

mgr inż. Stanisław Ubermanowicz

Piotr Fiorek

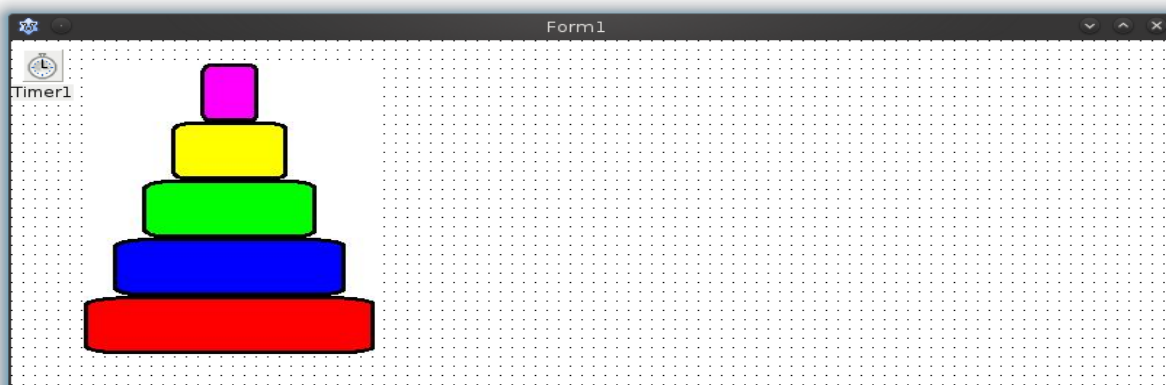
Wieże Hanoi – pokaz

Zaprojektuj program służący do prezentacji algorytmu z łamigłówki Wieże Hanoi. Program ma animować ruch obiektów (imitujących pierścienie) co 1 sekundę. Łamigłówka, to zadanie polegające na przestawieniu pojedynczo pięciu obiektów z lewego cokołu docelowo na cokół prawy, z pomocą cokołu środkowego.

Warunek: można kłaść tylko obiekty mniejsze na większe.



Okno formularza:



W oknie formularza należy umieścić kolejno pięć pól na obrazki (TImage) które będą przedstawiać kolejne krążki oraz obiekt Timera (TTimer).

W oknie „Object Inspector” można również ustawić nazwy poszczególnych elementów. W tej implementacji używamy nazw:

- ☐ „Krazek1”, „Krazek2”, „Krazek3”, „Krazek4”, „Krazek5” dla obiektów typu TImage przedstawiających kolejne krążki
- ☐ „Timer1” dla obiektu typu TTimer

Kod:

W celu uproszczenia obliczeń oznaczono POZYCJĘ z lewej jako 0, środkową 1, a prawą jako 2.

```

1 unit wieze_hanoi;
2 {$mode objfpc}{$H+}
3 interface
4 uses
5   Classes, SysUtils, FileUtil, LResources, Forms, Controls, Graphics, Dialogs,
   ExtCtrls;
6 type
7   {TOkno}
8   TOkno = class(TForm)
9     Krazek1: TImage;           {Zmienne reprezentujące obiekty}
10    Krazek2: TImage;
11    Krazek3: TImage;
12    Krazek4: TImage;
13    Krazek5: TImage;
14    Timer1: TTimer;
15    procedure FormCreate(Sender: TObject);           {deklaracja procedury
startowej}
16    procedure Timer1Timer(Sender: TObject);           {procedura wykonywana co
sekundę}
17  private
18    {private declarations}
19    ileP: array[0..2] of Integer; {Tablica 3-elementowa, pamięta liczby
określające ile pierścieni (od 0 do 5) jest w danej chwili na pozycjach 0, 1 i
2}
20    pozP: array[1..5] of Integer; {Tablica 5-elementowa pamięta liczby
określające aktualne pozycje (od 0 do 2) pięciu pierścieni}
21    i: Integer;                {Zmienna iteracyjna, wskazuje numer ruchu (liczbę
ruchów)}
22  public
23    {public declarations}
24  end;
25 var
26   Okno: TOkno;
27 implementation
28 {TOkno}

```

```

procedure TOkno.FormCreate(Sender: TObject); {Procedura wykonywana przy tworzeniu
okna, jej deklaracja tworzona jest po dwukrotnym kliknięciu na pustą przestrzeń
formularza}
30 begin                                     {przypisanie wartości początkowych wszystkim zmiennym}
31   ileP[0]:= 5;   ileP[1]:= 0;   ileP[2]:= 0;
32   pozP[1]:= 0;   pozP[2]:= 0;   pozP[3]:= 0;   pozP[4]:= 0;   pozP[5]:= 0;
33 end;
34 procedure TOkno.Timer1Timer(Sender: TObject); {Procedura wywoływana cyklicznie
przez Timer co sekundę; jej deklaracja tworzona jest po dwukrotnym kliknięciu w
obiekt Timer}
35 begin
36   i:= i+1;                                {Liczenie ruchów}
37   if((i-1) mod 2)=0 then {Przy ruchach nieparzystych obsłuż obiekt najmniejszy -
pierwszy}
38   begin
39     ileP[pozP[1]]:= ileP[pozP[1]]-1; {Na dotychczasowej pozycji zmniejsz
liczbę, gdyż ma stamtąd ubyc jeden obiekt}
40     pozP[1]:= pozP[1]-1;             {Cofnij w lewo pozycję obiektu}
41     if pozP[1]<0 then pozP[1]:= 2;    {Jeśli był on na pozycji 0, to zmień na
pozycję 2}
42     ileP[pozP[1]]:= ileP[pozP[1]]+1; {Na nowej pozycji zwiększ liczbę
obektów}
43     Krazek1.Left:= 50 + pozP[1] * 250; {Przesuń obiekt poziomo, w osi x}
44     Krazek1.Top:= 270 - ileP[pozP[1]] * 50; {Przesuń obiekt pionowo, w osi y}
45   end
46   else if((i-16) mod 32)=0 then      {Co 32 ruchy obsłuż obiekt największy -
piąty}
47   begin
48     ileP[pozP[5]]:= ileP[pozP[5]]-1; {Na dotychczasowej pozycji zmniejsz
liczbę}
49     pozP[5]:= pozP[5]-1;             {Cofnij w lewo pozycję obiektu}
50     if pozP[5]<0 then pozP[5]:= 2;    {Jeśli był on na pozycji 0, to zmień na
pozycję 2}
51     ileP[pozP[5]]:= ileP[pozP[5]]+1; {Na nowej pozycji zwiększ liczbę
obektów}
52     Krazek5.Left:= 50 + pozP[5] * 250; {Przesuń obiekt poziomo, w osi x}
53     Krazek5.Top:= 270 - ileP[pozP[5]] * 50; {Przesuń obiekt pionowo, w osi y}
54   end
55   else if((i-8) mod 16)=0 then       {Co 16 ruchów obsłuż krążek czwarty}
56   begin
57     ileP[pozP[4]]:= ileP[pozP[4]]-1; {Na dotychczasowej pozycji zmniejsz
liczbę}
58     pozP[4]:= pozP[4]+1;             {Przesuń w prawo pozycję obiektu}
59     if pozP[4]>2 then pozP[4]:= 0;    {Jeśli był na pozycji 2, to zmień na
pozycję 0}
60     ileP[pozP[4]]:= ileP[pozP[4]]+1; {Na nowej pozycji zwiększ liczbę
obektów}
61     Krazek4.Left:= 50 + pozP[4] * 250; {Przesuń obiekt poziomo, w osi x}
62     Krazek4.Top:= 270 - ileP[pozP[4]] * 50; {Przesuń obiekt pionowo, w osi y}
63   end
64   else if((i-4) mod 8)=0 then        {Co 8 ruchów obsłuż krążek trzeci}

```

```

begin
66   ileP[pozP[3]]:= ileP[pozP[3]]-1; {na dotychczasowej pozycji zmniejsz liczbę}
67   pozP[3]:=pozP[3]-1; {cofnij w lewo pozycję obiektu}
68   if pozP[3]<0 then pozP[3]:= 2; {jeśli był na pozycji 0, to zmień na pozycję
2}
69   ileP[pozP[3]]:= ileP[pozP[3]]+1; {na nowej pozycji zwiększ liczbę obiektów}
70   Krazek3.Left:= 50 + pozP[3] * 250; {przesuń obiekt w osi x}
71   Krazek3.Top:= 270 - ileP[pozP[3]] * 50; {przesuń obiekt w osi y}
72 end
73 else if((i-2) mod 4)=0 then           {Co 4 ruchy obsłuż krążek drugi}
74 begin
75   ileP[pozP[2]]:= ileP[pozP[2]]-1; {na dotychczasowej pozycji zmniejsz liczbę}
76   pozP[2]:= pozP[2]+1; {przesuń w prawo pozycję obiektu}
77   if pozP[2]>2 then pozP[2]:= 0; {jeśli był na pozycji 2, to zmień na pozycję
0}
78   ileP[pozP[2]]:= ileP[pozP[2]]+1; {na nowej pozycji zwiększ liczbę obiektów}
79   Krazek2.Left:= 50 + pozP[2] * 250; {przesuń obiekt w osi x}
80   Krazek2.Top:= 270 - ileP[pozP[2]] * 50; {przesuń obiekt w osi y}
81 end;
82 if ileP[2]=5 then Timer1.Enabled:=False;   {jeśli już wszystkie obiekty
przeniesione,}
83 end;                                         {to zakończ generowanie przerw
przez Timer}
84 initialization
85   {$I wieze_hanoi.lrs}
86 end.

```

Optymalna strategia przy ręcznym przemieszczaniu obiektów.

Dla parzystej liczby obiektów:

- obiekty o indeksach nieparzystych krążą w prawo, wracając skokiem z cokołu ostatniego {1,2,0,1,2, ...}, a o indeksach parzystych odwrotnie, skaczą najpierw na cokół ostatni i wracając w lewo {2,1,0,2,1...};
- oznacza to, że obiekt najmniejszy z indeksem 1 rozpoczyna przechodząc z cokołu 0 na cokół środkowy 1.

Dla nieparzystej liczby obiektów:

- obiekty o indeksach nieparzystych skaczą najpierw na cokół ostatni i wracając w lewo {2,1,0,2,1...}, a o indeksach parzystych odwrotnie, krążą w prawo i wracają skokiem z cokołu ostatniego {1,2,0,1,2,...}

Na przemian, przy każdym kroku o numerze nieparzystym co drugie posunięcie wykonuje obiekt najmniejszy, a dla pozostałych obiektów pozostaje tylko jedyny możliwy ruch podczas kroków parzystych.

UWAGA: Podczas zapisywania projektu nie wolno zapisywać plików ".pas" (plik z kodem) oraz ".lpi" (plik projektu) pod tymi samymi nazwami.