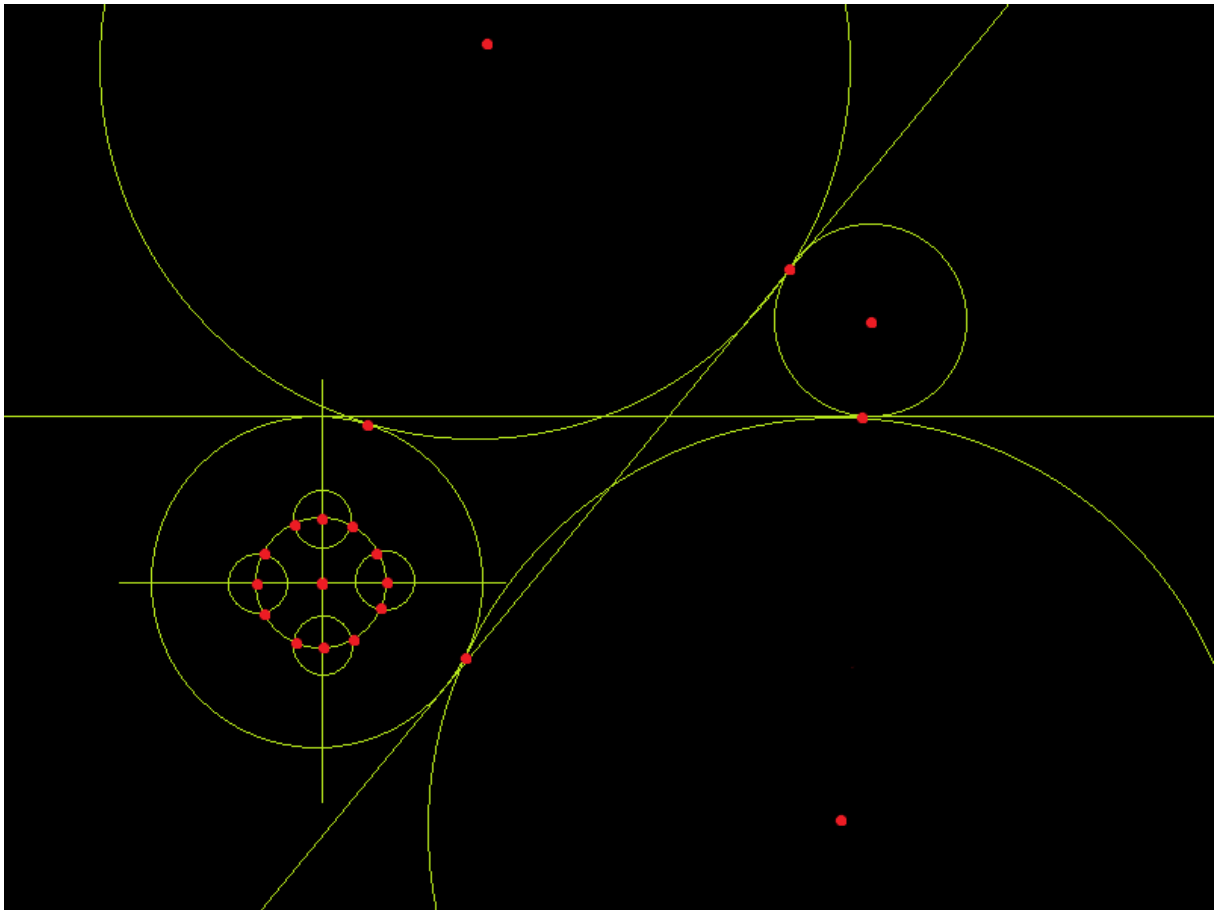


Cyfrowe projekty – PROGRAM OBLICZEŃ GEOMETRYCZNYCH



Niepozorny układ kresek i kółek, a w tle skomplikowany aparat matematyczny, pozwalający obliczać punkty przecięć i styczności geometrycznych obiektów. Po co to wszystko? Tych informacji potrzebuje każdy inżynier, który programuje obrabiarki, które to wykonują wszystkie te cudowne przedmioty, bez których nie wyobrażamy sobie codziennego życia. 1500 wierszy kodu w Pascalu, prawie 45000 znaków umożliwia dokonanie przeliczeń geometrycznych w celu znalezienia współrzędnych punktów na podstawie zdefiniowanej konfiguracji elementów podstawowych, jak: punkty, linie i okręgi.

W czasach, gdy komputery torowały sobie drogę do przemysłu, gdy standardem była klawiatura i monochromatyczny monitor (jednokolorowy), gdy słowo "windows" jeszcze nikomu nie kojarzyło się z dzisiejszymi systemami operacyjnymi, czyli w roku 1990 istniały już co prawda programy do projektowania (CAD) lecz były bardzo drogie i niedostępne w krajach tzw. bloku państwa socjalistycznych. Mój program POG z roku 1990, który powstał w wojskowych Zakładach Mechanicznych w Tarnowie próbuje załatać tę dziurę.

Definiowanie konfiguracji, czyli rysowanie punktów, linii i okręgów odbywa się za pomocą trzyliterowych kodów wpisywanych z klawiatury:

- PXY-punkt we współrzędnych prostokątnych X i Y
- PRQ-punkt we współrzędnych biegunowych R i Q

- PLC-punkt przecięcia prostej L i okręgu C
- PLL-punkt przecięcia dwóch prostych L1 i L2
- PCC-punkt przecięcia dwóch okręgów C1 i C2
- PSC-punkt jako środek okręgu C
- PLI-zbiór punktów na prostej
- PCR-zbiór punktów na okręgu
- TRA-zbiór punktów jako przesunięcie równoległe
- ROT-zbiór punktów po obrocie
- SYM-zbiór punktów jako odbicie symetryczne
- LPP-prosta przechodząca przez dwa punkty P1 i P2
- LPC-prosta przechodząca przez punkt P styczna do okręgu C
- LLD-prosta odległa od prostej L o odległość D
- LPQ-prosta przechodząca przez punkt P tworząca z prostą kąt Q
- LCQ-prosta styczna do okręgu C tworząca z osią X kąt Q
- LCC-prosta styczna do dwóch okręgów C1 i C2
- CPR-okrąg o środku w punkcie P i promieniu R
- CPL-okrąg styczny do prostej L przechodzący przez punkt P
- CLL-okrąg styczny do dwóch prostych L1 i L2
- CPP-okrąg przechodzący przez dwa punkty P1 i P2
- CLC-okrąg styczny do prostej L i okręgu C
- CCC-okrąg styczny do dwóch okręgów C1 i C2
- CPC-okrąg przechodzący przez punkt P styczny do okręgu C

Dodatkowo zdefiniowano szereg funkcji specjalnych

- THH-wywołanie ekranu pomocy
- TSA-zapis danych na dysk
- TLO-odczyt z dysku zbioru z danymi
- TDD-wydruk współrzędnych
- TWW-dane do dziurkowania na dysk (\$)
- TSS-przesuwanie i skalowanie ekranu
- TPP-odległość między dwoma punktami
- TTT-współrzędne pojedynczego elementu
- TRR-wyzerowanie danych
- T**-wyjście do systemu operacyjnego
- TNP(L,C)-zmiana numeracji
- TBP(L,C)-wywołanie danych z bufora
- TMP(L,C)-wywołanie danych z tablicy głównej
- <F11>-numeracja wszystkich elementów
- <F12>-wywołanie ostatniego bufora
- <F13>-wywołanie ostatniej tablicy
- <F14>-skalowanie rysunku
- <F16> lub TCM-tryb kolorowy<>tryb jednobarwny.

Do komputera, który miałem wtedy do dyspozycji podpięty był ploter i drukarka, więc program potrafił rysować rysunki i drukować zestawy znalezionych punktów. Można też było „wydziurkować” układ punktów na specjalnej dziurkarce – taką taśmę „wciągała” obrabiarka sterowana numerycznie i była gotowa do wiercenia, frezowania, wycinania...

Wygenerowany tak rysunek można było zapisać na dysku w postaci pliku tekstowego (fragment poniżej):

FYJ22.RYS

P	0	0.0000000000E+00	0.0000000000E+00
P	1	4.9500000000E+02	0.0000000000E+00
P	2	-5.5000000000E+01	-5.9551298552E+01
P	3	-5.5000000000E+01	2.2781745931E+01
P	4	-3.2636360562E+01	-7.8566754695E+01

...

L	1	-6.7090909091E+00	-8.0721670567E+01
L	2	-5.7275649276E+01	-5.7275649276E+01
L	3	-5.5000000000E+01	0.0000000000E+00
L	4	-3.8890872965E+01	3.8890872965E+01
L	5	0.0000000000E+00	5.5000000000E+01

...

C	1	4.9500000000E+02	0.0000000000E+00	1.2200000000E+02
C	2	0.0000000000E+00	0.0000000000E+00	8.1000000000E+01
C	3	0.0000000000E+00	0.0000000000E+00	5.5000000000E+01
C	4	-3.1808043750E+01	-6.8601119178E+01	1.0000000000E+01
C	5	4.5000000000E+02	0.0000000000E+00	6.8000000000E+01
C	6	4.9500000000E+02	0.0000000000E+00	6.8000000000E+01


```

procedure sav(znak:char;numer:byte);forward;

procedure save_poj(znak:char;numer:byte);
begin append(plik);sav(znak,numer);close(plik) end;

procedure wskaznik; forward;
procedure przerysuj; forward;

function sprawdz(z:char;n:byte;bee:boolean):boolean; {sprawdzenie czy tablica
jest pusta}
begin
  case z of
    'P':if (tabp[n,1]=0) and (tabp[n,2]=0) and (n>0) then
      begin
        gt;write('BRAK PUNKTU');sprawdz:=true;
        if n in [0..220] then write(' NUMER ',n);
      end
    else sprawdz:=false;
    'L':if (tabl[n,1]=0) and (tabl[n,2]=0) and (n>0) then
      begin
        gt;write('BRAK PROSTEJ');sprawdz:=true;
        if n in [0..220] then write(' NUMER ',n);
      end
    else sprawdz:=false;
    'C':if (tabc[n,3]=0) and (n>0) then
      begin
        gt;write('BRAK OKREGU');sprawdz:=true;
        if n in [0..220] then write(' NUMER ',n);
      end
    else sprawdz:=false
  end
end;

function lancuch(dlug:byte):string; {odczytanie z klawiatury ciagu znakow}
var znak:char; x,y,poz:byte;
    bufor:array[1..80] of char;

    procedure wprawo;
    begin if poz<dlug then poz:=poz+1 end;
    procedure wlewo;
    begin if poz>1 then poz:=poz-1 end;

begin
  poz:=1;for x:=1 to dlug do bufor[x]:=#32;
  x:=wherex;y:=wherey;
  repeat
    while keypressed do znak:=readkey;
    gotoxy(x+poz,y);write('_');
    znak:=readkey;
    case znak of
      #27:lancuch:=#32;
#32..#255:begin
        znak:=upcase(znak);bufor[poz]:=znak;
        gotoxy(x+poz,y);write(znak);wprawo
      end;
      #13:lancuch:=copy(bufor,1,poz-1);
      #8:begin wlewo;bufor[poz]:=#32;gotoxy(x+poz+1,y);write(' ') end
    end;
  until (znak=#13) or (znak=#27)
end;

```

```

procedure pomoc;          {obsługa zbioru z informacjami pomocniczymi}
var poz,p1,p2,p3,p4,p5,p6,p7:byte;
    pos:integer;
    menu,menut,menup,menul,menuc,menuf,sciezka:string;
    zbior:string;
procedure napis1;
begin
    gotoxy(2,24);write('      ESC-koniec kursory GORA i DOL-'+
        'ustawienie podswietlenia RETURN-wybor      ');
end;
procedure napis2;
begin
    gotoxy(2,24);write('      ESC i RETURN-koniec kursory GORA i DOL-'+
        'PGUP i PGDN przesuwanie tekstu      ');
end;

begin textmode(lastmode);          {procedura glowna wywolania okienka pomocy}
    textattr:=48+15;
    clrscr;
    menut:=' Edytor@ Tablica@ Bufor@';
    menup:='  PXY@  PRQ@  PLC@  PLL@  PCC@  PSC@  PLI@  PCR@  TRA@  ROT@  SYM@
KONIEC@@';
    menul:=' LPP@  LPC@  LLD@  LPQ@  LCQ@  LCC@ KONIEC@@';
    menuc:=' CPR@  CPL@  CLL@  CPP@  CLC@  CCC@  CPC@ KONIEC@@';
    menuf:='  F1  THH@  F2  TSA@  F3  TLO@  F4  TDD@  F5  TWW@'+
        '  F6  TSS@  F7  TPP@  F8  TTT@  F9  TRR@  F10  T**@'+
        '  F11  TN-@  F12  TB-@  F13  TM-@  F16  TCM@ KONIEC@@';
    menu:=' Informacje ogolne@ lista Kodow *POG*@ Tryby pracy *POG*'+
        ' Punkty@ Linie@ Okregi@ Funkcje specjalne@@';
    pos:=0;poz:=1;
repeat
    zmaz(5,5,25,16,16+15,' ');
    ramka(5,5,25,16);
    textbackground(1);
    napis1;
    gotoxy(6,6);write(' INFORMACJE OGOLNE ');
    gotoxy(6,7);write(' LISTA KODOW *POG* ');
    gotoxy(6,8);write(' OBSLUGA PROGRAMU ');
    gotoxy(6,9);write(' INFORMACJA 1 ');
    gotoxy(6,10);write(' INFORMACJA 2 ');
    gotoxy(6,11);write(' PUNKTY ');
    gotoxy(6,12);write(' LINIE ');
    gotoxy(6,13);write(' OKREGI ');
    gotoxy(6,14);write(' FUNKCJE SPECJALNE ');
    gotoxy(6,15);write(' KONIEC ');
    obsluga_podswietlenia(poz,6,6,24,15,1+32,15+16);
    str(poz,zbior);
    sciezka:='\POG\POMOC\' +zbior+'.POG';
    napis2;
    case poz of
1..5:okno_zbior(pos,5,5,75,15,1,1,sciezka,1+32,15+16);
6:begin
    p4:=1;
    repeat
    napis1;
    okno_menu(p4,15,5,25,16,'',menup,14+64,15+16);
    if p4=12 then p4:=0;
    if p4>0 then
        begin
            str(p4+5,zbior);
            sciezka:='\POG\POMOC\' +zbior+'.POG';

```

```

        napis2;
        okno_zbior(pos,5,5,75,15,1,1,sciezka,1+64,15+16);
    end;
    until p4=0
end;
7:begin
    p5:=1;
    repeat
        napis1;
        okno_menu(p5,15,5,25,11,'',menul,14+64,15+16);
        if p5=7 then p5:=0;
        if p5>0 then
            begin
                str(p5+16,zbior);
                sciezka:='\POG\POMOC\'+zbior+'.POG';
                napis2;
                okno_zbior(pos,5,5,75,15,1,1,sciezka,1+64,15+16);
            end;
        until p5=0
    end;
8:begin
    p6:=1;
    repeat
        napis1;
        okno_menu(p6,15,5,25,12,'',menuc,14+64,15+16);
        if p6=8 then p6:=0;
        if p6>0 then
            begin
                str(p6+22,zbior);
                sciezka:='\POG\POMOC\'+zbior+'.POG';
                napis2;
                okno_zbior(pos,5,5,75,15,1,1,sciezka,1+64,15+16);
            end;
        until p6=0
    end;
9:begin
    p7:=1;
    repeat
        napis1;
        okno_menu(p7,15,5,30,19,'',menuf,14+64,15+16);
        if p7=15 then p7:=0;
        if p7>0 then
            begin
                str(p7+29,zbior);
                sciezka:='\POG\POMOC\'+zbior+'.POG';
                napis2;
                okno_zbior(pos,5,5,75,15,1,1,sciezka,1+64,15+16);
            end;
        until p7=0
    end;
10:poz:=0
end;

until poz=0;
if tryb=2 then
    begin
        hires;hirescolor(kolor);graphwindow(0,0,640,200);colortable(1,0,1,0)
    end
else begin graphcolormode;palette(3);colortable(1,0,0,0) end;
przerysuj
end;

```

```

procedure sortuj(znak:char;ile:byte);
var i,j,wsk:byte;
begin
  wsk:=229;
  case znak of
    'P':begin
      for i:=230 to 229+ile do
        if (tabp[i,1]<>0) or (tabp[i,2]<>0) then
          begin
            for j:=i+1 to 229+ile do
              if (int(tabp[j,1]*1000)=int(tabp[i,1]*1000)) and
                (int(tabp[j,2]*1000)=int(tabp[i,2]*1000)) then
                begin tabp[j,1]:=0;tabp[j,2]:=0 end;
              wsk:=wsk+1;
              tabp[wsk]:=tabp[i]
            end
          end;
    'L':begin
      for i:=230 to 229+ile do
        if (tabl[i,1]<>0) or (tabl[i,2]<>0) then
          begin
            for j:=i+1 to 229+ile do
              if (int(tabl[j,1]*1000)=int(tabl[i,1]*1000)) and
                (int(tabl[j,2]*1000)=int(tabl[i,2]*1000)) then
                begin tabl[j,1]:=0;tabl[j,2]:=0 end;
              wsk:=wsk+1;
              tabl[wsk]:=tabl[i]
            end
          end;
    'C':begin
      for i:=230 to 229+ile do
        if (tabc[i,1]<>0) or (tabc[i,2]<>0) or (tabc[i,3]<>0) then
          begin
            for j:=i+1 to 229+ile do
              if (int(tabc[j,1]*1000)=int(tabc[i,1]*1000)) and
                (int(tabc[j,2]*1000)=int(tabc[i,2]*1000)) and
                (int(tabc[j,3]*1000)=int(tabc[i,3]*1000)) then
                begin tabc[j,1]:=0;tabc[j,2]:=0;tabc[j,3]:=0 end;
              wsk:=wsk+1;
              tabc[wsk]:=tabc[i]
            end
          end;
    end;
  end;
  rozwiazania:=wsk-229
end;

procedure byteread(var p:byte); {bezpieczne wczytanie liczby byte}
var ok:boolean;x,y:byte;liczba:string;kod:integer;
begin
  x:=wherex-2;y:=wherey;
  repeat
    gotoxy(x,y);write(' ');gotoxy(x,y);
    liczba:=lancuch(4);val(liczba,p,kod);
    if kod<>0 then beep(2)
  until kod=0
end;

procedure realread(var p:real); {bezpieczne wczytanie liczby real}
var ok:boolean;x,y:byte;liczba:string;kod:integer;
begin

```



```

x:=wherex-2;y:=wherey;
repeat
  gotoxy(x,y);write(' ');gotoxy(x,y);
  liczba:=lancuch(9);val(liczba,p,kod);
  if kod<>0 then beep(2)
until kod=0
end;

procedure na_czesci;          {obliczenie sin i cos do podzialu okregu na odcinki}
var czesc:real;i:byte;
begin
  czesc:=2*pi/ile_cz;
  for i:=1 to ile_cz do
    begin wsp_okr[i,1]:=sin(i*czesc);wsp_okr[i,2]:=cos(i*czesc) end;
  wsp_okr[ile_cz,1]:=0;
  wsp_okr[ile_cz,2]:=1
end;

procedure zeruj(znak:char;odd,dod:byte);    {zerowanie tablic plc}
var i,j:byte;
begin
  for i:=odd to dod do
    case znak of
      'P':for j:=1 to 2 do tabp[i,j]:=0;
      'L':for j:=1 to 2 do tabl[i,j]:=0;
      'C':for j:=1 to 3 do tabc[i,j]:=0
    end
  end;
end;

procedure wsplinii(var wspx,wspy:real;nrl:byte); {obliczenie wspolczynnikow linii}
begin
  ggg(nrl);
  if (abs(u1)<liczba_min) or (abs(u1)=90)
    then begin kat1:=90;kat:=pi/2 end
    else
      begin
        kat:=arctan(u2/u1);kat1:=kat*180/pi;
        if kat1<0 then kat1:=180+kat1;
        kat:=sin(kat)/cos(kat);
        wspy:=kat*(wspx-u1-x1)+u2+y1
      end
  end;
end;

procedure pixle(wspx,wspy,r:real); {konwersja wspolrzedne rysunku - wspolrzedne
ekranu}
var lx,ly,lr:string;kod:integer;
begin
  str((wspx-xsr1)*skalax:10:0,lx);val(lx,pixx1,kod);pixx1:=przesx+pixx1;
  str((wspy-ysr1)*skalay:10:0,ly);val(ly,pixy1,kod);pixy1:=przesy-pixy1;
  str(skala*r:10:0,lr);val(lr,prom,kod)
end;

procedure rysuj_punkt(nrp:byte;jak:rys);    {rysowanie punktu na ekranie}
var x,y:integer;
begin
  pixle(tabp[nrp,1],tabp[nrp,2],0);
  circle(pixx1,pixy1,3,jak);
  circle(pixx1,pixy1,2,jak);
  if jak<>0 then circle(pixx1,pixy1,2,jak);
  plot(pixx1,pixy1,jak);
  if (numeracja_p=true) and (nrp<230) then

```

```

begin
  x:=(pixx1 div 8)+2;y:=(pixy1 div 8)+1;
  if (x>0) and (x<77) and (y>0) and (y<25) then
    begin gotoxy(x,y);write(nrp) end
  end
end;

procedure rysuj_linie1(nrl:byte;jak:rys);      {rysowanie linii na ekranie}
var i,l:byte; j,k:real;kod,x,y:integer;jx:string;
begin
  wspx:=0;wspy:=0;wsplinii(wspx,wspy,nrl);
  if abs(u1)>liczba_min then
    begin
      if (abs(kat1)>=85) and (abs(kat1)<=95) then l:=10
      else if (abs(kat1)>=75) and (abs(kat1)<=105) then l:=15
      else if (abs(kat1)>=60) and (abs(kat1)<=120) then l:=20
      else if (abs(kat1)>=45) and (abs(kat1)<=135) then l:=25
      else if (abs(kat1)>=0) and (abs(kat1)<=180) then l:=30;
      kod:=-10;pixx2:=-10;pixy2:=200;i:=0;
      if l>5 then
        begin
          repeat
            j:=(kod-przesx)/skalax+xsr1;k:=kat*(j-u1-x1)+u2+y1;
            pixle(j,k,0);draw(pixx1,pixy1,pixx2,pixy2,jak);
            pixx2:=pixx1;pixy2:=pixy1;kod:=kod+1;
            if (i=0) and (((pixy2 div 8)+1)>1) and (((pixy2 div 8)+1)<24) then
              begin
                if (numeracja_l=true) and (nrl<230) then
                  begin
                    x:=(pixx2 div 8)+2;y:=(pixy2 div 8)+1;i:=1;
                    if (x>=0) and (x<77) and (y>1) and (y<25) then
                      begin gotoxy(x,y);write(nrl) end
                    end
                  end
                end
            until kod>640
          end
        else
          begin
            l:=8-l;pixx2:=-10;pixy2:=200;
            repeat
              j:=(kod-przesy)/skalay+ysr1;k:=(j-u2-y1)/kat+u1+x1;
              pixle(k,j,0);draw(pixx2,pixy2,pixx1,pixy1,jak);
              pixx2:=pixx1;pixy2:=pixy1;kod:=kod+1;
              if (i=0) and (((pixy2 div 8)+1)>1) and (((pixy2 div 8)+1)<24) then
                begin
                  if (numeracja_l=true) and (nrl<230) then
                    begin
                      x:=(pixx2 div 8)+2;y:=(pixy2 div 8)+1;i:=1;
                      if (x>=0) and (x<77) and (y>1) and (y<25) then
                        begin gotoxy(x,y);write(nrl) end
                      end
                    end
                  end
            until kod>200
          end
        end
      end
    else
      if (u1=0) and (u2=0) and (x1=0) and (y1=0) then exit
    else
      begin
        pixle(x1,0,0);pixx2:=pixx1;
        for i:=0 to 40 do

```

```

begin
  j:=i*5;j:=(przesy-j)/skalay+ysr1;pixle(x1,j,0);
  draw(pixx2,pixy2,pixx1,pixy1,jak);pixy2:=pixy1
end;
if (numeracja_l=true) and (nrl<230) then
begin
  x:=(pixx2 div 8)+1;y:=24;
  if (x>0) and (x<77) and (y>0) and (y<25) then
begin gotoxy(x,y);write(nrl) end
end
end;
gt
end;

procedure rysuj_okrag1(nro:byte;jak:rys); {rysowanie okregu na ekranie}
var x,y,r,r1,xsr,ysr:real;i,j:byte;x1,y1:integer;
  procedure okrag_xy(num:byte);
begin
  x:=xsr+r*wsp_okr[num,2];y:=ysr+r*wsp_okr[num,1];pixle(x,y,r)
end;
begin
  xsr:=tabc[nro,1];ysr:=tabc[nro,2];r:=tabc[nro,3];
  okrag_xy(ile_cz);r1:=r*skala;
  if r1<5 then j:=8;
  if (r1>=5) and (r1<30) then j:=6;
  if (r1>=30) and (r1<40) then j:=4;
  if (r1>=40) and (r1<60) then j:=3;
  if (r1>=60) and (r1<120) then j:=2;
  if r1>=120 then j:=1;
  for i:=1 to ile_cz do
begin
  if (i mod j)=0 then
begin
  pixx2:=pixx1;pixy2:=pixy1;okrag_xy(i);
  draw(pixx2,pixy2,pixx1,pixy1,jak)
end
end;
if (numeracja_c=true) and (nro<230) then
begin
  x1:=(pixx2 div 8)+2;y1:=(pixy2 div 8)+1;
  if (x1>0) and (x1<77) and (y1>0) and (y1<25) then
begin gotoxy(x1,y1);write(nro) end
end
end;

procedure na_ekran(znak2:char;num:byte;jak:rys); forward;
var kon1:boolean;

procedure wydruk(ile,poz:byte); {obsługa obliczonych wspolrzednych na
poczatku}
var koniec:boolean;
begin
  if ile=0 then
begin gt;write('BRAK ROZWIAZAN');beep(5); exit end;
  if ile=1 then
case znak of
  'P':begin tabp[p]:=tabp[poz];na_ekran('P',p,kolor_p) end;
  'L':begin tabl[p]:=tabl[poz];na_ekran('L',p,kolor_l) end;
  'C':begin tabc[p]:=tabc[poz];na_ekran('C',p,kolor_c) end
end
end
else

```

```

begin
  poz:=230;
  case znak of
    'P':na_ekran('P',poz,-1);
    'L':na_ekran('L',poz,-1);
    'C':na_ekran('C',poz,-1)
  end;
  repeat
    if tryb=1 then
      begin gotoxy(17,1);write('BUFOR          koniec-ESC') end
    else
      begin gotoxy(58,1);write('BUFOR          koniec-ESC') end;
      znak1:=readkey;
      case znak1 of
        #27:przerysuj;
        #13:case znak of
          'P':begin tabp[p]:=tabp[poz];na_ekran('P',p,kolor_p) end;
          'L':begin tabl[p]:=tabl[poz];na_ekran('L',p,kolor_l) end;
          'C':begin tabc[p]:=tabc[poz];na_ekran('C',p,kolor_c) end
        end;
        #0:begin
          znak1:=readkey;
          case znak of
            'P':na_ekran('P',poz,-1);
            'L':na_ekran('L',poz,-1);
            'C':na_ekran('C',poz,-1)
          end;
          case znak1 of
            #77:if poz<229+ile then poz:=poz+1 else poz:=230;
            #75:if poz<231 then poz:=229+ile else poz:=poz-1
            else beep(3)
          end;
          case znak of
            'P':na_ekran('P',poz,-1);
            'L':na_ekran('L',poz,-1);
            'C':na_ekran('C',poz,-1)
          end
        end
      end
    else beep(3)
  end;
  until (znak1=#27) or (znak1=#13);
end
end;

procedure przerysuj;          {przerysowanie ekranu z rysunkiem}
VAR i:byte;
begin
  clearscreen;
  i:=0;
  while i<=max_nr_p do
    begin if sprawdz('P',i,false)=false then rysuj_punkt(i,kolor_p);i:=i+1 end;
  i:=0;
  while i<=max_nr_l do
    begin if sprawdz('L',i,false)=false then rysuj_linie1(i,kolor_l);i:=i+1 end;
  i:=0;
  while i<=max_nr_c do
    begin if sprawdz('C',i,false)=false then rysuj_okrag1(i,kolor_c);i:=i+1 end;
  wsk_x1:=-10;wsk_y1:=-190;wskaznik;beep(2)
end;

```

□

```

unit procpog;
interface

var r0,r1,r2,r6,r7,a,a2,a3,a6,a7,a8,a9,b,b0,b1,b2,b6,b7,b8,b9,s1,s2,v1,v2,v9,
    w,w1,n,no,n1,x,y,x1,x2,x6,x7,y1,y2,z,u1,u2,u3,u4,d,d3:real;
    ktory,e1,m,m1,m2,m3,p,p0,j,j1,j2,j3,j4,j5,j6,j7,j8:byte;
    blad:boolean;
    tabp:array[0..255,1..2] of real;    {tablica punktow}
    tabl:array[0..255,1..2] of real;    {tablica linii }
    tabc:array[0..255,1..3] of real;    {tablica okregow}
const liczba_max=9.0e30;
      liczba_min=1E-5;

```

{PROCEDURY OBLICZAJACE WSPOLRZEDNE PLC}

```

procedure qqq(q:byte);
procedure ggg(j:byte);
procedure hhh(j:byte);
procedure bbb(j:byte);
procedure fff(j7:byte);
procedure eee(j6:byte);
procedure aaa(j8:byte);
procedure ccc(j4:byte);
procedure ddd(j5:byte);
procedure iii(j:byte);
procedure jjj(var m:byte);
procedure zzz(i,e1:byte);

```

implementation

```

procedure qqq(q:byte);
begin blad:=true end;

```

```

procedure ggg(j:byte);
begin
  if tabl[j,1]>=(liczba_max/10)
    then begin x1:=0;y1:=0;
           z:=tabl[j,2];
           if z=90 then begin u1:=0;u2:=1; exit end;
           u1:=cos(z);
           u2:=sin(z)
         end
    else begin x1:=tabl[j,1];
           y1:=tabl[j,2];
           u1:=-y1;
           u2:=x1
         end
end;

```

```

procedure hhh(j:byte);
begin x:=(x1*u2*u2-u1*u2*y1)/(u2*u2+u1*u1);
      y:=(y1*u1*u1-u1*u2*x1)/(u2*u2+u1*u1);
      if sqrt(x*x+y*y)>=d
        then begin tabl[j,1]:=x;tabl[j,2]:=y end
        else begin x:=liczba_max;
                 if abs(u1)<d
                   then begin y:=90;tabl[j,1]:=x;tabl[j,2]:=y end
                   else begin y:=arctan(u2/u1);tabl[j,1]:=x;tabl[j,2]:=y end
               end
end

```

```

end;

procedure bbb(j:byte);
begin ggg(j1);
      x2:=x1;y2:=y1;u3:=u1;u4:=u2;
      ggg(j2);
      w:=-u1*u4+u2*u3;
      w1:=- (x2-x1)*u4+u3*(y2-y1);
      if abs(w)>liczba_min
        then begin tabp[j,1]:=u1*(w1/w)+x1;
                tabp[j,2]:=u2*(w1/w)+y1
              end
        else qqq(6)
      end;
end;

procedure fff(j7:byte);
begin if sqrt(sqr(x1-x2)+sqr(y1-y2))<d
      then begin tabp[j7,1]:=x1;tabp[j7,2]:=y1;exit end;
      if (m<>1) and (m<>2)
        then if abs(y1-y2)>0
              then if (((y1-y2)<0) and (m=3)) or (((y1-y2)>0) and (m=4))
                    then begin tabp[j7,1]:=x1;tabp[j7,2]:=y1;exit end
                        else begin tabp[j7,1]:=x2;tabp[j7,2]:=y2;exit end
                    else begin qqq(2); exit end;
              if abs(x1-x2)>d
                then if (((x1-x2)<0) and (m=1)) or (((x1-x2)>0) and (m=2))
                      then begin tabp[j7,1]:=x1;tabp[j7,2]:=y1;exit end
                          else begin tabp[j7,1]:=x2;tabp[j7,2]:=y2;exit end
                    else qqq(3)
                end;
            end;
end;

procedure eee(j6:byte);
begin if (tabp[j1,1]=0) and (tabp[j2,1]=0)
      and (tabp[j1,2]=0) and (tabp[j2,2]=0) then qqq(3)
      else
        if (abs(tabp[j1,1]-tabp[j2,1])>d) or (abs(tabp[j1,2]-tabp[j2,2])>d)
          then begin x1:=tabp[j1,1];y1:=tabp[j1,2];
                    u1:=tabp[j2,1]-tabp[j1,1];
                    u2:=tabp[j2,2]-tabp[j1,2];
                    hhh(j6)
                  end
          else qqq(3)
        end;
end;

procedure aaa(j8:byte);
begin if v9<>1 then ggg(j1);
      s1:=x1;s2:=y1;v9:=0;
      x1:=(-u2)*a/(sqrt(u1*u1+u2*u2));
      y1:=u1*a/(sqrt(u1*u1+u2*u2));
      x2:=-x1;y2:=-y1;
      fff(254);
      if blad=true then exit;
      x1:=s1+tabp[254,1];
      y1:=s2+tabp[254,2];
      hhh(j8)
end;

procedure ccc(j4:byte);
begin if v9<>1 then begin ggg(j1);
                        a9:=tabc[j2,1];b9:=tabc[j2,2];r0:=tabc[j2,3]
                      end;
end;

```

```

a8:=x1;b8:=y1;
b:=u1*(x1-a9)+u2*(y1-b9);
b0:=b*b-(u1*u1+u2*u2)*(sqr(x1-a9)+sqr(y1-b9)-r0*r0);
if (b0<0) and ((abs(b0/(u1*u1+u2*u2)))>d)
  then begin qqq(3); exit end;
b1:=(-b-sqrt(abs(b0)))/(u1*u1+u2*u2);
b2:=(-b+sqrt(abs(b0)))/(u1*u1+u2*u2);
x1:=u1*b1+a8;y1:=u2*b1+b8;
x2:=u1*b2+a8;y2:=u2*b2+b8;
fff(j4);
v9:=0
end;

procedure ddd(j5:byte);
begin a7:=tabc[j1,1];b7:=tabc[j1,2];r7:=tabc[j1,3];
a6:=tabc[j2,1];b6:=tabc[j2,2];r6:=tabc[j2,3];
d3:=sqrt(sqr(a7-a6)+sqr(b7-b6));
if d3>d
  then begin d3:=(-(r7*r7)+r6*r6+a7*a7-(a6*a6)+b7*b7-(b6*b6))/(2*d3*d3);
u1:=(b7-b6);u2:=a7-a6;
x1:=(a7-a6-b7+b6)*d3;
y1:=(a7-a6+b7-b6)*d3;
a9:=a7;b9:=b7;r0:=r7;
v9:=1;
ccc(j5)
end
else qqq(3)
end;

procedure iii(j:byte);
begin tabc[j,1]:=tabc[j1,1];
tabc[j,2]:=tabc[j1,2];
tabc[j,3]:=tabc[j1,3]+a*(-ord(m=5)+ord(m=6))
end;

procedure jjj(var m:byte);
begin x1:=tabp[251,1];y1:=tabp[251,2];
x2:=tabp[252,1];y2:=tabp[252,2];
if abs(y1-y2)<=d
  then if x1<=(x2-d)
    then m:=4*ord(m=1)+3*ord(m=2)
    else m:=3*ord(m=1)+4*ord(m=2)
  else if y1<=(y2-d)
    then exit
    else m:=ord(m=2)+2*ord(m=1)
end;

procedure zzz(i,e1:byte);
begin
  tabc[e1,1]:=tabp[i,1];
  tabc[e1,2]:=tabp[i,2]
end;

{PROCEDURY OBLICZAJACE WSPOLRZEDNE DLA DANEGO KODU}

procedure PXY;
begin
  gt;write(ng,' ODN = 0');
  znak3:=readkey;
  case znak3 of

```

```

    #13:p0:=0;
    else bytread(p0)
end;
if sprawdz('P',p0,true)=true then begin beep(3);exit end;
gt;write('X = ');realread(x);
if x>10000 then begin beep(3);exit end;
gt;write('Y = ');realread(y);
if y>10000 then begin beep(3);exit end;
tabp[p,1]:=tabp[p0,1]+x;
tabp[p,2]:=tabp[p0,2]+y;
gt;wydruk(1,p);save_poj('P',p)
end;

procedure PRQ;
begin
    gt;write(ng,' ODN = 0');bytread(p0);
    if sprawdz('P',p0,true)=true then begin beep(3);exit end;
    gt;write(nj,' = ');realread(x);
    if x>10000 then begin beep(3);exit end;
    gt;write(nk,' = ');realread(y);
    if y>10000 then begin beep(3);exit end;
    y:=(y*pi)/180;
    tabp[p,1]:=tabp[p0,1]+x*cos(y);
    tabp[p,2]:=tabp[p0,2]+x*sin(y);
    gt;wydruk(1,p);save_poj('P',p)
end;

procedure PLCM;
begin
    gt;write(nh,' = ');bytread(j1);
    if sprawdz('L',j1,true)=true then begin beep(3);exit end;
    gt;write(ni,' = ');bytread(j2);
    if sprawdz('C',j2,true)=true then begin beep(3);exit end;
    zeruj('P',230,233);
    blad:=false;m1:=1;m:=1;ccc(230);blad:=false;m1:=2;m:=2;ccc(231);
    blad:=false;m1:=3;m:=3;ccc(232);blad:=false;m1:=4;m:=4;ccc(233);
    sortuj('P',4);wydruk(rozwiazania,230);save_poj('P',p)
end;

procedure PLL;
begin
    gt;write(nh,' PIERWSZEJ = ');bytread(j1);
    if sprawdz('L',j1,true)=true then begin beep(3);exit end;
    gt;write(nh,' DRUGIEJ = ');bytread(j2);
    if sprawdz('L',j2,true)=true then begin beep(3);exit end;
    bbb(p);
    if blad=true then begin blad:=false;exit end
    else begin wydruk(1,p);save_poj('P',p) end
end;

procedure PCCM;
begin
    gt;write(ni,' PIERWSZEGO = ');bytread(j1);
    if sprawdz('C',j1,true)=true then begin beep(3);exit end;
    gt;write(ni,' DRUGIEGO = ');bytread(j2);
    if sprawdz('C',j2,true)=true then begin beep(3);exit end;
    zeruj('P',230,233);
    blad:=false;m1:=1;m:=1;ddd(230);blad:=false;m1:=2;m:=2;ddd(231);
    blad:=false;m1:=3;m:=3;ddd(232);blad:=false;m1:=4;m:=4;ddd(233);
    sortuj('P',4);wydruk(rozwiazania,230);save_poj('P',p)
end;

```



```

procedure PSC;
begin
  gt;write(ni, ' = ');byteread(j1);
  if sprawdz('C',j1,true)=true then begin beep(3);exit end;
  tabp[p,1]:=tabc[j1,1];tabp[p,2]:=tabc[j1,2];
  wydruk(1,p);save_poj('P',p)
end;

procedure LPP;
begin
  gt;write(ng, '   PIERWSZEGO = ');byteread(j1);
  if sprawdz('P',j1,true)=true then begin beep(3);exit end;
  gt;write(ng, '   DRUGIEGO = ');byteread(j2);
  if sprawdz('P',j2,true)=true then begin beep(3);exit end;
  eee(p);
  if blad=true then begin blad:=false;exit end
  else begin wydruk(1,p);save_poj('P',p) end
end;

procedure lpcmp(p1:byte);
begin
  tabc[251,1]:=(tabp[j1,1]+tabc[j2,1])/2;
  tabc[251,2]:=(tabp[j1,2]+tabc[j2,2])/2;
  tabc[251,3]:=sqrt(sqr(tabp[j1,1]-tabc[j2,1])
    +sqr(tabp[j1,2]-tabc[j2,2]))/2;
  tabp[251,1]:=tabp[j1,1];
  tabp[251,2]:=tabp[j1,2];
  tabp[252,1]:=tabc[j2,1];
  tabp[252,2]:=tabc[j2,2];
  blad:=false;
  jjj(m);j3:=j1;j1:=251;ddd(251);
  if blad=true then begin blad:=false; exit end;
  j1:=j3;j2:=251;eee(p1);
end;

procedure LPCM;
var jp:byte;
begin
  gt;write(ng, ' = ');byteread(j1);
  if sprawdz('P',j1,true)=true then begin beep(3);exit end;
  gt;write(ni, ' = ');byteread(j2);
  if sprawdz('C',j2,true)=true then begin beep(3);exit end;
  zeruj('L',230,231);
  jp:=j2;m1:=1;m:=1;lpcmp(230);j2:=jp;m1:=2;m:=2;lpcmp(231);
  sortuj('L',2);wydruk(rozwiazania,230);save_poj('L',p)
end;

procedure LLDM;
begin
  gt;write(nh, '   ODN = ');byteread(j1);
  if sprawdz('L',j1,true)=true then begin beep(3);exit end;
  gt;write('ODLEGLOSC = ');realread(a);
  if a>10000 then begin beep(3);exit end;
  zeruj('L',230,233);
  blad:=false;m1:=1;m:=1;aaa(230);blad:=false;m1:=2;m:=2;aaa(231);
  blad:=false;m1:=3;m:=3;aaa(232);blad:=false;m1:=4;m:=4;aaa(233);
  sortuj('L',4);wydruk(rozwiazania,230);save_poj('L',p)
end;

procedure LPQ;

```

```

begin
  gt;write(ng, ' = ');byteread(j2);
  if sprawdz('P',j2,true)=true then begin beep(3);exit end;
  gt;write(nh, ' ODN = ');byteread(j1);
  if sprawdz('L',j1,true)=true then begin beep(3);exit end;
  gt;write(nk, ' = ');realread(x6);
  if x6>10000 then begin beep(3);exit end;
  x6:=(x6*pi)/180;
  ggg(j1);
  v1:=u1*cos(x6)-u2*sin(x6);
  v2:=u1*sin(x6)+u2*cos(x6);
  x1:=tabp[j2,1];y1:=tabp[j2,2];u1:=v1;u2:=v2;
  hhh(p);
  wydruk(1,p);save_poj('L',p)
end;

procedure LCQM;
begin
  gt;write(ni, ' = ');byteread(j1);
  if sprawdz('C',j1,true)=true then begin beep(3);exit end;
  gt;write(nk, ' = ');realread(x7);
  if x7>10000 then begin beep(3);exit end;
  zeruj('L',230,233);
  blad:=false;x1:=tabc[j1,1];y1:=tabc[j1,2];a:=tabc[j1,3];
  x7:=(x7*pi)/180;
  u1:=cos(x7);u2:=sin(x7);
  v9:=1;m1:=1;m:=1;aaa(230);
  blad:=false;x1:=tabc[j1,1];y1:=tabc[j1,2];v9:=1;m1:=2;m:=2;aaa(231);
  blad:=false;x1:=tabc[j1,1];y1:=tabc[j1,2];v9:=1;m1:=3;m:=3;aaa(232);
  blad:=false;x1:=tabc[j1,1];y1:=tabc[j1,2];v9:=1;m1:=4;m:=4;aaa(233);
  sortuj('L',4);wydruk(rozwiazania,230);save_poj('L',p)
end;

procedure LCCMP(p1,m1,m2:byte);
var b0:byte;
begin
  if tabc[j3,3]<tabc[j4,3] then
    begin
      b0:=j3;j3:=j4;j4:=b0;b0:=m1;
      m1:=m2*ord(m1<>m2)+(ord(m1=2)+2*ord(m1=1))*ord(m1=m2);
      m2:=b0*ord(b0<>m2)+m1*ord(b0=m2)
    end;
  u3:=tabc[j3,1]-tabc[j4,1];
  u4:=tabc[j3,2]-tabc[j4,2];
  a2:=sqrt(u3*u3+u4*u4);
  if a2<abs(tabc[j3,3]-tabc[j4,3]) then exit; {blad3}
  if (m1<>m2) and (a2<(tabc[j3,3]+tabc[j4,3])) then exit; {blad3}
  u3:=u3/a2;u4:=u4/a2;
  a7:=abs(tabc[j3,3]+(ord(m1<>m2)-ord(m1=m2))*tabc[j4,3]);
  u1:=- (u4*a7/a2+(ord(m1=1)-ord(m1=2))*u3*sqrt(a2*a2-a7*a7)/a2);
  u2:=u3*a7/a2+(ord(m1=2)-ord(m1=1))*u4*sqrt(a2*a2-a7*a7)/a2;
  x1:=tabc[j3,1]-tabc[j3,3]*u2;
  y1:=tabc[j3,2]+tabc[j3,3]*u1;
  hhh(p1)
end;

procedure LCCM;
begin
  gt;write(ni, ' PIERWSZEGO = ');byteread(j3);
  if sprawdz('C',j3,true)=true then begin beep(3);exit end;
  gt;write(ni, ' DRUGIEGO = ');byteread(j4);

```

```

    if sprawdz('C',j4,true)=true then begin beep(3);exit end;
    x2:=j3;y2:=j4;
    zeruj('L',230,233);
    lccmp(230,1,1);lccmp(231,1,2);lccmp(232,2,1);lccmp(233,2,2);
    sortuj('L',4);wydruk(rozwiazania,230);save_poj('L',p)
end;

procedure CPR;
var j5:real;
begin
    gt;write(ng,' SRODKOWEGO = ');byteread(j4);
    if sprawdz('P',j4,true)=true then begin beep(3);exit end;
    gt;write(nj,' = ');realread(j5);
    if j5>10000 then begin beep(3);exit end;
    zzz(j4,p);
    tabc[p,3]:=j5;
    wydruk(1,p);save_poj('C',p)
end;

procedure CPLMP(p1:byte);
begin
    ggg(j4);
    r1:=- (tabp[j3,1]-x1)*u2+(tabp[j3,2]-y1)*u1;
    if sqrt(sqrt(u2*r1)+sqrt(u1*r1))>=d then
        if abs(u2*r1)>=d
            then m:=2*ord((-u2*r1)>d)+ord((-u2*r1)<d)
            else m:=4*ord((u1*r1)>d)+3*ord((u1*r1)<d);
    j1:=j4;a:=r2;
    aaa(253);
    if blad=true then begin blad:=false;exit end;
    zzz(j3,251);
    tabc[251,3]:=r2;m:=m1;j1:=253;j2:=251;
    ccc(251);
    if blad=true then begin blad:=false;exit end;
    zzz(251,p1);
    tabc[p1,3]:=r2
end;

procedure CPLM;
begin
    gt;write(ng,' = ');byteread(j3);
    if sprawdz('P',j3,true)=true then begin beep(3);exit end;
    gt;write(nh,' = ');byteread(j4);
    if sprawdz('L',j4,true)=true then begin beep(3);exit end;
    gt;write(nj,' = ');realread(r2);
    if r2>10000 then begin beep(3);exit end;
    zeruj('C',230,233);
    blad:=false;m1:=1;m:=1;cplmp(230);blad:=false;m1:=2;m:=2;cplmp(231);
    blad:=false;m1:=3;m:=3;cplmp(232);blad:=false;m1:=4;m:=4;cplmp(233);
    sortuj('C',4);wydruk(rozwiazania,230);save_poj('C',p)
end;

procedure CLLMP(p1,m1,m2:byte);
begin
    m:=m1;a:=r2;j1:=j3;aaa(253);
    if blad=true then begin blad:=false;exit end;
    m:=m2;j1:=j4;aaa(254);
    if blad=true then begin blad:=false;exit end;
    j1:=253;j2:=254;bbb(254);
    if blad=true then begin blad:=false;exit end;
    zzz(254,p1);

```

```
tabc[p1,3]:=r2
end;
```

```
procedure CLLM;
```

```
begin
```

```
gt;write(nh,' PIERWSZEJ = ');byteread(j3);
if sprawdz('L',j3,true)=true then begin beep(3);exit end;
gt;write(nh,' DRUGIEJ = ');byteread(j4);
if sprawdz('L',j4,true)=true then begin beep(3);exit end;
gt;write(nj,' = ');realread(r2);
if r2>10000 then begin beep(3);exit end;
zeruj('C',230,237);
blad:=false;cllmp(230,1,3);blad:=false;cllmp(231,1,4);
blad:=false;cllmp(232,2,3);blad:=false;cllmp(233,2,4);
blad:=false;cllmp(234,3,1);blad:=false;cllmp(235,4,1);
blad:=false;cllmp(236,3,2);blad:=false;cllmp(237,4,2);
sortuj('C',8);wydruk(rozwiazania,230);save_poj('C',p)
```

```
end;
```

```
procedure CPPMP(p1:byte);
```

```
begin
```

```
zzz(j3,251);tabc[251,3]:=r2;
zzz(j4,252);tabc[252,3]:=r2;j1:=251;j2:=252;
ddd(251);
if blad=true then begin blad:=false;exit end;
zzz(251,p1);tabc[p1,3]:=r2
```

```
end;
```

```
procedure CPPM;
```

```
begin
```

```
gt;write(ng,' PIERWSZEGO = ');byteread(j3);
if sprawdz('P',j3,true)=true then begin beep(3);exit end;
gt;write(ng,' DRUGIEGO = ');byteread(j4);
if sprawdz('P',j4,true)=true then begin beep(3);exit end;
gt;write(nj,' = ');realread(r2);
if r2>10000 then begin beep(3);exit end;
zeruj('C',230,233);
blad:=false;m1:=1;m:=1;cppmp(230);blad:=false;m1:=2;m:=2;cppmp(231);
blad:=false;m1:=3;m:=3;cppmp(232);blad:=false;m1:=4;m:=4;cppmp(233);
sortuj('C',4);wydruk(rozwiazania,230);save_poj('C',p)
```

```
end;
```

```
procedure CLCMP(p1,m1,m2,m3:byte);
```

```
begin
```

```
j1:=j3;a:=r2;m:=m1;
aaa(253);
if blad=true then begin blad:=false;exit end;
m:=m2;j1:=j4;
iii(251);
j1:=253;j2:=251;m:=m3;
ccc(251);
if blad=true then begin exit end;
zzz(251,p1);
tabc[p1,3]:=r2
```

```
end;
```

```
procedure CLCM;
```

```
begin
```

```
gt;write(nh,' = ');byteread(j3);
if sprawdz('L',j3,true)=true then begin beep(3);exit end;
gt;write(ni,' = ');byteread(j4);
```

```

if sprawdz('C',j4,true)=true then begin beep(3);exit end;
gt;write(nj,' = ');realread(r2);
if r2>10000 then begin beep(3);exit end;
zeruj('C',230,255);
blad:=false;clcmp(230,3,5,1);blad:=false;clcmp(231,3,5,2);
blad:=false;clcmp(232,3,6,1);blad:=false;clcmp(233,3,6,2);
blad:=false;clcmp(234,4,5,1);blad:=false;clcmp(235,4,5,2);
blad:=false;clcmp(236,4,6,1);blad:=false;clcmp(237,4,6,2);
blad:=false;clcmp(238,1,5,3);blad:=false;clcmp(239,1,5,4);
blad:=false;clcmp(240,1,6,3);blad:=false;clcmp(241,1,6,4);
blad:=false;clcmp(242,2,5,3);blad:=false;clcmp(243,2,5,4);
blad:=false;clcmp(244,2,6,3);blad:=false;clcmp(245,2,6,4);
sortuj('C',16);wydruk(rozwiazania,230);save_poj('C',p)
end;

```

```

procedure CCCMP(p1,m1,m2,m3:byte);
begin
  m:=m1;j1:=j3;
  iii(251);
  m:=m2;j1:=j4;
  iii(252);
  j1:=251;j2:=252;m:=m3;
  ddd(251);
  if blad=true then begin blad:=false;exit end;
  zzz(251,p1);
  tabc[p1,3]:=a
end;

```

```

procedure CCCM;
begin
  gt;write(ni,' PIERWSZEGO = ');byteread(j3);
  if sprawdz('C',j3,true)=true then begin beep(3);exit end;
  gt;write(ni,' DRUGIEGO = ');byteread(j4);
  if sprawdz('C',j4,true)=true then begin beep(3);exit end;
  gt;write(nj,' = ');realread(a);
  if a>10000 then begin beep(3);exit end;
  zeruj('C',230,255);
  blad:=false;cccmp(230,5,5,1);blad:=false;cccmp(231,5,5,2);
  blad:=false;cccmp(232,5,6,1);blad:=false;cccmp(233,5,6,2);
  blad:=false;cccmp(234,6,5,1);blad:=false;cccmp(235,6,5,2);
  blad:=false;cccmp(236,6,6,1);blad:=false;cccmp(237,6,6,2);
  blad:=false;cccmp(238,5,5,3);blad:=false;cccmp(239,5,5,4);
  blad:=false;cccmp(240,5,6,3);blad:=false;cccmp(241,5,6,4);
  blad:=false;cccmp(242,6,5,3);blad:=false;cccmp(243,6,5,4);
  blad:=false;cccmp(244,6,6,3);blad:=false;cccmp(245,6,6,4);
  sortuj('C',16);wydruk(rozwiazania,230);save_poj('C',p)
end;

```

```

procedure CPCMP(p1,m1,m2:byte);
begin
  a:=r2;
  if m1<>6 then a:=-r2+2*tabc[j4,3];
  m:=m1;j1:=j4;
  iii(251);
  zzz(j3,252);
  tabc[252,3]:=r2;
  j1:=251;j2:=252;m:=m2;
  ddd(251);
  if blad=true then begin blad:=false;exit end;
  zzz(251,p1);
  tabc[p1,3]:=r2
end;

```

```

end;

procedure CPCM;
begin
  gt;write(ng, ' = ');byteread(j3);
  if sprawdz('P',j3,true)=true then begin beep(3);exit end;
  gt;write(ni, ' = ');byteread(j4);
  if sprawdz('C',j4,true)=true then begin beep(3);exit end;
  gt;write(nj, ' = ');realread(r2);
  if r2>10000 then begin beep(3);exit end;
  zeruj('C',230,237);
  blad:=false;cpcmp(230,5,1);blad:=false;cpcmp(231,5,2);
  blad:=false;cpcmp(232,6,1);blad:=false;cpcmp(233,6,2);
  blad:=false;cpcmp(234,5,3);blad:=false;cpcmp(235,6,3);
  blad:=false;cpcmp(236,5,4);blad:=false;cpcmp(237,6,4);
  sortuj('C',8);wydruk(rozwiazania,230);save_poj('C',p)
end;

procedure pli;
var j,p0,p,j3,j4:byte;b7:real;
begin
  kolp:=kolp+1;
  gt;write(ng, ' POCZ. = ',kolp);
  znak3:=readkey;
  case znak3 of
    #13:p0:=kolp;
    else byteread(kolp)
  end;
  p0:=kolp;
  gt;write(ng, ' ODN. = ');byteread(p);
  if sprawdz('P',p,true)=true then begin beep(3);exit end;
  gt;write(nh, ' = ');byteread(j3);
  if sprawdz('L',j3,true)=true then begin beep(3);exit end;
  gt;write('PRZYROST = ');realread(b7);
  if b7>10000 then begin beep(3);exit end;
  gt;write('ILOSC PKT. = ');byteread(j4);
  if j4>255 then begin beep(3);exit end;
  ggg(j3);
  if (abs(u1)<liczba_min)
    then begin u1:=u1*u1; u2:=u2*u2 end
    else
      if u2<>0 then begin u1:=u1*u2; u2:=u2*u2 end
      else begin u1:=u1*u1; u2:=u2*u2 end;
  x2:=u1/(sqrt(u1*u1+u2*u2));
  y2:=u2/(sqrt(u1*u1+u2*u2));
  for j:=p0 to (j4+p0-1) do
    begin
      tabp[j,1]:=tabp[p,1]+(j-p0+1)*b7*x2;
      tabp[j,2]:=tabp[p,2]+(j-p0+1)*b7*y2;
      na_ekran('P',j,kolor_p);save_poj('P',j)
    end;
  if (p0+j4-1) > max_nr_p then max_nr_p:=p0+j4-1;
  kolp:=max_nr_p
end;

procedure pcr;
var j:byte;j4:real;
begin
  kolp:=kolp+1;
  gt;write(ng, ' POCZ. = ',kolp);
  znak3:=readkey;

```

```

case znak3 of
  #13:p:=kolp;
  else bytread(kolp)
end;
p:=kolp;
gt;write(ni, ' = ');bytread(j3);
if sprawdz('C',j3,true)=true then begin beep(3);exit end;
gt;write('KAT POCZ. = ');realread(b7);
if b7>10000 then begin beep(3);exit end;
gt;write('PRZYROST KATA = ');realread(j4);
if j4>10000 then begin beep(3);exit end;
gt;write('ILOSC PUNKTOW = ');bytread(p0);
if p0>255 then begin beep(3);exit end;
b7:=(b7*pi)/180;j4:=(j4*pi)/180;
for j:=p to p0+p-1 do
  begin
    tabp[j,1]:=tabc[j3,1]+tabc[j3,3]*cos(b7+(j-p)*j4);
    tabp[j,2]:=tabc[j3,2]+tabc[j3,3]*sin(b7+(j-p)*j4);
    na_ekran('P',j,kolor_p);save_poj('P',j)
  end;
if (p+p0-1) > max_nr_p then max_nr_p:=p+p0-1;
kolp:=max_nr_p
end;

procedure tra;
var j,i,b7,ul,ile:byte;a1:real;
begin
  kolp:=kolp+1;
  gt;write(ng, ' POCZ. = ',kolp);
  znak3:=readkey;
  case znak3 of
    #13:p:=kolp;
    else bytread(kolp)
  end;
  p:=kolp;
  gt;write('TRA OD ',pu,' nr ');bytread(j3);
  if sprawdz('P',j3,true)=true then begin beep(3);exit end;
  gt;write(' DO ',pu,' nr ');bytread(j4);
  if sprawdz('P',j4,true)=true then begin beep(3);exit end;
  gt;write(' CO ILE PUNKTOW ');bytread(b7);
  if b7>255 then begin beep(3);exit end;
  gt;write('PRZYROST x = ');realread(a);
  if a>10000 then begin beep(3);exit end;
  gt;write('PRZYROST y = ');realread(a1);
  if a1>10000 then begin beep(3);exit end;
  gt;write('ILOSC TRA = ');bytread(p0);
  if p0>255 then begin beep(3);exit end;
  ul:=p;x2:=a;u4:=a1;j:=j3-b7;ile:=0;
  for i:=1 to p0 do
    begin
      repeat
        j:=j+b7;
        tabp[ul,1]:=tabp[j,1]+x2;
        tabp[ul,2]:=tabp[j,2]+u4;
        na_ekran('P',ul,kolor_p);save_poj('P',ul);
        ile:=ile+1;ul:=ul+1
      until j>=j4;
      x2:=x2+a;u4:=u4+a1;j:=j3-b7
    end;
  if (p+ile-1) > max_nr_p then max_nr_p:=p+ile-1;
  kolp:=max_nr_p

```

```

end;

procedure rot;
var j,i,x1,b7,u1,ile:byte;
    p0,a1:real;
begin
    kolp:=kolp+1;
    gt;write(ng,' POCZ. = ',kolp);
    znak3:=readkey;
    case znak3 of
        #13:p:=kolp;
        else byteread(kolp)
    end;
    p:=kolp;
    gt;write('ROT OD ',pu,' nr ');byteread(j3);
    if sprawdz('P',j3,true)=true then begin beep(3);exit end;
    gt;write(' DO ',pu,' nr ');byteread(j4);
    if sprawdz('P',j4,true)=true then begin beep(3);exit end;
    gt;write(' CO ILE PUNKTOW ');byteread(x1);
    if x1>255 then begin beep(3);exit end;
    gt;write('SRODEK OBROTU x = ');realread(a);
    if a>10000 then begin beep(3);exit end;
    gt;write(' y = ');realread(a1);
    if a1>10000 then begin beep(3);exit end;
    gt;write('KAT OBROTU = ');realread(p0);
    if p0>10000 then begin beep(3);exit end;
    gt;write('ILOSC OBROTOW = ');byteread(b7);
    if b7>255 then begin beep(3);exit end;
    p0:=(p0*pi)/180;u1:=p;u2:=p0;j:=j3-x1;ile:=0;
    for i:=1 to b7 do
        begin
            repeat
                j:=j+x1;
                tabp[u1,1]:=(tabp[j,1]-a)*cos(u2)-(tabp[j,2]-a1)*sin(u2)+a;
                tabp[u1,2]:=(tabp[j,2]-a1)*cos(u2)+(tabp[j,1]-a)*sin(u2)+a1;
                na_ekran('P',u1,kolor_p);save_poj('P',u1);
                ile:=ile+1;u1:=u1+1
            until j>=j4;
            u2:=u2+p0;j:=j3-x1
        end;
    if (p+ile-1) > max_nr_p then max_nr_p:=p+ile-1;
    kolp:=max_nr_p
end;

procedure sym;
var j,i,a,r,ile:byte;
begin
    kolp:=kolp+1;
    gt;write(ng,' POCZ. = ',kolp);
    znak3:=readkey;
    case znak3 of
        #13:p:=kolp;
        else byteread(kolp)
    end;
    p:=kolp;
    gt;write('SUM OD ',pu,' nr ');byteread(j3);
    if sprawdz('P',j3,true)=true then begin beep(3);exit end;
    gt;write(' DO ',pu,' nr ');byteread(j4);
    if sprawdz('P',j4,true)=true then begin beep(3);exit end;
    gt;write(' CO ILE PUNKTOW ');byteread(a);
    if a>255 then begin beep(3);exit end;

```



```

gt;write(nh,' - OSI SYM. = ');byteread(p0);
if sprawdz('L',p0,true)=true then begin beep(3);exit end;
ggg(p0);
u3:=-u2;u4:=u1;r:=p;j:=j3-a;ile:=0;
repeat
  j:=j+a;x2:=tabp[j,1];y2:=tabp[j,2];w:=(-u2)*u3+u1*u4;
  w1:=- (x1-x2)*u4+u3*(y1-y2);
  if abs((x1-x2)*u2-(y1-y2)*u1) < liczba_min
    then begin end
    else
      begin
        tabp[254,1]:=u1*(w1/w)+x1;
        tabp[254,2]:=u2*(w1/w)+y1;
        tabp[r,1]:=2*(tabp[254,1]-tabp[j,1])+tabp[j,1];
        tabp[r,2]:=2*(tabp[254,2]-tabp[j,2])+tabp[j,2];
        na_ekran('P',r,kolor_p);save_poj('P',r);
        ile:=ile+1;r:=r+1
      end
until j>=j4;
if (p+ile-1) > max_nr_p then max_nr_p:=p+ile-1;
kolp:=max_nr_p
end;

procedure sav(znak:char;numer:byte); {zapis na dysk - zabezpieczenie}
begin
  case znak of
    'P':writeln(plik,znak,numer:4,tabp[numer,1]:20,tabp[numer,2]:20);
    'L':writeln(plik,znak,numer:4,tabl[numer,1]:20,tabl[numer,2]:20);
    'C':writeln(plik,znak,numer:4,tabc[numer,1]:20,
               tabc[numer,2]:20,tabc[numer,3]:20)
  end;
end;

procedure save_norm; {zapis zbioru plc na dysk}
var i:byte;num:string;
    plik1:text;
    zn:char;
procedure sav1(znak:char;numer:byte);
begin
  case znak of
    'P':writeln(plik1,znak,numer:4,tabp[numer,1]:20,tabp[numer,2]:20);
    'L':writeln(plik1,znak,numer:4,tabl[numer,1]:20,tabl[numer,2]:20);
    'C':writeln(plik1,znak,numer:4,tabc[numer,1]:20,
               tabc[numer,2]:20,tabc[numer,3]:20)
  end;
end;
begin
  gt;write('NAZWA RYSUNKU : ');
  num:=lancuch(7);
  if num='' then exit;
  assign(plik1,rysunki+num+rozs);
  {$I-} reset(plik1);{$I+}
  if ioreult=0 then
    begin
      gt;write('ZBIOR ISTNIEJE-ZAPISAC ? (t/n) ');
      beep(3);zn:=readkey;zn:=upcase(zn);
      if zn<>'T' then exit
    end;
  rewrite(plik1);
  writeln(plik1,num+rozs);

```

```

for i:=0 to max_nr_p do if sprawdz('P',i,false)=false then sav1('P',i);
for i:=0 to max_nr_l do if sprawdz('L',i,false)=false then sav1('L',i);
for i:=0 to max_nr_c do if sprawdz('C',i,false)=false then sav1('C',i);
close(plik1);
assign(plik,zabezpiecz);rewrite(plik);
writeln(plik,'ZABEZPIECZENIE');close(plik);
assign(plik,zabezpiecz)
end;

procedure save_dziur;           {zapis zbioru plc na dysk do odczytu przez TAPE}
var num:string;                {na tasme dziorkowana}
    znak:char;i,j:byte;
    plik1:text;numer1:real;
procedure sav2(znak:char;numer:byte);
begin
    case znak of
        'P':begin
            writeln(plik1,znak);writeln(plik1,numer);
            writeln(plik1,tabp[numer,1]:10:3,tabp[numer,2]:10:3)
        end;
        'L':begin
            writeln(plik1,znak);writeln(plik1,numer);
            if tabl[numer,1]=9.0E36 then numer1:=99999
                else numer1:=tabl[numer,1];
            writeln(plik1,numer1:10:3,tabl[numer,2]:10:3)
        end;
        'C':begin
            writeln(plik1,znak);writeln(plik1,numer);
            writeln(plik1,tabc[numer,1]:10:3,
                tabc[numer,2]:10:3,tabc[numer,3]:10:3)
        end
    end;
end;
end;
begin
    gt;write('NAZWA RYSUNKU : ');
    num:=lancuch(7);
    assign(plik1,rysunki+num+'$'+rozs);
    rewrite(plik1);writeln(plik1,num+rozs);
    for i:=0 to max_nr_p do if sprawdz('P',i,false)=false then sav2('P',i);
    for i:=0 to max_nr_l do if sprawdz('L',i,false)=false then sav2('L',i);
    for i:=0 to max_nr_c do if sprawdz('C',i,false)=false then sav2('C',i);
    writeln(plik1,'T');close(plik1)
end;

procedure load_norm;           {odczytanie z dysku danych plc}
var numer:byte;num:string;plik1:text;
    procedure lod1(znak:char);
    begin
        case znak of
            'P':begin
                readln(plik1,numer,tabp[numer,1],tabp[numer,2]);
                if numer>max_nr_p then max_nr_p:=numer
            end;
            'L':begin
                readln(plik1,numer,tabl[numer,1],tabl[numer,2]);
                if numer>max_nr_l then max_nr_l:=numer
            end;
            'C':begin
                readln(plik1,numer,tabc[numer,1],
                    tabc[numer,2],tabc[numer,3]);
                if numer>max_nr_c then max_nr_c:=numer
            end;
        end;
    end;
end;

```

```

        end
    end;
end;
begin
    gt;write('NAZWA RYSUNKU : ');
    num:=lancuch(7);
    assign(plik1,rysunki+num+rozs);
    {$I-} reset(plik1);{$I+}
    if ioreult<>0 then begin beep(3);exit end;
    max_nr_p:=0;max_nr_l:=0;max_nr_c:=0;
    readln(plik1,num);
    while not eof(plik1) do
    begin read(plik1,znak);lod1(znak) end;
    close(plik1);
    kolp:=max_nr_p;koll:=max_nr_l;kolc:=max_nr_c;
    skalay:=skalayp*skala;skalax:=skalaxp*skala;
    przerysuj
end;

procedure TTT;          {odczyt danych z tablicy pojedynczego elementu}
var znak:char;numer:byte;          {i zmiana koloru}
begin
    repeat
        while keypressed do znak:=readkey;
        gt;write('P/L/C/ESC-koniec/BS-kolor');
        znak:=readkey;znak:=upcase(znak);write(znak);
        IF ZNAK=#27 THEN exit;
        if znak=#8 then
            if tryb=1 then
                begin
                    kolor:=((kolor+1) mod 4);palette(kolor);
                    skalaxp:=1.25;przesx:=160
                end
            else
                begin
                    kolor:=((kolor+1) mod 16);hirescolor(kolor);
                    skalaxp:=2.5;przesx:=320
                end
            else
                begin gt;write('NUMER : ');byteread(numer) end;
        case znak of
            'P':begin
                na_ekran('P',numer,-1);
                repeat until keypressed;
                na_ekran('P',numer,kolor_p)
            end;
            'L':begin
                na_ekran('L',numer,-1);
                repeat until keypressed;
                na_ekran('L',numer,kolor_l)
            end;
            'C':begin
                na_ekran('C',numer,-1);
                repeat until keypressed;
                na_ekran('C',numer,kolor_c)
            end
        end;
    end;
    repeat until keypressed;
until (znak='P') or (znak='L') or (znak='C') or (znak=#27)
end;

```

```

procedure wsp_wsk;          {procedury do obsługi wskaznika}
  var x,y:real;              {obliczenie wspolrzednej srodka}
begin
  x:=(wsk_x-przesx)/skalax+xsr1;y:=(przesy-wsk_y)/skalay+ysr1;
  if tryb =1 then
    begin gotoxy(17,1);write('X ',x:7:2,' Y ',y:7:2) end
  else begin gotoxy(58,1);write('X ',x:7:2,' Y ',y:7:2) end;
  gotoxy(1,1);write('THH - POMOC');
end;

```

```

procedure rys_wsk(wsk_x2,wsk_y2:integer;jak:rys); {rysowanie wskaznika}
begin
  draw(wsk_x2,wsk_y2-2,wsk_x2,0,jak);
  draw(wsk_x2,wsk_y2+2,wsk_x2,199,jak);
  draw(wsk_x2-3,wsk_y2,0,wsk_y2,jak);
  draw(wsk_x2+3,wsk_y2,639,wsk_y2,jak);
  wsp_wsk
end;

```

```

procedure wskaznik;      {obsługa wskaznika}
begin
  if (wsk_x<640) or (wsk_x>0) or (wsk_y>0) or (wsk_y<200) then
    begin
      rys_wsk(wsk_x1,wsk_y1,-1);
      rys_wsk(wsk_x,wsk_y,-1);
      wsk_x1:=wsk_x;wsk_y1:=wsk_y
    end
  else begin wsk_x:=20;wsk_x1:=-10;wsk_y:=180;wsk_y1:=-190 end
end;

```

```

procedure TSS;{PRZESUWAMY SRODEK EKRANU TAK JAK WSPOLRZEDNE}
var x,y:real;          {i skalowanie}
    zn:char;
begin
  x:=(wsk_x-przesx)/skalax+xsr1;y:=(przesy-wsk_y)/skalay+ysr1;
  gt;write('PRZESUN W POZIOMIE DO : ',x:10:3);
  zn:=readkey;
  if zn=#13 then xsr1:=x
  else
    begin
      gotoxy(25,1);write(' ');
      gotoxy(25,1);realread(xsr1)
    end;
  gt;write('PRZESUN W PIONIE DO : ',y:10:3);
  zn:=readkey;
  if zn=#13 then ysr1:=y
  else
    begin
      gotoxy(25,1);write(' ');
      gotoxy(25,1);realread(ysr1)
    end;
  gt;write('POWIEKSZENIE (',skala:10:2,') : ');realread(skala);
  skalay:=skalayp*skala;skalax:=skalaxp*skala;
  przerysuj
end;

```

```

procedure TPP;          {odleglosc miedzy dwoma punktami}
var odl:real;
begin
  gt;write(ng,' PIERWSZEGO = ');byteread(j1);
  gt;write(ng,' DRUGIEGO = ');byteread(j2);

```

```

    odl:=sqrt(sqr(tabp[j2,1]-tabp[j1,1])+sqr(tabp[j2,2]-tabp[j1,2]));
    gt;write('ODLEGLOSC = ',odl:10:3);
    repeat until keypressed
end;

                                {na ekran wspolrzedne elementu i jego rysunek}
procedure na_ekran(znak2:char;num:byte;jak:rys);
begin
    gt;
    case znak2 of
        'P':begin
            if sprawdz('P',num,true)=true then begin end
            else
                begin
                    if num in [0..220] then write(num);
                    write('  X =',tabp[num,1]:8:3,'  Y =',tabp[num,2]:8:3);
                    rysuj_punkt(num,jak)
                end
            end;
        'L':begin
            if sprawdz('L',num,true)=true then begin end
            else
                begin
                    rysuj_linie1(num,jak);
                    if num in [0..220] then write(num);
                    write('  KAT = ',kat1:8:3)
                end
            end;
        'C':begin
            if sprawdz('C',num,true)=true then begin end
            else
                begin
                    if num in [0..220] then write(num);
                    write(' X=',tabc[num,1]:8:3,'  Y=',tabc[num,2]:8:3,
                        '  R=',tabc[num,3]:8:3);
                    rysuj_okrag1(num,jak)
                end
            end
        end
    end;
end;

procedure menu;          {obsługa tablic z plc}
var znak:char;num1,numl1:byte;jak1:rys;
    procedure nastepny;   {nastepny element z aktualnej tablicy}
    var ile:byte;
    begin
        numl1:=num1;
        if znak1='B' then
            if num1<249 then num1:=num1+1 else num1:=230
            else
                begin
                    case znak2 of
                        'P':ile:=max_nr_p;
                        'L':ile:=max_nr_l;
                        'C':ile:=max_nr_c
                    end;
                    if num1<ile then num1:=num1+1 else num1:=0
                end
            end;
    end;

procedure poprzedni;     {poprzedni element}

```

```

begin
  numl1:=numl;
  if znak1='B' then
    if numl>230 then numl:=numl-1 else numl:=249
  else
    if numl>0 then numl:=numl-1
    else
      begin
        case znak2 of
          'P':numl:=max_nr_p;
          'L':numl:=max_nr_l;
          'C':numl:=max_nr_c
        end
      end
    end
  end;

procedure do_menu; {przeniesienie elementu z bufora na ekran i do tablicy}
var nr:byte;
begin
  gt;
  case znak2 of
    'P':begin
      kolp:=kolp+1;
      write('DEF PUNKTU NR ',kolp);
      znak3:=readkey;
      if znak3=#13 then nr:=kolp
      else begin bytread(nr);kolp:=nr end;
      if nr>max_nr_p then max_nr_p:=nr;
      tabp[nr]:=tabp[numl];
      numl1:=0;na_ekran(znak2,nr,kolor_p);
      save_poj(znak2,nr)
    end;
    'L':begin
      koll:=koll+1;
      write('DEF PROSTEJ NR ',koll);
      znak3:=readkey;
      if znak3=#13 then nr:=koll
      else begin bytread(nr);koll:=nr end;
      if nr>max_nr_l then max_nr_l:=nr;
      tabl[nr]:=tabl[numl];
      numl1:=0;na_ekran(znak2,nr,kolor_l);
      save_poj(znak2,nr)
    end;
    'C':begin
      kolc:=kolc+1;
      write('DEF OKREGU NR ',kolc);
      znak3:=readkey;
      if znak3=#13 then nr:=kolc
      else begin bytread(nr);kolc:=nr end;
      if nr>max_nr_c then max_nr_c:=nr;
      tabc[nr]:=tabc[numl];
      numl1:=0;na_ekran(znak2,nr,kolor_c);
      save_poj(znak2,nr)
    end
  end
end;

procedure do_buf; {przeniesienie elementu z tablicy do bufora 249 - zmazanie}
begin
  case znak2 of
    'P':begin tabp[249]:=tabp[numl];tabp[numl]:=tabp[0] end;
  end;
end;

```

```

'L':begin tabl[249]:=tabl[numl];tabl[numl]:=tabl[0] end;
'C':begin tabc[249]:=tabc[numl];tabc[numl]:=tabc[0] end
end;
case znak2 of
  'P':na_ekran('P',numl,-1);
  'L':na_ekran('L',numl,-1);
  'C':na_ekran('C',numl,-1)
end;
beep(3)
end;
begin
if znak1='B' then numl:=230 else numl:=1;
rys_wsk(wsk_x,wsk_y,0);
case znak2 of
  'P':na_ekran('P',numl,-1);
  'L':na_ekran('L',numl,-1);
  'C':na_ekran('C',numl,-1)
end;
repeat
while keypressed do znak:=readkey;
if tryb=1 then gotoxy(17,1) else gotoxy(58,1);
if znak1='B' then write('BUFOR          koniec-ESC')
                    else write('TABLICA      koniec-ESC');
znak:=readkey;
case znak of
  #13:if znak1='B' then do_menu;
  #9:if znak1='M' then do_buf;
  #27:przerysuj;
  #0:begin
      znak:=readkey;
      if znak1='M' then
        case znak2 of
          'P':na_ekran('P',numl,kolor_p);
          'L':na_ekran('L',numl,kolor_l);
          'C':na_ekran('C',numl,kolor_c)
        end
      else
        case znak2 of
          'P':na_ekran('P',numl,-1);
          'L':na_ekran('L',numl,-1);
          'C':na_ekran('C',numl,-1)
        end;
      case znak of
        #77:nastepny;
        #75:poprzedni;
        else beep(3)
      end;
      case znak2 of
        'P':na_ekran('P',numl,-1);
        'L':na_ekran('L',numl,-1);
        'C':na_ekran('C',numl,-1)
      end;
    end
  else begin
    gt;beep(1);
    if znak1='B' then write('PROSZE OPUSCIC BUFOR <ESC>')
                    else write('PROSZE OPUSCIC TABLICE <ESC>');
    beep(6)
  end
end;
until (znak=#27)

```

```

end;

procedure reset;          {reset}
var znak:char;
begin
  gt;write('NA PEWNO RESET (t/n) ? ');
  znak:=readkey;znak:=upcase(znak);if znak<>'T' then exit;
  max_nr_p:=0;max_nr_l:=0;max_nr_c:=0;
  xsrl:=0;ysrl:=0;kolp:=0;koll:=0;kolc:=0;
  zeruj('P',0,200);zeruj('L',0,200);zeruj('C',0,200);
  przerysuj
end;

procedure druk;          {wydruk elementow na drukarce}
var pie,ost,i:byte;znak,zn:char;liczba:real;num:string;
  procedure gwiazdki;
  var i:byte;
  begin for i:=1 to 60 do write(lst,'*'); writeln(lst) end;

  procedure wydr;
  var i:byte;
  begin
    case znak of
      'P':for i:=pie to ost do
        if sprawdz('P',i,false)=false then
          writeln(lst,'          P',i:3,' ', 'X=',tabp[i,1]:8:3,
            '          Y=',tabp[i,2]:8:3);
      'L':for i:=pie to ost do
        if sprawdz('L',i,false)=false then
          begin
            wspx:=0;wspy:=0;wsplinii(wspx,wspy,i);
            if wspy>0 then zn:='+' else zn:='-';
            wspy:=abs(wspy);
            write(lst,'          L',i:3,' ', 'Y=',kat:8:3,' X ',zn,wspy:8:3);
            writeln(lst,'          KAT = ',kat1:8:3)
          end;
      'C':for i:=pie to ost do
        if sprawdz('C',i,false)=false then
          writeln(lst,'          C',i:3,' ', 'X=',tabc[i,1]:10:3,
            '          Y=',tabc[i,2]:10:3,'          R=',tabc[i,3]:10:3);
        else exit
    end
  end;
end;

begin
  gt;write('NAZWA RYSUNKU : ');
  num:=copy(lancuch(7),1,8);
  gt;write('P/L/C/W-wszystkie/ESC-koniec');
  znak:=readkey;znak:=upcase(znak);
  if znak=#27 then exit;
  if znak='W' then
    begin
      pie:=0;writeln(lst,num+'.RYS');gwiazdki;
      ost:=max_nr_p;znak:='P';wydr;gwiazdki;
      ost:=max_nr_l;znak:='L';wydr;gwiazdki;
      ost:=max_nr_c;znak:='C';wydr;gwiazdki;
      writeln(lst,#12)
    end
  else
    begin
      gt;write('PIERWSZY ');byteread(pie);

```



```

        gt;write('OSTATNI      ');byteread(ost);
        writeln(lst,num+'.RYS');gwiazdki;wydr;gwiazdki
    end;
end;

procedure mono_kolor;
begin
    if tryb=1 then tryb:=2 else tryb:=1;
    if tryb=1 then
        begin
            kolor_p:=1;kolor_l:=2;kolor_c:=3;
            graphcolormode;palette(3);colortable(1,0,0,0);
            skalaxp:=1.25;przesx:=160;skalax:=skalaxp*skala;
        end
    else
        begin
            kolor_p:=1;kolor_l:=1;kolor_c:=1;
            hires;hirescolor(kolor);graphwindow(0,0,640,200);colortable(1,0,1,0);
            skalaxp:=2.5;przesx:=320;skalax:=skalaxp*skala;
        end;
    przerysuj
end;

procedure direkt;
var droga:string;
    zbior:searchrec;
    i:byte;
    znak:char;
begin
    graphwindow(0,0,160,200);
    fillscreen(0);
    draw(1,1,159,1,1);
    draw(159,1,159,199,1);
    draw(159,199,1,199,1);
    draw(1,199,1,1,1);
    droga:='rys\*.rys';
    findfirst(droga,0,zbior);
    i:=2;
    with zbior do
        begin
            gotoxy(5,i);
            write(copy(name,1,(length(name)-4)));
        end;
    znak:='D';
    while (doserror=0) and (znak='D') do
        begin
            repeat
                findnext(zbior);
                i:=i+1;
                with zbior do
                    begin
                        gotoxy(5,i);
                        write(copy(name,1,(length(name)-4)));
                    end;
            until i=23;
            i:=2;
            writeln;
            gotoxy(wherex+1,wherey);
            writeln('(K)oniec / (D)alej');
            znak:=readkey;znak:=upcase(znak);
            fillscreen(0);

```

```

        draw(1,1,159,1,1);
        draw(159,1,159,199,1);
        draw(159,199,1,199,1);
        draw(1,199,1,1,1);
        if (znak<>'D') and (znak<>'K') then znak:='D';
    end;
    graphwindow(0,0,640,200);
    przerysuj;
end;

procedure sprawdz_wsk;      {obsługa klawiszy z kodem #0}
begin
    znak3:=readkey;
    case znak3 of
        't':begin wsk_x1:=wsk_x;wsk_x:=wsk_x+1 end;
        's':begin wsk_x1:=wsk_x;wsk_x:=wsk_x-1 end;
        'G':begin wsk_y1:=wsk_y;wsk_y:=wsk_y-1 end;
        'S':begin xsr1:=80/skala+xsr1; przerysuj end;
        'R':begin xsr1:=-80/skala+xsr1; przerysuj end;
        'I':begin ysr1:=50/skala+ysr1; przerysuj end;
        'Q':begin ysr1:=-50/skala+ysr1; przerysuj end;
        #77:begin wsk_x1:=wsk_x;wsk_x:=wsk_x+20 end;
        #75:begin wsk_x1:=wsk_x;wsk_x:=wsk_x-20 end;
        #72:begin wsk_y1:=wsk_y;wsk_y:=wsk_y-10 end;
        #80:begin wsk_y1:=wsk_y;wsk_y:=wsk_y+10 end;
        #59:pomoc;
        #60:save_norm;
        #61:load_norm;
        #62:druk;
        #63:save_dziur;
        #64:tss;
        #65:tpp;
        #66:ttt;
        #67:reset;
        #68:kod:='T**';
        #84:begin
            numeracja_p:=true;numeracja_l:=true;numeracja_c:=true;przerysuj
        end;
    #85,#86:begin
        case znak3 of
            #85:znak1:='B';
            #86:znak1:='M'
        end;
        znak:='T';
        case kod[1] of
            'P','L','C':znak2:=kod[1]
            else znak2:='C'
        end;
        menu
    end;
    #87:begin
        skala:=skala*0.5;
        skalax:=skalaxp*skala;skalay:=skalayp*skala;
        przerysuj
    end;
    #88:begin
        skala:=skala/0.5;
        skalax:=skalaxp*skala;skalay:=skalayp*skala;
        przerysuj
    end;
    #89:mono_kolor;
end;

```

```

    #90:direkt;
    { #91:}
end
end;

procedure numeracja;          {zmiana numeracji elementow}
begin
    numeracja_p:=false;numeracja_l:=false;numeracja_c:=false;
    case znak2 of
        'P':numeracja_p:=true;
        'L':numeracja_l:=true;
        'C':numeracja_c:=true
    end;
    przerysuj
end;

procedure defplc(i:byte);
begin
    if i in [38..42] then exit;
    case znak of
        'P':begin
            if kolp<229 then kolp:=kolp+1;
            write('DEF PUNKTU NR ',kolp);
            znak3:=readkey;
            if znak3=#13 then p:=kolp else begin bytread(p);kolp:=p end;
            if p>max_nr_p then max_nr_p:=p
            end;
        'L':begin
            if koll<229 then koll:=koll+1;
            write('DEF PROSTEJ NR ',koll);
            znak3:=readkey;
            if znak3=#13 then p:=koll else begin bytread(p);koll:=p end;
            if p>max_nr_l then max_nr_l:=p
            end;
        'C':begin
            if kolc<229 then kolc:=kolc+1;
            write('DEF OKREGU NR ',kolc);
            znak3:=readkey;
            if znak3=#13 then p:=kolc else begin bytread(p);kolc:=p end;
            if p>max_nr_c then max_nr_c:=p
            end
        end
    end
end;

begin
    assign(plik,zabezpiecz);
    pozycja:=25;tryb:=2;
    if tryb=1 then
        begin
            kolor_p:=1;kolor_l:=2;kolor_c:=3;
            graphcolormode;palette(3);colortable(1,0,0,0);
            skalaxp:=1.25;przesx:=160
        end
    else
        begin
            kolor_p:=1;kolor_l:=1;kolor_c:=1;
            hires;hirescolor(kolor);graphwindow(0,0,640,200);colortable(1,0,1,0);
            skalaxp:=2.5;przesx:=320
        end;
    v9:=0;d:=liczba_min;kolp:=0;koll:=0;kolc:=0;
    skalay:=skalayp*skala;skalax:=skalaxp*skala;

```

```

na_czesci;
zeruj('P',0,255);zeruj('L',0,255);zeruj('C',0,255);
tabp[0,1]:=0;tabp[0,2]:=0;tabl[255,1]:=-9999;tabl[255,2]:=0;
tabl[211,1]:=liczba_max;tabl[212,1]:=liczba_max;
tabl[211,2]:=0;tabl[212,2]:=90;
wsk_x:=20;wsk_x1:=-10;wsk_y:=180;wsk_y1:=-190;
repeat
  while keypressed do znak:=readkey;
  blad:=false;
  wskaznik;
  gt;write('DEF  KOD:  ');
  znak:=readkey;znak:=upcase(znak);write(znak);
  case znak of
'T','P',
'L','C',
'R','S':begin
  znak1:=readkey;znak1:=upcase(znak1);write(znak1);
  znak2:=readkey;znak2:=upcase(znak2);write(znak2);
  kod:=znak+znak1+znak2;
  i:=0;repeat i:=i+1;until (proc[i]=kod) or (i=l_proc+1);
  if i<=l_proc then
    begin
      gt;beep(1);defplc(i);
      case i of
        1:pxy;
        2:prq;
        3:plcm;
        4:p11;
        5:pccm;
        6:psc;
        7:lpp;
        8:lpcm;
        9:l1dm;
        10:lpq;
        11:lcqm;
        12:lccm;
        13:cpr;
        14:cplm;
        15:c11m;
        16:cppm;
        17:clcm;
        18:cccm;
        19:cpcm;
        20:ttt;
        21:tss;
        22:tpp;
        23:save_norm;
        24:load_norm;
        25:pomoc;
        26:druk;
        27:reset;
        28:save_dziur;
        29..34:menu;
        35..37:numeracja;
        38:pli;
        39:pcr;
        40:tra;
        41:rot;
        42:sym;
        43:mono_kolor;
        44:direkt;

```

```
        end
    end
    else beep(3)
end;
#27:begin
    numeracja_p:=false;
    numeracja_l:=false;
    numeracja_c:=false;
    przerysuj
end;
#0:sprawdz_wsk
else beep(3)
end;
if kod='T**' then
    begin
        gt;write('NA PEWNO KONIEC (t/n) ? ');
        znak5:=readkey;znak5:=upcase(znak5);
        if znak5<>'T' then kod:='PPP';
    end
until kod='T**';
hires;
textmode(lastmode);
end.
```