwaTabeli:="DB\Nakr_82165";
  end;
  //wczytywanie widoków nakrętek; obie takie same
  case RadioGroup5.ItemIndex of
    0: Image2.Picture.LoadFromFile('rys\82144.bmp');
    1: Image2.Picture.LoadFromFile('rys\82144.bmp');
  end;
  RadioGroup9.OnClick(RadioGroup9);
  RadioGroup12.OnClick(RadioGroup5);
end;
procedure TForm1.RadioGroup12Click(Sender: TObject);
begin //ustalenie dopuszczalnych własności nakrętki
  if RadioGroup12.ItemIndex=2 then begin
    ComboBox6.Items.Clear;
    ComboBox6.Items.CommaText:=('4,5');
  end
  else begin
    ComboBox6.Items.Clear;
    ComboBox6.Items.CommaText:=('5,6,8,10,12');
  end;
  if TComboBox(Sender).Name<>'ComboBox6' then
    ComboBox6.Text:=ComboBox6.Items[0];
  RadioGroup9.OnClick(RadioGroup12);
end;
procedure TForm1.Button1Click(Sender: TObject);
begin //rysowanie
  case PageControl1.ActivePage.TabIndex of //wybór rodzaju elementu
    0: //śruby
      case RadioGroup3.ItemIndex of //wybór rodzaju śruby
        0:Rys82101; //z łbem sześciokątnym
        1:Rys82105; //z łbem sześciokątnym i gwintem na całej długości
        2:Rys82302; //z łbem wałcowym
      end;
    1: //nakrętki
      case RadioGroup5.ItemIndex of //wybór rodzaju nakrętki
        0:Rys82144; //nakrętka sześciokątna
        1:Rys82165; //nakrętka sześciokątna zmniejszona
      end;
    2: //gwinty
      case RadioGroup13.ItemIndex of //wybór rodzaju gwintu
        0:RysW02013; //gwinty wewnętrzne
        1:RysZ02013; //gwinty zewnętrzne
      end;
    3: //podkładki

```

```
        case RadioGroup7.ItemIndex of //wybór rodzaju podkładki
        0: //okrągła
            case RadioGroup8.ItemIndex of
            0:Rys82005; //okrągła zgrubna
            1:Rys82006; //okrągła dokładna
            end;
        1: Rys82008; //sprężysta
        end;
    end;
end;
procedure TForm1.Rys82101;
var
    r,b:Double;
begin
    if RadioGroup4.ItemIndex=2 then
        r:=Query1.FieldByName('RMin_III').AsFloat
    else
        r:=Query1.FieldByName('RMin').AsFloat;
    if MyStrToFloat(ComboBox1.Text)<=125 then
        b:=StrToFloat(Query1.FieldByName('B').AsString)
    else
        if StrToFloat(ComboBox1.Text)<=200 then
            b:=StrToFloat(Query1.FieldByName('B125').AsString)
        else
            b:=StrToFloat(Query1.FieldByName('B200').AsString);
    try //powołanie obiektu i przypisanie wartości
        with Query1 do
            Sr82101:=TSr82101.Create(FieldByName('d').AsFloat,
                FieldByName('k').AsFloat,FieldByName('s').AsFloat,
                MyStrToFloat(ComboBox1.Text),r,b,FieldByName('Nazwa').AsString,
                ComboBox3.Text,RadioGroup4.Items[RadioGroup4.ItemIndex],
                ComboBox2.Text,CheckBox1.Checked);
            case RadioGroup1.ItemIndex of //wywołanie metody rysowania
            0: Sr82101.RysWG;
            1: Sr82101.RysWB;
            end;
        finally
            Sr82101.Free; //zwolnienie obiektu
        end;
    end;
end;
procedure TForm1.Rys82105;
var
    r,a:Double;
begin
    if RadioGroup4.ItemIndex=2 then begin
        a:=Query1.FieldByName('A_III').AsFloat;
        r:=Query1.FieldByName('RMin_III').AsFloat;
    end
    else begin
        a:=Query1.FieldByName('A').AsFloat;
        r:=Query1.FieldByName('RMin').AsFloat;
    end;
    try //powołanie obiektu i przypisanie wartości
        with Query1 do
            Sr82105:=TSr82105.Create(
                FieldByName('d').AsFloat,
                FieldByName('k').AsFloat,FieldByName('s').AsFloat,
                MyStrToFloat(ComboBox1.Text),r,a,FieldByName('Nazwa').AsString,
```

```

        ComboBox3.Text, RadioGroup4.Items[RadioGroup4.ItemIndex],
        ComboBox2.Text, CheckBox1.Checked);
    case RadioGroup1.ItemIndex of //wywołanie metody rysowania
        0: Sr82105.RysWG;
        1: Sr82105.RysWB;
    end;
finally
    Sr82105.Free; //zwolnienie obiektu
end;
end;
procedure TForm1.Rys82302;
var
    b:Double;
begin
    b:=StrToFloat(Query1.FieldByName('B').AsString);
    if MyStrToFloat(ComboBox1.Text)<Query1.FieldByName('LGC').AsFloat then
        b:=MyStrToFloat(ComboBox1.Text)-0.15*Query1.FieldByName('d').
            AsFloat-Query1.FieldByName('RMin').AsFloat; //gwint na całej długości
    try //powołanie obiektu i przypisanie wartości
        with Query1 do
            Sr82302:=TSr82302.Create(FieldByName('d').AsFloat,
                FieldByName('k').AsFloat,FieldByName('s').AsFloat,
                MyStrToFloat(ComboBox1.Text),FieldByName('RMin').AsFloat,
                b,FieldByName('DLba').AsFloat,FieldByName('Nazwa').AsString,
                ComboBox3.Text, RadioGroup4.Items[RadioGroup4.ItemIndex],
                ComboBox2.Text, CheckBox1.Checked);
            case RadioGroup1.ItemIndex of //wywołanie metody rysowania
                0: Sr82302.RysWG;
                1: Sr82302.RysWB;
            end;
        finally
            Sr82302.Free; //zwolnienie obiektu
        end;
    end;
end;
procedure TForm1.Rys82005;
begin
    try //powołanie obiektu i przypisanie wartości
        with Query1 do
            Pod82005:=TPod82005.Create(FieldByName('dz').AsFloat,
                FieldByName('dw').AsFloat,FieldByName('g').AsFloat,
                ComboBox2.Text, CheckBox1.Checked);
            case RadioGroup1.ItemIndex of //wywołanie metody rysowania
                0: Pod82005.RysWG;
                1: Pod82005.RysWB;
            end;
        finally
            Pod82005.Free; //zwolnienie obiektu
        end;
    end;
end;
procedure TForm1.Rys82006;
var
    x,f:Double;
begin
    if RadioGroup11.ItemIndex=0 then begin
        x:=0; f:=0;
    end
end

```

```

    else begin
        x:=Query1.FieldByName('x').AsFloat;
        f:=Query1.FieldByName('Fmin').AsFloat;
    end;
try //powołanie obiektu i przypisanie wartości
    with Query1 do
        if RadioGroup10.ItemIndex=0 then
            Pod82006:=TPod82006.Create(FieldByName('dz').AsFloat,
                FieldByName('dw').AsFloat,FieldByName('g').AsFloat,
                f,x,'',ComboBox2.Text,CheckBox1.Checked)
        else
            Pod82006:=TPod82006.Create(FieldByName('dzz').AsFloat,
                FieldByName('dw').AsFloat,FieldByName('gz').AsFloat,
                f,x,'Zm',ComboBox2.Text,CheckBox1.Checked);
//wywołanie metody rysowania
        case RadioGroup1.ItemIndex of
            0: Pod82006.RysWG;
            1: Pod82006.RysWB;
        end;
    finally //zwolnienie obiektu
        Pod82006.Free;
    end;
end;
procedure TForm1.Rys82008;
var
    OzTypu:String;
begin
    if RadioGroup8.ItemIndex=0 then
        OzTypu:='Z' //zwykłe
    else
        OzTypu:=''; //lekkie
try //powołanie obiektu i przypisanie wartości
    with Query1 do
        Pod82008:=TPod82008.Create(FieldByName('dz').AsFloat,
            FieldByName('dw').AsFloat,FieldByName('g').AsFloat,OzTypu,
            ComboBox2.Text,CheckBox1.Checked);
        case RadioGroup1.ItemIndex of //wywołanie metody rysowania
            0: Pod82008.RysWG;
            1: Pod82008.RysWB;
        end;
    finally
        Pod82008.Free; //zwolnienie obiektu
    end;
end;
procedure TForm1.Rys82144;
begin
    try //powołanie obiektu i przypisanie wartości
        with Query1 do
            case RadioGroup12.ItemIndex of
//wykonanie A
                0: Nak82144:=TNak82144.Create(FieldByName('d').AsFloat,
                    FieldByName('s').AsFloat,FieldByName('w').AsFloat,
                    FieldByName('w1AB').AsFloat,FieldByName('Nazwa').AsString,
                    ComboBox2.Text,'A',ComboBox6.Text,CheckBox1.Checked);
//wykonanie B
                1: Nak82144:=TNak82144.Create(FieldByName('d').AsFloat,
                    FieldByName('s').AsFloat,FieldByName('w').AsFloat,

```

```

        FieldByName('w1AB').AsFloat,FieldByName('Nazwa').AsString,
        ComboBox2.Text,'B',ComboBox6.Text,CheckBox1.Checked);
//wykonanie C
    2: Nak82144:=TNak82144.Create(FieldByName('d').AsFloat,
        FieldByName('s').AsFloat,FieldByName('w').AsFloat,
        FieldByName('w1C').AsFloat,FieldByName('Nazwa').AsString,
        ComboBox2.Text,'C',ComboBox6.Text,CheckBox1.Checked);
    end;
    case RadioGroup1.ItemIndex of //wywołanie metody rysowania
        0: Nak82144.RysWG;
        1: Nak82144.RysWB;
    end;
finally
    Nak82144.Free; //zwolnienie obiektu
end;
end;
procedure TForm1.Rys82165;
begin
    Rys82144;
end;
procedure TForm1.RysW02013;
begin
    try //powołanie obiektu i przypisanie wartości
        with Query1 do
            GwW02013:=TGwintWewn.Create(FieldByName('d').AsFloat,MyStrToFloat(
                ComboBox8.Text),FieldByName('Skok').AsFloat,FieldByName('Nazwa').
                AsString,CheckBox1.Checked);
            case RadioGroup1.ItemIndex of //wywołanie metody rysowania
                0: GwW02013.RysWG;
                1: GwW02013.RysWB;
            end;
        finally
            GwW02013.Free; //zwolnienie obiektu
        end;
    end;
end;
procedure TForm1.RysZ02013;
begin //powołanie obiektu i przypisanie wartości
    try
        with Query1 do
            GwZ02013:=TGwintZewn.Create(FieldByName('d').AsFloat,MyStrToFloat(
                ComboBox8.Text),FieldByName('Skok').AsFloat,FieldByName('Nazwa').
                AsString,CheckBox1.Checked);
            case RadioGroup1.ItemIndex of //wywołanie metody rysowania
                0: GwZ02013.RysWG;
                1: GwZ02013.RysWB;
            end;
        finally
            GwZ02013.Free; //zwolnienie obiektu
        end;
    end;
end;
procedure TForm1.FormDestroy(Sender: TObject);
begin
    try //zapisanie zmiany do rejestru
        Reg:=TRegistry.Create;
        Reg.RootKey:=HKEY_CURRENT_USER;
        Reg.OpenKey(NazwaKlucza,True);
        Reg.WriteInteger('Ostatni',RadioGroup1.ItemIndex);
    end;
end;

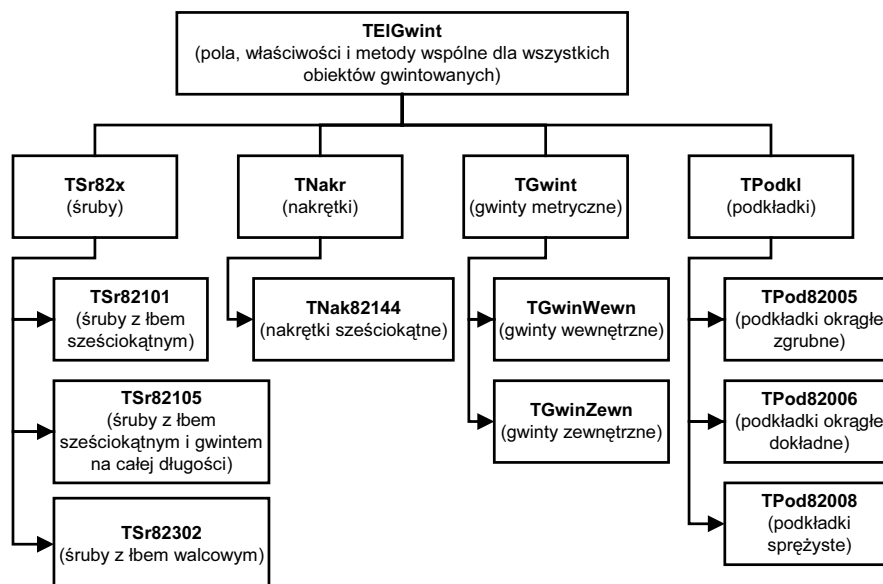
```

```

finally
  Reg.Free;
end;
end;
initialization //ustawienie katalogu roboczego
  SetCurrentDir(ExtractFilePath(Application.ExeName));
end.

```

W module *RysSruby* umieszczono definicję klasy *TElGwint* zawierającą pola i metody wspólne dla wszystkich obiektów opisujących elementy połączeń śrubowych użytych w programie *PolaczGwint*, dziedziczącą po nim klasę *TSr82x*, opisującą cechy wspólne dla śrub i wyprowadzone z niej klasy *TSr82101*, *TSr82105* i *TSr82302* określające odpowiednio śruby z łbem sześciokątnym, śruby z łbem sześciokątnym z gwintem na całej długości i śruby z łbem walcowym. Strukturę dziedziczenia dla obiektów elementów połączeń gwintowych przedstawiono na rysunku 5.4.



Rysunek 5.4. Hierarchia obiektów opisujących elementy połączeń gwintowych

Kod modułu *RysSruby* przedstawia się następująco:

```

unit RysSruby;
interface
uses
  PolaczAutoCAD, AcadConst, WarstwyLinie;
const
  Sk0si=0.2; // współczynnik długości osi symetrii
type
  TElGwint=class
    //klasa bazowa, tylko do dziedziczenia
    x0,y0,d,Sciecie,e,h1,r12,l,Skok:Double;
    NazwaBloku,NrNormy,OznGwintu,Powloka:String;
    RysujOsie:Boolean;
    BlokRef,ObjRef,BlockDwg:OLEVariant;
    ObjNr,Pisak,Warstwa:Integer;

```

```

public
    constructor Create(dp,sp:Double;OznGwP,PowlP:string;RysOsieP:Boolean);
    destructor Destroy;override;
    procedure RysWG;
    procedure RysWB;
protected
    procedure RysOsAcadWB(DlOsi:Double);
    procedure RysOsAcadWG(DlOsi:Double);
    procedure RysGwintWewAcadWG;
    procedure RysGwintZewAcadWG;
    procedure RysSzescAcadWG;
    procedure DodajOznac;
    procedure UmiescObiekt;
    function UsunNawiasy(StrZNaw:String):String;
    function BlokIstnieje(Nazwa:String):Boolean;
    procedure RysAcadWG;virtual;abstract;
    procedure RysAcadWB;virtual;abstract;
private
    FOznaczenie: String;
    FS:Double;
    procedure SetOznaczenie(const Value: String);
    procedure SetS(const Value: Double);
    procedure UmiescObiektAcad;
    procedure DodajOznacAcad;
published
    property Oznaczenie:String read FOznaczenie write SetOznaczenie;
    property S:Double read FS write SetS;
end;
// definicje klas opisujących śruby jest to klasa pochodna od TELGwint
// bazowa dla trzech klas opisujących śruby: TSr82101, TSr82105, TSr82302
TSr82x=class(TELGwint)
    k,l,r:Double;
    Wytrz, KlasDokl:String;
private //wspólne dla śrub
    procedure RysLebSzescAcadWB;
    procedure RysTrzypienAcadWB;
    procedure RysOsAcadWB(DlOsi:Double);
public
    constructor Create(dp,kp,sp,lp,rp:Double;OznGwP,Wytrz,KlasDokl,
        PowlP:String;OsieP:Boolean);
    procedure RysAcadWG;override; //procedury rysujące widok śrub z góry
end;
TSr82101=class(TSr82x) //śruba z łbem sześciokątnym
private
    b:Double;
    procedure RysGwintAcadWB;
public
    constructor Create(dp,kp,sp,lp,rp,bp:Double;OznGwP,Wytrz,
        KlasDokl:String;PowlP:String;OsieP:Boolean);
    procedure RysAcadWB;override; //nadpisanie procedury rysujące widok śrub
end;
TSr82105=class(TSr82x) //śruba z łbem sześciokątnym i gwintem na
    a:Double; //całej długości
private
    procedure RysGwintAcadWB;
public
    constructor Create(dp,kp,sp,lp,rp,ap:Double;OznGwP,Wytrz,KlasDokl,
        PowlP:String;OsieP:Boolean);
    procedure RysAcadWB;override;

```

```
end;
TSr82302=class(TSr82x) //śruba z łbem walcowym
private
  DLba, b:Double;
  procedure RysGwintAcadWB;
public
  constructor Create(dp,kp,sp,lp,rp,bp,DLbaP:Double;OznGwP,Wytrz,
    KlasDokl,PowlP:String;OsieP:Boolean);
  procedure RysAcadWG:override; //Nadpisanie proc. rysujące widok z góry
  procedure RysAcadWB:override;
end;
var
  Sr82101:TSr82101;
  Sr82105:TSr82105;
  Sr82302:TSr82302;
implementation
uses
  SysUtils, Forms, Dialogs, Math;
constructor TElGwint.Create(dp,sp:Double;OznGwP,PowlP:string;
  RysOsieP:Boolean); // Konstruktor klasy bazowej
begin
  inherited Create;
  x0:=-100; y0:=-100; Pisak:=1; Warstwa:=-1;
  Powloka:=PowlP; OznGwintu:=UsunNawiasy(OznGwP); RysujOsie:=RysOsieP;
  d:=dp; Sciecie:=0.2*d/2*sin(Pi/4); s:=sp;
  if PolaczZAcad then begin //jeśli nawiązano połączenie
    ZapamietajUstawienia; DefiniujWarstwe; WczytajTypyLinii;
  end;
end;
constructor TSr82x.Create(dp,kp,sp,lp,rp:Double;OznGwP,Wytrz,KlasDokl,
  PowlP:String;OsieP:Boolean); // Konstruktor klasy bazowej śrub
begin
  inherited Create(dp,sp,OznGwP,PowlP,OsieP);
  k:=kp; l:=lp; r:=rp;
  Oznaczenie:=Format('Śruba %sx%-s-%s %s %s',[OznGwP,l,Wytrz,KlasDokl,
    Powloka,NrNormy]);
end;
constructor TSr82101.Create(dp,kp,sp,lp,rp,bp:Double;OznGwP,Wytrz,
  KlasDokl,PowlP:String;OsieP:Boolean); //Konstruktor śruby z łbem sześć.
begin
  NrNormy:='PN-M82101';
  inherited Create(dp,kp,sp,lp,rp,OznGwP,Wytrz,KlasDokl,PowlP,OsieP);
  b:=bp;
end;

{usunięte metody obiektów TSr82105 i TSr82302}

destructor TElGwint.Destroy; //usuwanie obiektu i przywracanie ustawień
begin
  PrzywrocUstawienia;
  inherited Destroy;
end;
procedure TElGwint.UmiescObiektAcad; //umieszczanie bloku w programie
begin
  Application.Minimize; //minimalizacja okna programu
  try
    try
      BlokRef:=AcadDwg.InsertBlock(AcadDoc.Utility.GetPoint( ,
```

```

        'Wskaż punkt wstawienia'),NazwaBlok, 1, 1, 1, 0);
        BlokRef.Update;           //odświeżenie bloku
    finally
        Application.Restore;      //przywrócenie okna programu
    end;
except
    MessageDlg('Przed wskazaniem punktu wywołano polecenie,'+
        #13+'które spowodowało utratę połączenia z serwerem OLE'+#13+
        #13+'Wstawianie elementu należy powtórzyć.',mtError,[mbOK],0);
end;
end;
procedure TElGwint.DodajOznacAcad; //dodawanie oznacz. do rys. elementu;
var
    Mode:LongInt;
begin
    Mode:=acAttributeModeInvisible; //atrybut niewidoczny po wstawieniu
    BlockDwg.AddAttribute(3.5,Mode,'',p2D(x0,y0),'Nazwa',Oznaczenie);
    BlockDwg.AddAttribute(3.5,Mode,'',p2D(x0,y0),'NrRys','NrNormy');
end;
procedure TElGwint.RysWB; //metoda rysowania widoku z boku
begin
    if Not VarIsEmpty(AcadDwg) then RysAcadWB;
end;
procedure TElGwint.RysWG; //metoda rysowania widoku z góry
begin
    if Not VarIsEmpty(AcadDwg) then RysAcadWG;
end;
procedure TElGwint.SetOznaczenie(const Value: String);
begin
    FOznaczenie:=Value;
end;
procedure TElGwint.UmiescObiekt;
begin
    UmiescObiektAcad;
end;
procedure TSr82x.RysOsAcadWB(D10si: Double); //rys. oś dla widoku bocznego
begin
    if RysujOsie then begin
        ObjRef:=BlockDwg.AddLine(p2D(x0-1.5*k,y0),p2D(x0+D10si,y0));
        ObjRef.Color:=AcRed; ObjRef.Linetype:='Center';
    end;
end;
procedure TSr82101.RysAcadWB; //śruba z łbem sześciokątnym; widok z boku;
begin
    NazwaBlok:='Oznaczenie+'_WB';
    if not BlokIstnieje(NazwaBlok) then begin
        BlockDwg:=AcadDoc.Blocks.Add(p2D(x0,y0),NazwaBlok);
        RysLebSzescAcadWB;
        RysTrzpieńAcadWB;
        RysGwintAcadWB;
        DodajOznacAcad;
    end;
    UmiescObiektAcad;
end;

{usunięte metody obiektów TSr82105 i TSr82302}

procedure TSr82101.RysGwintAcadWB; //gwint na trzpieniu w widoku bocznym;

```

```

begin
  BlockDwg.AddLine(p2D(x0+1-b,y0-d/2),p2D(x0+1-b,y0+d/2));
  ObjRef:=BlockDwg.AddLine(p2D(x0+1-b,y0-d/2+Sciecie),
    p2D(x0+1,y0-d/2+Sciecie)); //linia gwintu
  ObjRef.Color:=acRed; //ustawienie koloru i odbicie lustrzane
  ObjRef.Mirror(p2D(x0,y0),p2D(x0+1,y0));
  BlockDwg.AddLine(p2D(x0+1-b,y0-d/2+Sciecie),p2D(
    x0+1-b-(Sciecie/tan(Pi/6)),y0-d/2)); //wybiegi gwintu
  ObjRef.Color:=acRed;
  ObjRef.Mirror(p2D(x0,y0),p2D(x0+1,y0));
end;
procedure TSr82105.RysGwintAcadWB; //gwint widok z boku
begin
  BlockDwg.AddLine(p2D(x0+a,y0-d/2),p2D(x0+a,y0+d/2));
  ObjRef:=BlockDwg.AddLine(p2D(x0+a,y0-d/2+Sciecie),
    p2D(x0+1,y0-d/2+Sciecie));
  ObjRef.Color:=acRed; ObjRef.Mirror(p2D(x0,y0),p2D(x0+1,y0));
  ObjRef:=BlockDwg.AddLine(p2D(x0+a,y0-d/2+Sciecie),p2D(
    x0+a-(Sciecie/tan(Pi/6)),y0-d/2));
  ObjRef.Color:=acRed; ObjRef.Mirror(p2D(x0,y0),p2D(x0+1,y0));
end;
procedure TSr82302.RysGwintAcadWB;
begin
  BlockDwg.AddLine(p2D(x0+1-b,y0-d/2),p2D(x0+1-b,y0+d/2));
  ObjRef:=BlockDwg.AddLine(p2D(x0+1-b,y0-d/2+Sciecie),
    p2D(x0+1,y0-d/2+Sciecie));
  ObjRef.Color:=acRed; ObjRef.Mirror(p2D(x0,y0),p2D(x0+1,y0));
  ObjRef:=BlockDwg.AddLine(p2D(x0+1-b,y0-d/2+Sciecie),
    p2D(x0+1-b-(Sciecie/tan(Pi/6)),y0-d/2));
  ObjRef.Color:=acRed;
  ObjRef.Mirror(p2D(x0,y0),p2D(x0+1,y0));
end;
function TEIGwint.UsunNawiasy(StrZNaw: String):String;
begin //usuwa nawiasy z podanego tekstu (text)->text
  StrZNaw:=Trim(StrZNaw);
  if (Pos('(',StrZNaw)<>0) and (Pos(')',StrZNaw)<>0) then
    Result:=Copy(StrZNaw,Pos('(',StrZNaw)+1,Pos(')',StrZNaw)-2)
  else
    Result:=StrZNaw;
end;
function TEIGwint.BlokIstnieje(Nazwa:String):Boolean;
begin
  Result:=False;
  BlokRef:=Unassigned;
  try
    BlokRef:=AcadDoc.Blocks.Item(Nazwa); //próba przypisania bloku
    Result:=True;
    Application.Restore; //aby komunikat był widoczny na pierwszym planie
    MessageDlg(Format('Blok %s istnieje.'+#13+'Wstawiony zostanie '+'
      'istniejący blok', [Nazwa]),MtInformation,[MBOK],0);
    Application.Minimize; //ponowne ukrycie na czas wstawiania bloku
  except
    end;
end;
procedure TEIGwint.RysGwintWewAcadWG; //gwint wewnętrzny w widoku z góry;
begin
  ObjRef:=BlockDwg.AddArc(p2D(x0,y0),d/2,DegToRad(35), DegToRad(305));
  ObjRef.Color:=acRed;
  BlockDwg.AddCircle(p2D(x0,y0).(d/2)-Sciecie);

```

```

end;
procedure TEIGwint.RysGwintZewAcadWG; //gwint zewnętrzny w widoku z góry;
begin
  ObjRef:=BlockDwg.AddArc(p2D(x0,y0),d/2-Sciecie,DegToRad(35),
    DegToRad(305));
  ObjRef.Color:=acRed;
  BlockDwg.AddCircle(p2D(x0,y0),d/2);
end;
procedure TEIGwint.RysSzescAcadWG; //łeb sześciokątny, widok z góry;
begin
  BlockDwg.AddCircle(p2D(x0,y0),s/2); //okrąg
  ObjRef:=BlockDwg.AddLine(p2D(x0+e/2,y0),p2D(
    x0+e/2*cos(Pi/3),y0+s/2)).ArrayPolar(6,2*Pi,p2D(x0,y0)); //sześciokąt
end;
procedure TSr82x.RysLebSzescAcadWB; //łeb sześć. dla śruby; wid. bocz.;
begin
  BlockDwg.AddLine(p2D(x0,y0-e/2),p2D(x0,y0+e/2)); //łeb śruby
  BlockDwg.AddLine(p2D(x0-k,y0-s/2),p2D(x0-k,y0+s/2));
  BlockDwg.AddLine(p2D(x0-k,y0-s/2),p2D(x0-k+h1,y0-e/2)); //krawędź dolna
  BlockDwg.AddLine(p2D(x0,y0-e/2),p2D(x0-k+h1,y0-e/2));
  BlockDwg.AddLine(p2D(x0-k,y0+s/2),p2D(x0-k+h1,y0+e/2)); //krawędź górna
  BlockDwg.AddLine(p2D(x0,y0+e/2),p2D(x0-k+h1,y0+e/2));
//krawędzie środkowe
  BlockDwg.AddLine(p2D(x0,y0-e/2*cos(Pi/3)),p2D(x0-k+h1,y0-e/4));
  BlockDwg.AddLine(p2D(x0,y0+e/2*cos(Pi/3)),p2D(x0-k+h1,y0+e/4));
  BlockDwg.AddArc(p2D(x0-k+0.75*e,y0),0.75*e,Pi-arcsin(1/3),
    Pi+arcsin(1/3)); //łuki
  BlockDwg.AddArc(p2D(x0-k+r12,y0-3/8*e),r12,Pi-arcsin(0.125*e/
    r12),Pi+arcsin(0.125*e/r12));
  BlockDwg.AddArc(p2D(x0-k+r12,y0+3/8*e),r12,Pi-arcsin(0.125*e/
    r12),Pi+arcsin(0.125*e/r12));
end;
procedure TSr82x.RysTrzpieńAcadWB; //trzpień śruby widok boczny
begin
  BlockDwg.AddArc(p2D(x0+r,y0-d/2-r),r,Pi/2,Pi); //łuki pod łbem
  BlockDwg.AddArc(p2D(x0+r,y0+d/2+r),r,Pi,3/2*Pi);
//zarys trzpienia
  BlockDwg.AddLine(p2D(x0+r,y0-d/2),p2D(x0+l-Sciecie,y0-d/2));
  BlockDwg.AddLine(p2D(x0+l-Sciecie,y0-d/2), p2D(x0+l,y0-d/2+Sciecie));
  BlockDwg.AddLine(p2D(x0+l,y0-d/2+Sciecie), p2D(x0+l,y0+d/2-Sciecie));
  BlockDwg.AddLine(p2D(x0+l,y0+d/2-Sciecie), p2D(x0+l-Sciecie,y0+d/2));
  BlockDwg.AddLine(p2D(x0+r,y0+d/2),p2D(x0+l-Sciecie,y0+d/2));
  BlockDwg.AddLine(p2D(x0+l-Sciecie,y0-d/2), p2D(x0+l-Sciecie,y0+d/2));
  RysOsAcadWB(1.2*l); //oś symetrii
end;
procedure TSr82x.RysAcadWG;
//śruba z łbem sześciokątnym; widok z góry; AutoCAD
begin
  NazwaBlok:=Oznaczenie+'_WG';
  if not BlokIstnieje(NazwaBlok) then begin
    BlockDwg:=AcadDoc.Blocks.Add(p2D(x0,y0),NazwaBlok);
    RysSzescAcadWG;
    RysOsAcadWG(e);
    DodajOznaczAcad;
  end;
  UmiescObiektAcad;
end;
procedure TEIGwint.SetS(const Value: Double);

```

```

begin
  FS := Value;
  e:=s/cos(Pi/6);
  h1:=0.75*e-sqrt(sqr(0.75*e)-sqr(0.25*e));
  r12:=0.75*e*0.2714461;
end;
procedure TE1Gwint.RysOsAcadWB(D10si: Double);
begin //oś gwintu w widoku z boku;
  if RysujOsie then begin
    ObjRef:=BlockDwg.AddLine(p2D(x0-Sk0si*D10si,y0),
      p2D(x0+D10si+Sk0si*D10si,y0));
    ObjRef.Color:=AcRed; ObjRef.Linetype:='Center';
  end;
end;
procedure TE1Gwint.RysOsAcadWG(D10si: Double);
begin //oś gwintu w widoku z góry;
  if RysujOsie then begin
    ObjRef:=BlockDwg.AddLine(p2D(x0-D10si/2-Sk0si*D10si,y0),
      p2D(x0+D10si/2+Sk0si*D10si,y0));
    ObjRef.Color:=AcRed; ObjRef.Linetype:='CENTER';
    ObjRef.ArrayPolar(2,Pi/2,p2D(x0,y0));
  end;
end;
procedure TE1Gwint.DodajOznac;
begin //oznaczenie elementu
  DodajOznacAcad;
end;
end.

```

Okno programu *PolaczGwint* pokazano na rysunku 5.5. W oknie widać wybór śruby M16 wraz z jej właściwościami (wykonanie, wytrzymałość, powłoka itp.). Natomiast na rysunku 5.6 pokazano okno tego programu otrzymane po wstawieniu uprzednio wybranej śruby i przełączeniu na zakładkę o nazwie *Nakrętka*. Program automatycznie zapamiętał ostatni wybór śruby i wyselekcjonował wstępnie do tej śruby odpowiednią nakrętkę.

Rysunek 5.5.
Okno programu
PolaczGwint podczas
wyboru śruby



Rysunek 5.6.
Okno programu
PolaczGwint podczas
wyboru nakrętki



Rysunek 5.7.

Przykłady elementów
wygenerowanych
za pomocą programu
PolaczGwint w oknie
programu AutoCAD

