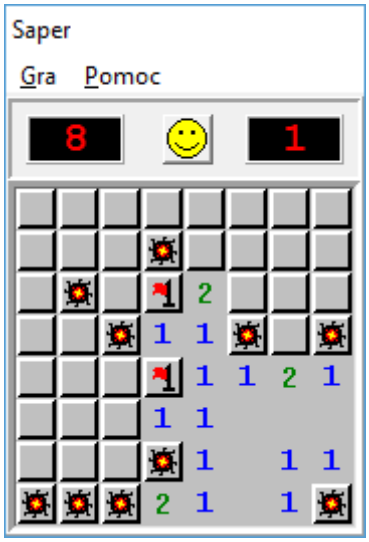
Cyfrowe projekty – SAPER - czyli, jak sprawdzić sprawność umysłu?

SAPER - komputerowa gra, napisana w 1981 roku, polegająca na odkrywaniu pól na planszy w taki sposób, aby nie trafić na minę. Na każdym odkrytym polu napisana jest liczba min, które stykają się z odkrytym polem. I te proste zasady sprawiają, że gra stała się moim ulubionym sprawdzianem ze sprawnego i logicznego myślenia. Jest tylko maleńki problem - oryginalną grę dostępną w systemach Windows da się oszukać! Aby zrobić z nij sprawdzian, musiałem napisać Sapera od nowa. W wersji dla Windows to 1500 wierszy kodu, 30000 znaków.



Mój Saper „robi" dokładnie to samo, co oryginalny Saper, ale pozbawiony jest różnego rodzaju "ułatwień". Najważniejszy jest moduł, za pomocą którego uczeń pisze sprawdzian i wystawiona zostaje ocena.

Pierwsza wersja powstała w 1995 roku i została napisana w Turbo Pascalu na systemy MSDOS, a kolejne powstawały już w Borland Delphi na systemy MS Windows.

```
{SAPER WL 1995 - MS DOS}
program saper;
uses crt;

const
  max_X=100;
  max_Y=100;
var
  pole:array[1..max_X,1..max_Y] of byte;      {tablica z minami}
  ile_X,ile_Y,ile_min:byte;                   {obrane wsp˘rz©dne pola}

{9    - mina}
{1..8 - pole puste a wko o tyle min}
{0    - pole bez miny, sytuacja przed losowaniem min}

procedure wyczysc(ile_x,ile_y:byte);
var
  x,y:byte;
begin
  for x:=1 to ile_x do
    for y:=1 to ile_y do
      pole[x,y]:=0
end;

procedure minuj(ile_x,ile_y,ile_min:byte);
var
  x,y,i:byte;
begin
  randomize;
  for i:=1 to ile_min do
    begin
      repeat
        x:=random(ile_x)+1;
        y:=random(ile_y)+1;
      until pole[x,y]<>9;      {losuj nowe wsp˘rz©dne aľ znajdziesz puste}
      pole[x,y]:=9;
    end;
  {teraz by moľna od razu obliczaţ ile min wko o}
  {do pol wok˘ dodajemy jedynk© je li nie zaj©te przez min©}
  {albo moľna to zrobiţ w osobnej procedurze po rotlosowaniu min}
end;

{1 - moľna szukaţ min i objechaţ wok˘ miny dodajĄc jedynk© lub}
{2 - szukaţ wolnych p˘l i policzyţ wok˘ miny}
{moľna by opisaţ sytuacj© w polu dopiero po wybraniu pola w kolejnym ruchu}
procedure opisz_miny_1(ile_x,ile_y:byte);
var
  x,y:byte;

  {sprawdzenie czy nowe wsp˘rz©dne oznaczajĄ pole czy poza tablicĄ}
  function czy_pole(x,y:byte):boolean;
  begin
    if (x<1) or (x>ile_x) or
       (y<1) or (y>ile_y)
      then czy_pole:=false
      else czy_pole:=true;
  end;

begin
  for x:=1 to ile_x do
    for y:=1 to ile_y do
```

```
      if pole[x,y]=9 then
        begin
          if czy_pole(x+1,y+1) then
            if pole[x+1,y+1]<>9 then inc(pole[x+1,y+1]);
          if czy_pole(x+1,y  ) then
            if pole[x+1,y  ]<>9 then inc(pole[x+1,y  ]);
          if czy_pole(x+1,y-1) then
            if pole[x+1,y-1]<>9 then inc(pole[x+1,y-1]);
          if czy_pole(x  ,y-1) then
            if pole[x  ,y-1]<>9 then inc(pole[x  ,y-1]);
          if czy_pole(x-1,y-1) then
            if pole[x-1,y-1]<>9 then inc(pole[x-1,y-1]);
          if czy_pole(x-1,y  ) then
            if pole[x-1,y  ]<>9 then inc(pole[x-1,y  ]);
          if czy_pole(x-1,y+1) then
            if pole[x-1,y+1]<>9 then inc(pole[x-1,y+1]);
          if czy_pole(x  ,y+1) then
            if pole[x  ,y+1]<>9 then inc(pole[x  ,y+1]);
        end;
end;

{szukamy wolnych p˘l}
procedure opisz_miny_2(ile_x,ile_y:byte);
var
  x,y:byte;

  {sprawdzenie czy nowe wsp˘rz©dne oznaczajĄ pole czy poza tablicĄ}
  function czy_pole(x,y:byte):boolean;
  begin
    if (x<1) or (x>ile_x) or
       (y<1) or (y>ile_y)
      then czy_pole:=false
      else czy_pole:=true;
  end;

begin
  for x:=1 to ile_x do
    for y:=1 to ile_y do
      if pole[x,y]<>9 then
        begin
          if czy_pole(x+1,y+1) then
            if pole[x+1,y+1]=9 then inc(pole[x,y]);
          if czy_pole(x+1,y  ) then
            if pole[x+1,y  ]=9 then inc(pole[x,y]);
          if czy_pole(x+1,y-1) then
            if pole[x+1,y-1]=9 then inc(pole[x,y]);
          if czy_pole(x  ,y-1) then
            if pole[x  ,y-1]=9 then inc(pole[x,y]);
          if czy_pole(x-1,y-1) then
            if pole[x-1,y-1]=9 then inc(pole[x,y]);
          if czy_pole(x-1,y  ) then
            if pole[x-1,y  ]=9 then inc(pole[x,y]);
          if czy_pole(x-1,y+1) then
            if pole[x-1,y+1]=9 then inc(pole[x,y]);
          if czy_pole(x  ,y+1) then
            if pole[x  ,y+1]=9 then inc(pole[x,y]);
        end;
end;

{skr˘cimy zapis czy_pole_2 przez zapisanie skok˘w w tablicy}
procedure opisz_miny_3(ile_x,ile_y:byte);
```

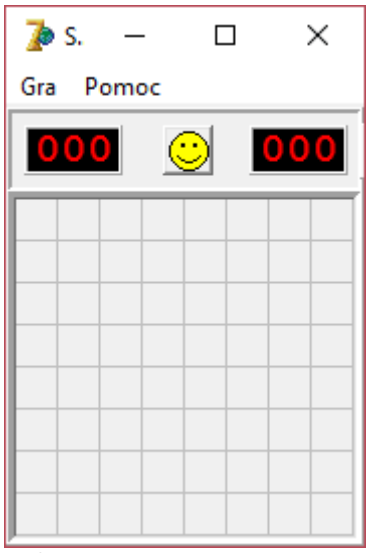
```
const
  sx:array[1..8] of shortint =(+1,+1,+1, 0,-1,-1,-1, 0);
  sy:array[1..8] of shortint =(+1, 0,-1,-1,-1, 0,+1,+1);
var
  x,y,i:byte;

  {sprawdzenie czy nowe wsp˘rz©dne oznaczajĄ pole czy poza tablicĄ}
  function czy_pole(x,y:byte):boolean;
  begin
    if (x<1) or (x>ile_x) or
       (y<1) or (y>ile_y)
      then czy_pole:=false
      else czy_pole:=true;
  end;

begin
  for x:=1 to ile_x do
    for y:=1 to ile_y do
      if pole[x,y]<>9 then
        for i:=1 to 8 do
          if czy_pole(x+sx[i],y+sy[i]) then
            if pole[x+sx[i],y+sy[i]]=9 then inc(pole[x,y]);
end;

procedure wypisz_pole(ile_x,ile_y:byte);
var
  x,y:byte;
begin
  for y:=1 to ile_y do
    begin
      for x:=1 to ile_x do
        write(pole[x,y]);
      writeln;
    end;
end;

procedure czysc_bufor;
var znak:char;
begin
  while keypressed do znak:= readkey;
end;

procedure wybieraj(var ile_x,ile_y,ile_min:byte);
var
  znak:char;
begin
  writeln('poczĄtkujĄcy - p');
  writeln('zaawansowany - z');
  writeln('ekspert      - e');
  writeln;
  writeln('wybierz poziom gry');
  czysc_bufor;
  znak:=readkey;
  case upcase(znak) of
    'P':begin ile_x:= 8; ile_y:= 8; ile_min:=10 end;
    'Z':begin ile_x:=16; ile_y:=16; ile_min:=40 end;
    'E':begin ile_x:=30; ile_y:=16; ile_min:=99 end
  end;
end;

procedure rysuj_pole(ile_x,ile_y:byte);
```

```
var
  x,y:byte;
begin
  for y:=1 to ile_y do
    begin
      for x:=1 to ile_x do
        write({'ţ'}'ţ');
      writeln;
    end;
end;

procedure zapal_puste(x,y:byte);
const
  sx:array[1..8] of shortint =(+1,+1,+1, 0,-1,-1,-1, 0);
  sy:array[1..8] of shortint =(+1, 0,-1,-1,-1, 0,+1,+1);
var
  i,x1,y1:Byte;

{sprawdzenie czy nowe wsp˘rz©dne oznaczajĄ pole czy poza tablicĄ}
function czy_pole(x,y:byte):boolean;
begin
  if (x<1) or (x>ile_x) or
     (y<1) or (y>ile_y)
    then czy_pole:=false
    else czy_pole:=true;
end;

begin
  gotoxy(x,y);
  write(' ');
  pole[x,y]:=10; {bo by si© zap©tla o bez koäca}
  for i:=1 to 8 do
  begin
    x1:=x+sx[i];
    y1:=y+sy[i];
    if czy_pole(x1,y1) then
      if pole[x1,y1]=0
        then zapal_puste(x1,y1)
        else if pole[x1,y1] in [1..8] then
          begin
            gotoxy(x1,y1);
            write(pole[x1,y1])
          end;
  end;
end;

procedure pokaz_pola(ile_x,ile_y:byte);
var
  x,y:byte;
begin
  for x:=1 to ile_x do
    for y:=1 to ile_y do
      begin
        gotoxy(x,y);
        case pole[x,y] of
        {  0,10:write(' ');
          1..8:write(pole[x,y]);
        }
             9:write('Ű');
        end;
      end;
```

```
end;

function czy_koniec(ile_x,ile_y:byte):boolean;
var
  x,y:byte;
begin
  for x:=1 to ile_x do
    for y:=1 to ile_y do
      begin

      end;
end;

procedure gramy(ile_x,ile_y,ile_min:byte);
var
  x,y:byte;
  znak:char;
  koniec:boolean;

begin
  x:=1;
  y:=1;
  koniec:=false;
  repeat
    gotoxy(x,y);
    czysc_bufor;
    znak:=readkey;
    case znak of
     ' ',#13:case pole[x,y] of
               1..8:write(pole[x,y]);
                  9:begin
                      pokaz_pola(ile_x,ile_y);
                      gotoxy(x,y);
                      textcolor(7+blink);
                      write('Ű');
                      koniec:=true;
                      textcolor(7);
                    end;
                  0:zapal_puste(x,y);
             end;
         #27:koniec:=true;
     'm',#9 :write('Ű');
         #8:write('ţ');
         #0 :begin
               znak:=readkey;
               case znak of
          {l}    #75:if x>1     then dec(x);
          {p}    #77:if x<ile_x then inc(x);
          {g}    #72:if y>1     then dec(y);
          {d}    #80:if y<ile_y then inc(y);
               end;
            end;
    end;

  until koniec;
end;

procedure info;
begin
  writeln('           SKOCZEK');
  writeln('');
```

```
  writeln('Wacek Libront   Bobowa 1995');
  writeln('');
  writeln('kursory - poruszanie saperem');
  writeln('spacja  - odminowanie');
  writeln('TAB     - zaznaczenie miny');
  writeln('BS      - cofni©cie zaznaczenia');
  writeln('ESC     - koniec');

end;

begin
  ile_x:=8;
  ile_y:=8;
  ile_min:=10;
  clrscr;
  info;
  writeln;
  wybieraj(ile_x,ile_y,ile_min);
  clrscr;
 { TextMode(C80 + Font8x8);}
  window(10,10,10+ile_x,10+ile_y+3);
  rysuj_pole(ile_x,ile_y);
  wyczysc(ile_x,ile_y);
  minuj(ile_x,ile_y,ile_min);
  opisz_miny_3(ile_x,ile_y);
  gramy(ile_x,ile_y,ile_min);
  gotoxy(1,ile_y+2);
  writeln('koniec');
  readln;
  textmode(lastmode);
end.
```

Borland Delphi - 1997 - wersja MS Windows




```
unit saper_1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Grids, Menus, ExtCtrls, StdCtrls, Buttons;

type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    Gra1: TMenuItem;
    Pomoc1: TMenuItem;
    Nowa1: TMenuItem;
    N1: TMenuItem;
    Pocztkujcy1: TMenuItem;
    Zaawansowany1: TMenuItem;
    Ekspert1: TMenuItem;
    Uytkownik1: TMenuItem;
    N2: TMenuItem;
    Najlepszewyniki1: TMenuItem;
    N3: TMenuItem;
    Koniec1: TMenuItem;
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    SpeedButton1: TSpeedButton;
    Panel4: TPanel;
    Panel5: TPanel;
    DrawGrid1: TDrawGrid;
    Image2: TImage;
    Image1: TImage;
    Timer1: TTimer;
    Image3: TImage;
    Image4: TImage;
    Image5: TImage;
    Image6: TImage;
    Image7: TImage;
    Image8: TImage;
    Image9: TImage;
    Image10: TImage;
```

```
    Image11: TImage;
    Image12: TImage;
    Image13: TImage;
    Image14: TImage;
    Image15: TImage;
    Image16: TImage;
    Image17: TImage;
    Sprawdzian1: TMenuItem;
    N4: TMenuItem;
    procedure przerysuj;
    procedure FormCreate(Sender: TObject);
    procedure Pocztkujcy1Click(Sender: TObject);
    procedure Zaawansowany1Click(Sender: TObject);
    procedure Ekspert1Click(Sender: TObject);
    procedure Nowa1Click(Sender: TObject);
    procedure Uytkownik1Click(Sender: TObject);
    procedure DrawGrid1MouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure wyniki;
    procedure WpiszOcene;
    procedure rys_przycisk(k,w,num:integer);
    procedure ustal_tablice;
    procedure rysuj_pole(k,w:integer);
    procedure rysuj_cala_tablice;
    procedure DrawGrid1MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure DrawGrid1MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure Timer1Timer(Sender: TObject);
    procedure zapal_puste(x,y:byte);
    function czy_koniec:boolean;
    procedure Najlepszewyniki1Click(Sender: TObject);
    procedure Koniec1Click(Sender: TObject);
    procedure DrawGrid1DrawCell(Sender: TObject; Col, Row: Longint;
      Rect: TRect; State: TGridDrawState);
    procedure Sprawdzian1Click(Sender: TObject);
    procedure DrawGrid1KeyPress(Sender: TObject; var Key: Char);
    procedure Pomoc1Click(Sender: TObject);
    procedure Panel2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

  TTablica=array [0..29 ,0..23] of integer;

var
  Form1: TForm1;
  NrSprawdzian, {numer rozwiązanych sprawdzianów}
  poziom,   {jaki wybrano poziom gry}
  ile_min,  {licznik zaznaczonych min}
  SprCzas,
  MaxCzas,  {maksymalny czas pracy}
  czas,     {licznik czasu gry}
  kolumny,  {ile kolumn (od 1)}
  wiersze,  {ile wierszy (od 1)}
  miny:integer; {ile min}
               {kolumny,wiersze}
  {rodzaj pól w tablicy
      0-puste pole przed badaniem
```

```
   1..8-wokół pola jest mina
      9-pole z miną
     10-puste pole po badaniu
 11..18-pole z chorągiewką, obok jest mina
     19-pole z chorągiewką na minie
     20-puste pole z chorągiewką}
  tablica:TTablica;
  {jeśli true to pole było wybrane - wciśnięty przycisk}
  tablica_1:array [0..29 ,0..23] of boolean;
  pauza,
  CzySprawdzian,    {jeśli mamy sprawdzian}
  bum,           {trafiona mina i koniec}
  pierwszy,      {naciśnięto pierwszy przycisk - do liczenia czasu}
  wcisk:boolean; {przycisk wciśnięty i przesunięty- true}
  wcisk_k,
  wcisk_w:integer; {kolumna i wiersz pola które wciśnięto}
  {tablice z rekordowymi wynikami}
  rek_c:array[0..6] of integer;
  rek_n:array[0..6] of string;
  nazwisko:string; {wpisane po nowym rekordzie}

implementation

uses saper2,saper3,saper4,saper6,saper7;

{$R *.DFM}

{rekordy zapisane na dysk}
procedure ZapiszRekordy;
var
  plik:TextFile;
  i:integer;
begin
  AssignFile(plik,'Saper.rek');
  Rewrite(plik);
  For i:=0 to 6 do
    Writeln(plik,rek_c[i],' ',rek_n[i]);
  CloseFile(plik);
end;

{rekordy odczytane z dysku i sprawdzone}
procedure CzytajRekordy;
var
  plik:TextFile;
  i:integer;
begin
  AssignFile(plik,'Saper.rek');
  {$I-}
  Reset(plik);
  {$I+}
  if IOResult = 0 then
  begin {gdyby nie było pliku to stworzyć rekordy}
    For i:=0 to 6 do
    begin
      {na wszelki wypadek przed wczytaniem}
      rek_c[i]:=MaxCzas; rek_n[i]:='Anonim';
      Readln(plik,rek_c[i],rek_n[i]);
    end;
    CloseFile(plik);
  end;
end;
```

```
{tablice wyzerowane - 0 i false}
procedure zeruj_tablice;
var
  w,k:integer;
begin
  for w:=0 to wiersze-1 do
    for k:=0 to kolumny-1 do
    begin
      tablica[k,w]:=0;
      tablica_1[k,w]:=false;
    end;
end;

{losowanie ułożenia min}
procedure minuj(k,w,m:byte);
var
  x,y,i:byte;
begin
  randomize;
  for i:=1 to m do
    begin {tyle losuje aż trafi na puste pole}
      repeat
        x:=random(k);
        y:=random(w);
      until tablica[x,y]<>9;
      tablica[x,y]:=9;
    end;
end;

{opisanie pól niezaminowanych cyframi 0..8}
procedure opisz_miny(k,w:byte);
const
  sx:array[1..8] of shortint =(+1,+1,+1, 0,-1,-1,-1, 0);
  sy:array[1..8] of shortint =(+1, 0,-1,-1,-1, 0,+1,+1);
var
  x,y,i:byte;

  function czy_pole(x,y:byte):boolean;
  begin
    if (x<0) or (x>k-1) or
       (y<0) or (y>w-1)
      then czy_pole:=false
      else czy_pole:=true;
  end;

begin
  for x:=0 to k-1 do
    for y:=0 to w-1 do
      if tablica[x,y]<>9 then
        for i:=1 to 8 do
          if czy_pole(x+sx[i],y+sy[i]) then
            {jeśli obok pola mina to zwiększamy tablicę o jeden}
            if tablica[x+sx[i],y+sy[i]]=9 then
              inc(tablica[x,y]);
end;

{koniec jeśli liczba wciśniętych pól
 jest równa polom bez min}
Function TForm1.czy_koniec:boolean;
var
```

```
  k,w,l:integer;
begin
  l:=0;
  if not(bum) then {nie sprawdzamy jeśli puknięto w minę}
  for k:=0 to kolumny-1 do
    for w:=0 to wiersze-1 do
      if tablica_1[k,w] then inc(l);
  if wiersze*kolumny-miny=l
    then czy_koniec:=true
    else czy_koniec:=false;
end;

{jeśli puknięto w puste pole,
 to zapalamy puste pola przylegające do niego - rekurencja}
procedure TForm1.zapal_puste(x,y:byte);
const
  sx:array[1..8] of shortint =(+1,+1,+1, 0,-1,-1,-1, 0);
  sy:array[1..8] of shortint =(+1, 0,-1,-1,-1, 0,+1,+1);
var
  i,x1,y1:Byte;

  function czy_pole(x,y:byte):boolean;
  begin
  if (x<0) or (x>kolumny-1) or
     (y<0) or (y>wiersze-1)
    then czy_pole:=false
    else czy_pole:=true;
  end;

begin
  {poprzednio 0, ale wstawiamy 10
   bo by się rekurencja nie skończyła}
  tablica[x,y]:=10;
  rys_przycisk(x,y,0);
  {sprawdzamy osiem kierunków}
  for i:=1 to 8 do
  begin
    x1:=x+sx[i];
    y1:=y+sy[i];
    if czy_pole(x1,y1) then
      if tablica[x1,y1]=0 then
        begin {jeśli następne pole puste}
          tablica_1[x1,y1]:=true;
          zapal_puste(x1,y1);    {tu rekurencja !}
        end
        else if tablica[x1,y1] in [1..8] then
             begin  {jeśli pole z numerem}
               rys_przycisk(x1,y1,tablica[x1,y1]);
               tablica_1[x1,y1]:=true;
             end;
  end;
end;

{rysujemy przycisk o podanym numerze}
procedure TForm1.rys_przycisk(k,w,num:integer);
var
  cela:TRect;
begin
  with DrawGrid1 do
  begin
    cela:=CellRect(k,w);
```

```
    case num of
     -1:Canvas.Draw(cela.left,cela.Top,Image2.Picture.Bitmap);{przycisk
pusty}
      0:Canvas.Draw(cela.left,cela.Top,Image1.Picture.Bitmap);{puste pole}
      1:Canvas.Draw(cela.left,cela.Top,Image3.Picture.Bitmap);{przyciski z
numerami}
      2:Canvas.Draw(cela.left,cela.Top,Image4.Picture.Bitmap);
      3:Canvas.Draw(cela.left,cela.Top,Image5.Picture.Bitmap);
      4:Canvas.Draw(cela.left,cela.Top,Image6.Picture.Bitmap);
      5:Canvas.Draw(cela.left,cela.Top,Image7.Picture.Bitmap);
      6:Canvas.Draw(cela.left,cela.Top,Image8.Picture.Bitmap);
      7:Canvas.Draw(cela.left,cela.Top,Image9.Picture.Bitmap);
      8:Canvas.Draw(cela.left,cela.Top,Image10.Picture.Bitmap);
      9:Canvas.Draw(cela.left,cela.Top,Image11.Picture.Bitmap);{przycisk
bombą}
     10:Canvas.Draw(cela.left,cela.Top,Image12.Picture.Bitmap);{przycisk z
chorągiewką}
     11:Canvas.Draw(cela.left,cela.Top,Image17.Picture.Bitmap);{mmina
rozbrojona}
    end;
  end;
end;

{rysowanie konkretnego pola tablicy}
procedure TForm1.rysuj_pole(k,w:integer);
begin
  case tablica[k,w] of
          0:rys_przycisk(k,w,-1);
       1..8:if tablica_1[k,w]
              then rys_przycisk(k,w,tablica[k,w])
              else rys_przycisk(k,w,-1);
       9   :if tablica_1[k,w]
              then rys_przycisk(k,w,9)
              else rys_przycisk(k,w,-1);
       10  :if tablica_1[k,w]
              then rys_przycisk(k,w,0)
              else rys_przycisk(k,w,10);
     11..18:rys_przycisk(k,w,10);
         19:if tablica_1[k,w]
              then rys_przycisk(k,w,11)
              else rys_przycisk(k,w,10);
    end;
end;

{przerysowanie całej tablicy - dowolny stan pól}
procedure TForm1.rysuj_cala_tablice;
var
  w,k:integer;
begin
  for w:=0 to wiersze-1 do
    for k:=0 to kolumny-1 do
      rysuj_pole(k,w);
end;

{przerysowanie formularza po zmianie wymiarów}
procedure TForm1.przerysuj;
begin
  Form1.Visible:=false;
  with DrawGrid1 do
  begin
    Col:=0;
```

```
    Row:=0;
    colCount:=kolumny;
    RowCount:=wiersze;
    {rozmiary i położenie poszczególnych komponentów}
    Form1.ClientHeight:=(DefaultRowHeight+1)*RowCount+10+
                        Panel2.Height;
    Form1.ClientWidth:=(DefaultColWidth+1)*ColCount+10;
    Panel4.Left:=Panel2.left+8;
    Panel5.Left:=Panel2.Left+Panel2.Width-(Panel5.Width+10);
    SpeedButton1.Left:=Panel2.Left+(Panel2.Width div 2)-
                       (SpeedButton1.Width div 2);
    Form1.Position:=poScreenCenter;
  end;
  Form1.Visible:=true;
  Panel4.Font.Color:=clRed;
  Panel4.Caption:=inttostr(ile_min);
  Panel5.Caption:=inttostr(czas);
end;

{początkowe operacje na tablicach i zmiennych}
procedure TForm1.ustal_tablice;
begin
  zeruj_tablice;
  minuj(kolumny,wiersze,miny);
  opisz_miny(kolumny,wiersze);
  ile_min:=miny;
  pierwszy:=false;
  czas:=0;
  bum:=false;
  Timer1.Enabled:=true;
end;

{rozpoczyna początkujący}
procedure TForm1.Pocztkujcy1Click(Sender: TObject);
begin
  CzySprawdzian:=false;
  kolumny:=8;
  wiersze:=8;
  miny:=10;
  poziom:=1;
  MaxCzas:=999;
  ustal_tablice;
  przerysuj;
end;

{rozpoczyna zaawansowany}
procedure TForm1.Zaawansowany1Click(Sender: TObject);
begin
  CzySprawdzian:=false;
  kolumny:=16;
  wiersze:=16;
  miny:=40;
  poziom:=2;
  MaxCzas:=999;
  ustal_tablice;
  przerysuj;
end;

{rozpoczyna ekspert}
procedure TForm1.Ekspert1Click(Sender: TObject);
begin
```

```
  CzySprawdzian:=false;
  kolumny:=30;
  wiersze:=16;
  miny:=99;
  poziom:=3;
  MaxCzas:=999;
  ustal_tablice;
  przerysuj;
end;

{plansza od nowa - te same wymiary}
procedure TForm1.Nowa1Click(Sender: TObject);
begin
  if CzySprawdzian then MaxCzas:=czas;
  ustal_tablice;
  Panel4.Font.Color:=clRed;
  Panel4.Caption:=inttostr(ile_min);
  Panel5.Caption:=inttostr(czas);
  SpeedButton1.Glyph:=Image13.Picture.Bitmap;
  rysuj_cala_tablice;
  if CzySprawdzian then czas:=MaxCzas;
end;

{rozpoczyna użytkownik}
procedure TForm1.Uytkownik1Click(Sender: TObject);
begin
  Form2.showmodal;
  poziom:=0;
  MaxCzas:=999;
  ustal_tablice;
  przerysuj;
end;

{początkowe ustawienia po starcie}
procedure TForm1.FormCreate(Sender: TObject);
begin
  kolumny:=8;
  wiersze:=8;
  miny:=10;
  poziom:=1;
  MaxCzas:=999;
  SprCzas:=600;
  CzySprawdzian:=false;
  pauza:=false;
  nazwisko:='Anonim';
  with DrawGrid1 do
  begin
    DefaultColWidth:=20;
    DefaultRowHeight:=20;
  end;
  ustal_tablice;
  CzytajRekordy;

  {nie można tutaj przerysować bo głupieje
   formularz jest dopiero tworzony}
end;

{wyniki do formularza 7}
procedure TForm1.Wyniki;
begin
  With Form7.Panel1 do
```

```
  case NrSprawdzian of
   0,1:Caption:='niedostateczny';
     2:Caption:='mierny';
     3:Caption:='dostateczny';
     4:Caption:='dobry';
     5:Caption:='bardzo dobry';
     6:Caption:='celujący';
   end;
   CzySprawdzian:=false;
   Timer1.Enabled:=false;
   {jeśli rekord sprawdzianu - ale na 6}
   if SprCzas-czas < rek_c[4]then
   begin
     rek_c[4]:=SprCzas-czas;
     rek_n[4]:=nazwisko;
   end;
   {gdy nie zmieścił się w czasie to zapisana ocena}
   if NrSprawdzian<6 then czas:=SprCzas-NrSprawdzian;
   rek_c[5]:=SprCzas-czas;
   rek_n[5]:=nazwisko;
   ZapiszRekordy;
   Form7.ShowModal;
   MainMenu1.Items[0].Enabled:=true;
   MainMenu1.Items[1].Enabled:=true;
   borderStyle:=bsSingle;
   poziom:=1;
   MaxCzas:=999;
   ustal_tablice;
   przerysuj;
end;

{wpisujemy ocenę do leeewego panelu w kolorze zielonym}
procedure TForm1.wpiszOcene;
begin
  Panel4.Font.Color:=clLime;
  case NrSprawdzian of
  0,1:Panel4.Caption:='ndt';
    2:Panel4.Caption:='mrn';
    3:Panel4.Caption:='dst';
    4:Panel4.Caption:='db';
    5:Panel4.Caption:='bdb';
    6:Panel4.Caption:='cel';
  end;
end;

{puszczono przycisk myszy}
procedure TForm1.DrawGrid1MouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
var
  kol,wie:longint;
  cz:integer;
  {wymazywanie chorągiewki}
  procedure UsunChor;
  begin
    inc(ile_min);
    Panel4.Font.Color:=clRed;
    Panel4.Caption:=inttostr(ile_min);
    tablica[kol,wie]:=tablica[kol,wie]-10;
    rys_przycisk(kol,wie,-1)
  end;
  {rysowanie chorągiewki}
```

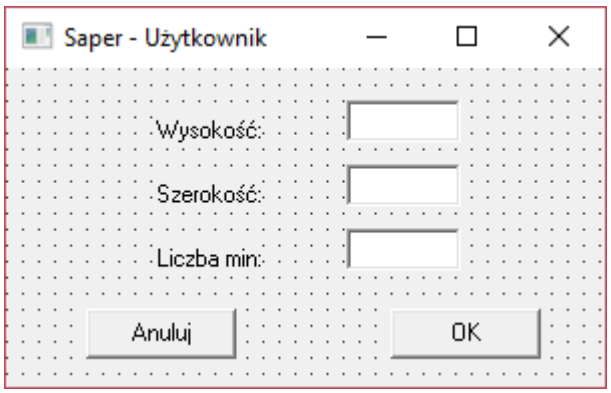
```
  procedure RysujChor;
  begin
    dec(ile_min);
    Panel4.Font.Color:=clRed;
    Panel4.Caption:=inttostr(ile_min);
    tablica[kol,wie]:=tablica[kol,wie]+10;
    rys_przycisk(kol,wie,10)
  end;

begin
  {puszczono przycisk}
  wcisk:=false;
  SpeedButton1.Glyph:=Image13.Picture.Bitmap;
  {pole na którym puszczono przycisk}
  DrawGrid1.MouseToCell(X, Y, kol, wie);
  {wyprane pole jeśli jest w tablicy}
  if (kol in [0..kolumny-1]) and (wie in [0..wiersze-1]) then
   {gdy wciśnięto prawy przycisk a pole nie badane to chorągiewka}
   if (button=mbRight) and (tablica_1[kol,wie]=false) then
    {jeżeli pole z chorągiewką to usuwamy chorągiewkę}
    {jeśli nie było chorągiewki to ją ustawiamy}
    if tablica[kol,wie]>=10 then UsunChor
    else RysujChor
  else
  {lewym przyciskiem puknięto w chorągiewkę
   to tylko usuwamy znacznik}
  if (tablica[kol,wie]>=10) and (tablica_1[kol,wie]=false) then
    UsunChor
  else
  {tu wreszcie normalna obsługa puknięcia}
  begin
    {jeśli rozpoczynamy}
    if not(pierwszy) then pierwszy:=true;
    {pole zostało wybrane}
    tablica_1[kol,wie]:=true;
    {jaki przycisk zostanie odsłonięty}
    case tablica[kol,wie] of
    0   :zapal_puste(kol,wie);
    1..8:rys_przycisk(kol,wie,tablica[kol,wie]);
    9   :begin
           {trafienie w bombę blokuje resztę bomb}
           bum:=true;
           {zła mordka do przycisku}
           SpeedButton1.Glyph:=Image15.Picture.Bitmap;
           {nie liczymy czasu gdy nie ma sprawdzianu}
           if not(CzySprawdzian) then Timer1.Enabled:=false
           else WpiszOcene;
           for wie:=0 to wiersze-1 do
             for kol:=0 to kolumny-1 do
               {te z chorgiewką też trzeba odkryć}
               if tablica[kol,wie] mod 10 = 9 then
               begin
                 tablica_1[kol,wie]:=true;
                 rys_przycisk(kol,wie,9);
               end;
         end;
    end; {case}
  end; {sprawdzenie czy puszczono w tablicy}
  {sprawdzenie czy odkryte wszystkie pola}
  if czy_koniec then
  begin
```

```
    {mordka w okularach do przycisku}
    SpeedButton1.Glyph:=Image16.Picture.Bitmap;
    {ilość min wpisana do wyświetlacza}
    Panel4.Font.Color:=clRed;
    Panel4.Caption:=IntToStr(miny);
    for wie:=0 to wiersze-1 do
      for kol:=0 to kolumny-1 do
      begin
        tablica_1[kol,wie]:=true;
        {uzupełniamy nie odkryte bez chorągiewek}
        if tablica[kol,wie]=9 then tablica[kol,wie]:=19;
        {przyciski z chorągiewkami zapalamy minami}
        if tablica[kol,wie]=19 then rys_przycisk(kol,wie,11);
      end;

    if Not(CzySprawdzian) then Timer1.Enabled:=false;
    {jeśli czas rekordowy}
    if czas < rek_c[poziom] then
    begin
      if not(CzySprawdzian) then
        begin {gdy nie jest to sprawdzian to pytaj o nazwisko}
          Form4.Caption:='Gratulacje - nowy rekord !!!';
          Form4.ShowModal;
        end;
      rek_c[poziom]:=czas;
      rek_n[poziom]:=nazwisko;
      {po wpisaniu zachowujemy nowe rekordy}
      ZapiszRekordy;
    end;
    if CzySprawdzian then
      begin
        inc(NrSprawdzian);
        WpiszOcene;
        if NrSprawdzian=6 then wyniki;
      end;
  end; {CzyKoniec=true odkryte wszystkie pola}
end;

{wciśnięto przycisk myszy}
procedure TForm1.DrawGrid1MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
var
  kol,wie:longint;
begin
  {gdzie wciśnięto przycisk myszy}
  DrawGrid1.MouseToCell(X, Y, kol, wie);
  {zdziwiona morda do przycisku}
  SpeedButton1.Glyph:=Image14.Picture.Bitmap;
  {wymazujemy tylko pole nie odkryte}
  if tablica_1[kol,wie] = false then
    rys_przycisk(kol,wie,0);
  {zapamiętujemy fakt wciśnięcia przycisku}
  wcisk:=true;
  wcisk_k:=kol;
  wcisk_w:=wie;
end;

{mysza jest przesuwana z wciśniętym przyciskiem}
procedure TForm1.DrawGrid1MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
var
```

```
  kol,wie:longint;
begin
  {badamy tylko wtedy, gdy przycisk wciśnięty}
  if wcisk then
  begin
    {odczytujemy nowe położenie myszy}
    DrawGrid1.MouseToCell(X, Y, kol, wie);
    {jeżeli mysza przesunęła się na nowe pole}
    if (kol<>wcisk_k) or (wie<>wcisk_w) then
    begin
      {na starym miejscu spowrotem przycisk jeśli nie odkryte}
      if tablica_1[wcisk_k,wcisk_w]=false then
        {jeszcze sprawdzenie czy nie bya chorgggiewka}
        if tablica[wcisk_k,wcisk_w]>=10
          then rys_przycisk(wcisk_k,wcisk_w,10)
          else rys_przycisk(wcisk_k,wcisk_w,-1);
      {na nowym miejscy wymazujemy jeśli nie odkryte
       i jeśli mysza na tablicy}
      if tablica_1[kol,wie] =false then
        if (kol>=0) and (wie>=0) then
          rys_przycisk(kol,wie,0);
      {zapamiętujemy nowe położenie myszy}
      wcisk_k:=kol;
      wcisk_w:=wie;
    end;
  end;
end;

{odliczanie czasu trwania gry}
procedure TForm1.Timer1Timer(Sender: TObject);
begin

  if not(CzySprawdzian) then
  begin
    {tylko do 999 sekund i gdy naciśnięto pierwszy raz}
    if (czas < MaxCzas) and (pierwszy) then inc(czas)
  end
  else
    if czas=0 then wyniki
    else dec(czas);
  Panel5.Caption:=inttostr(czas);
end;

{Formularz z rekordami - najpierw wpisujemy do pół dane}
procedure TForm1.Najlepszewyniki1Click(Sender: TObject);
var s:string;
begin
  Form3.Label4.Caption:=inttostr(rek_c[1])+' sek.';
  Form3.Label5.Caption:=inttostr(rek_c[2])+' sek.';
  Form3.Label6.Caption:=inttostr(rek_c[3])+' sek.';
  if rek_c[4]<=6 then s:='' else s:=' sek.';
  Form3.Label11.Caption:=inttostr(rek_c[4])+s;
  if rek_c[5]<=6 then s:='' else s:=' sek.';
  Form3.Label14.Caption:=inttostr(rek_c[5])+s;
  Form3.Label7.Caption:=rek_n[1];
  Form3.Label8.Caption:=rek_n[2];
  Form3.Label9.Caption:=rek_n[3];
  Form3.Label12.Caption:=rek_n[4];
  Form3.Label15.Caption:=rek_n[5];
  Form3.ShowModal;
  ZapiszRekordy;
```

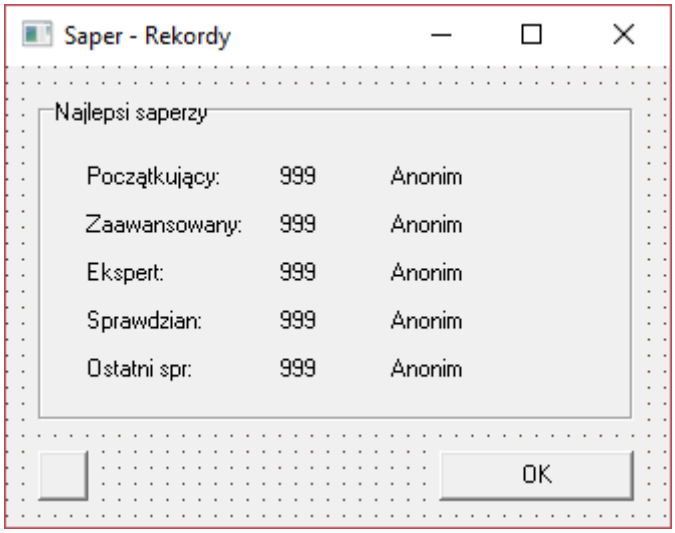
```
end;

{koniec sapera}
procedure TForm1.Koniec1Click(Sender: TObject);
begin
  Application.Terminate;
end;

{informacja o autorze}
{rysuje tylko te komórki które potrzeba
 i reaguje na gaszenie w czasie przesuwania}
procedure TForm1.DrawGrid1DrawCell(Sender: TObject; Col, Row: Longint;
  Rect: TRect; State: TGridDrawState);
begin
  {przerysuj tylko wtedy, gdy przycisk myszy nie wciśnięty}
  if not(wcisk) then rysuj_pole(col,row);
end;

procedure TForm1.Sprawdzian1Click(Sender: TObject);
begin
  {if MessageDlg('Puknij w OK a rozpoczniesz sprawdzian (6 min)'
             ,mtWarning,[mbOK,mbCancel],0) = mrOK then}
  Form4.Caption:='Sprawdzian - 10 minut !!!';
  Form4.ShowModal;
  if nazwisko<>'' then
  begin
    CzySprawdzian:=true;
    NrSprawdzian:=0;
    kolumny:=8;
    wiersze:=8;
    miny:=10;
    poziom:=4;
    ustal_tablice;
    MainMenu1.Items[0].Enabled:=false;
    MainMenu1.Items[1].Enabled:=false;
    borderStyle:=bsNone;
    przerysuj;
    MaxCzas:=SprCzas+1;
    czas:=MaxCzas;
  end;
end;

procedure TForm1.DrawGrid1KeyPress(Sender: TObject; var Key: Char);
begin
  {jeśli sprawdzian to nie ma pauzy}
  if not(CzySprawdzian) then
  if upcase(key)='P' then
  begin
    pauza:=not(pauza);
    {do póki saper jest aktywny to da się wrócić
     ale można tą procedurę wstawić do Form1KeyPress
     bo formularz jest aktywny}
    if pauza then
     begin
       Timer1.Enabled:=false;
       DrawGrid1.Visible:=false;
       SpeedButton1.Enabled:=false;
       MainMenu1.Items[0].Enabled:=false;
       MainMenu1.Items[1].Enabled:=false;
     end
    else
```

```
      begin
        Timer1.Enabled:=true;
        DrawGrid1.Visible:=true;
        SpeedButton1.Enabled:=true;
        MainMenu1.Items[0].Enabled:=true;
        MainMenu1.Items[1].Enabled:=true;
      end;
  end;
end;

procedure TForm1.Pomoc1Click(Sender: TObject);
begin
  Form6.ShowModal;
end;

procedure TForm1.Panel2Click(Sender: TObject);
begin

end;

end.
```



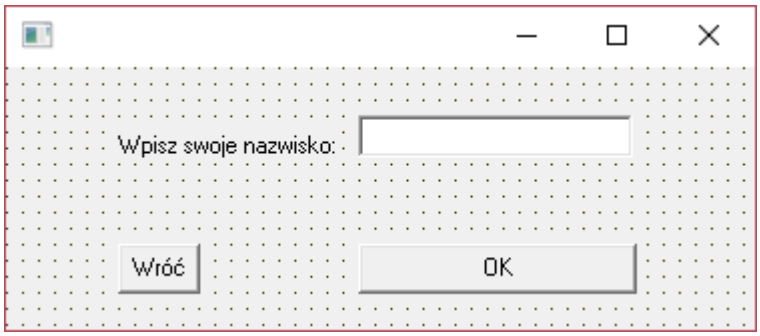

```
unit saper2;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TForm2 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Label3: TLabel;
    Edit3: TEdit;
    Button1: TButton;
    Button2: TButton;
    procedure Button2Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure przypisz(w,s,m:integer);
    procedure odbierz(var w,s,m:integer);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
```

```
    { Public declarations }
  end;

var
  Form2: TForm2;
  wys,
  sze,
  min:integer;

implementation
uses saper_1;
{$R *.DFM}

procedure TForm2.przypisz(w,s,m:integer);
begin
  wys:=w;
  sze:=s;
  min:=m;
end;

procedure TForm2.odbierz(var w,s,m:integer);
begin
  w:=wys;
  s:=sze;
  m:=min;
end;

procedure TForm2.Button1Click(Sender: TObject);
var w,s,m:integer;
begin
  w:=StrToInt(Edit1.Text);
  if w in [8..24] then wys:=w else wys:=24;
  s:=StrToInt(Edit2.Text);
  if s in [8..30] then sze:=s else sze:=30;
  m:=StrToInt(Edit3.Text);
  if m in [1..wys*sze] then min:=m else min:= (wys*sze) div 5;
  odbierz(wiersze,kolumny,miny);
  close;
end;

procedure TForm2.Button2Click(Sender: TObject);
begin
  close;
end;

procedure TForm2.FormCreate(Sender: TObject);
begin
  przypisz(wiersze,kolumny,miny);
  Edit1.Text:=intToStr(wys);
  Edit2.Text:=intToStr(sze);
  Edit3.Text:=intToStr(min);
end;

end.
```



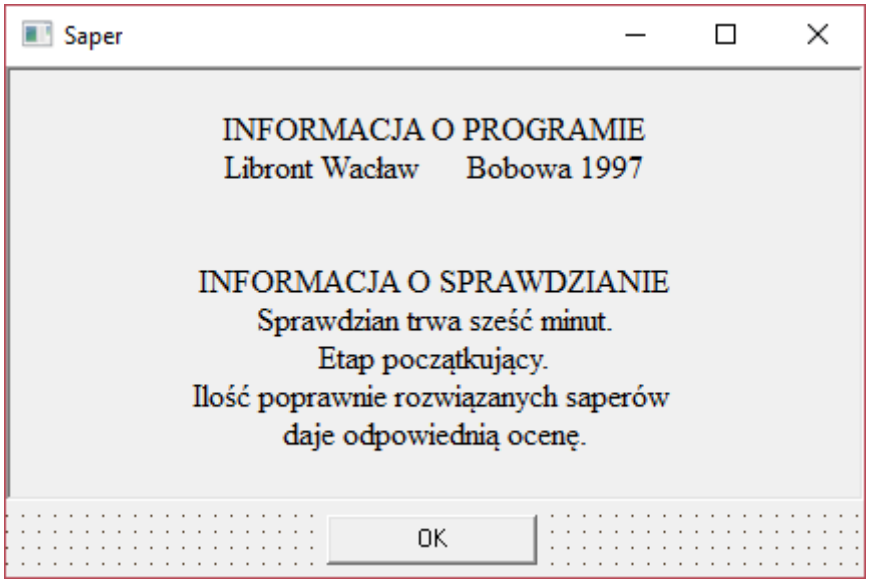

```
unit saper3;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TForm3 = class(TForm)
    GroupBox1: TGroupBox;
    Button1: TButton;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Button2: TButton;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    procedure Button1Click(Sender: TObject);
    procedure Button2MouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form3: TForm3;



implementation
uses saper_1;
```

```
{$R *.DFM}

procedure TForm3.Button1Click(Sender: TObject);
begin
  close;
end;

procedure TForm3.Button2MouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  if (button=mbRight) and (shift=[ssShift]) then
  begin
    rek_c[1]:=MaxCzas;rek_n[1]:='Anonim';
    rek_c[2]:=MaxCzas;rek_n[2]:='Anonim';
    rek_c[3]:=MaxCzas;rek_n[3]:='Anonim';
    rek_c[4]:=SprCzas;rek_n[4]:='Anonim';
    rek_c[5]:=SprCzas;rek_n[5]:='Anonim';
  end;
  Close
end;

end.
```




```
unit saper4;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TForm4 = class(TForm)
    Label1: TLabel;
    Edit1: TEdit;
    Button1: TButton;
    Button2: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form4: TForm4;

implementation
uses saper_1;
```

```
{$R *.DFM}

procedure TForm4.Button1Click(Sender: TObject);
begin
  nazwisko:=Edit1.Text;
  close
end;

procedure TForm4.Button2Click(Sender: TObject);
begin
  nazwisko:='';
  close;
end;

end.
```




```
unit saper6;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls;

type
  TForm6 = class(TForm)
    Memo1: TMemo;
    Button1: TButton;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form6: TForm6;

implementation

{$R *.DFM}
```

```
procedure TForm6.Button1Click(Sender: TObject);
begin
  close
end;

end.
```


```
unit saper7;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls;

type
  TForm7 = class(TForm)
    Panel1: TPanel;
    procedure Button1Click(Sender: TObject);
    procedure Panel1DblClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form7: TForm7;

implementation
uses saper_1;
{$R *.DFM}

procedure TForm7.Button1Click(Sender: TObject);
begin
  close
end;

procedure TForm7.Panel1DblClick(Sender: TObject);
begin
  close;
end;

end.
```