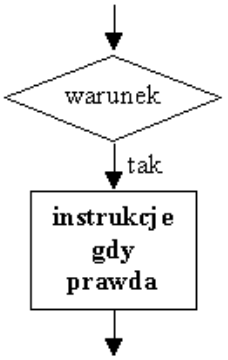
LEKCJA 7 – INSTRUKCJA WARUNKOWA

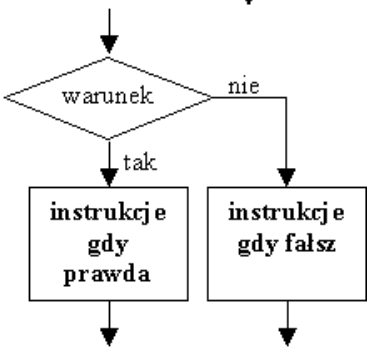


Tylko bardzo prosty program można skonstruować tak, aby wszystkie instrukcje wykonywane były kolejno. Bardzo często zachodzi sytuacja, w której musimy zadecydować, czy wykonywać te, czy też inne instrukcje. **Instrukcja warunkowa** pełni taką rolę. W zależności od wyniku operacji logicznej (poznanej na poprzedniej lekcji), której wynikiem może być PRAWDA lub FAŁSZ decydujemy, które instrukcje wykonać.

IF warunek logiczny **THEN** instrukcja

IF x < 640 THEN x:=x+1;

Jeśli warunek x<640 jest prawdziwy, to wykonaj instrukcję x:=x+1
Jeśli w zmiennej X znajduje się liczba mniejsza od 640, to zwiększ X o 1 (jeśli X jest równe lub większe to wykonuj kolejną instrukcję programu)



IF warunek logiczny **THEN** instrukcja1 **ELSE** instrukcja2

IF x < 640 THEN x:=x+1 ELSE x:=0;

Jeśli warunek x<640 jest prawdziwy, to wykonaj instrukcję x:=x+1, w przeciwnym wypadku wykonaj instrukcję x:=0 (gdy warunek jest fałszywy).

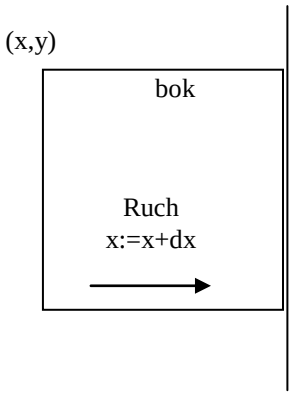
PRZYKŁAD LEKCJA071 - Animacja kwadratu z poprzedniej lekcji z wykorzystaniem warunków logicznych.

- nowa aplikacja
- deklaracja zmiennej globalnej **KONIEC:BOOLEAN=FALSE;**
- zdarzenie ONKEYDOWN – wywołać procedurę i wpisać **KONIEC:=TRUE;** (koniec animacji, gdy dowolny klawisz)
- napisać procedurę rysowania kwadratu (ramka obok)
- nagłówek procedury (bez TForm1.) umieścić po słowie TYPE
- w procedurze do zdarzenia ONRESIZE formatki zadeklarować:
  var x,y,bok,dx,dy:integer;
- wpisać fragment programu odpowiedzialny za animację i sprawdzanie odbić od krawędzi (0,0,640,480) (ramka poniżej)

```
procedure TForm1.KWADRAT
      (x,y,bok,kolor:integer);
begin
  with canvas do
  begin
    pen.color:=kolor;
    moveto(x,y);
    lineto(x+bok,y);
    lineto(x+bok,y+bok);
    lineto(x,y+bok);
    lineto(x,y);
  end;
end;
```

Odbicia

Jeżeli kwadrat przesuwa się w prawo, to zwiększa się jego współrzędna x. Sprawdzamy za pomocą warunku logicznego, kiedy współrzędna x osiągnie wartość prawego brzegu ekranu (640) i zmieniamy kierunek ruchu na przeciwny.



IF x+bok > 640 THEN dx:=-dx;

JEŻELI prawy bok kwadratu (X+BOK) jest większy od 640 (zaczyna znikać,)TO zmieniamy kierunek na przeciwny (gdy dx=1 to zmieni się na -1 i na odwrót)

UWAGA - Nie stosuje się warunków logicznych typu X+BOK = 640. Gdyby ruch odbywał się np., co 3 piksele (DX=3), mogłoby okazać się, że współrzędna X nigdy nie osiągnie wartości 640 (…, 636, 639, 642, 645, …) i kwadrat nie odbije się.

```
procedure TForm1.FormResize
          (Sender: TObject);
var x,y,bok,dx,dy:integer;
begin
  x:=0;
  y:=0;
  dx:=1;
  dy:=1;
  bok:=100;
  repeat
    kwadrat(x,y,bok,clRed);
    sleep(10);
    kwadrat(x,y,bok,clBtnFace);
    x:=x+dx;
    y:=y+dy;
    if (x+bok > 640) or (x < 0) then dx:=-dx;
    if (y+bok > 480) or (y < 0) then dy:=-dy;
    application.ProcessMessages;
  until KONIEC;
end;
```

Z matematycznego punktu widzenia nie ma różnicy pomiędzy zapisem X+BOK > 640 oraz X > 640-BOK.
Po zmianie wartości w zmiennej DX na przeciwną wartości we współrzędnej X będą się za każdym razem zmniejszać.

W podobny sposób sprawdzamy, czy kwadrat osiągnie lewy brzeg ekranu. Gdy przekroczy lewy brzeg ekranu, zmieniamy znak DX na przeciwny (był ujemny, a teraz stanie się dodatni).

if x < 0 then dx:=-dx;

Wykorzystując operator sumy logicznej, można te dwa warunki połączyć

if (x+bok > 640) or (x < 0) then dx:=-dx;

W identyczny sposób należy sprawdzić ruch po współrzędnej Y

if (y+bok > 480) or (y < 0) then dy:=-dy;

Jeśli jednocześnie będziemy zmieniać współrzędną X i Y, kwadrat będzie poruszał się po przekątnych. Jeśli wartości DX i DY będą jednakowe to pod kątem 45 stopni. Gdy te wartości będą się zmieniać, to kwadrat zacznie poruszać się po innych przekątnych.

PRZYKŁAD LEKCJA072 – animacja za pomocą komponentu TIMER

- zadeklarować zmienne globalne: x,y,dx,dy,bok
- w procedurze do zdarzenia ONCREATE dla formatki ustawić początkowe wartości zmiennych jak w lekcja071 lub podobnie jak na poprzedniej lekcji od razu podczas deklarowania
- właściwość INTERWAL dla TIMERa ustawić na 10
- w procedurze do zdarzenie ONTIMER dla TIMERa wpisać instrukcje w ramce obok

```
kwadrat(x,y,bok,clBtnFace);
x:=x+dx;
y:=y+dy;
kwadrat(x,y,bok,clRed);
if (x+bok > 640) or (x < 0) then dx:=-dx;
if (y+bok > 480) or (y < 0) then dy:=-dy;
```

Można ustawić wysokość formatki na 515 a szerokość na 648 – kwadrat będzie odbijał się od brzegów okienka. Program można przerwać zamykając po prostu okienko. W procedurze Timer1Timer (zdarzenie ONTIMER) najpierw jest wymazywany poprzednio narysowany kwadrat, a następnie rysowany nowy – po obliczeniu współrzędnych, gdyż wszystko zachodziło wewnątrz pętli REPEAT, po wstrzymaniu wyświetlania instrukcją SLEEP.

PRZYKŁAD LEKCJA073 – „uciekający” przycisk

- nowa aplikacja
- przycisk o wymiarach 50x50
- w procedurze do zdarzenia ONMOUSEMOVE dla przycisku wpisać program z ramki
- w procedurze do zdarzenia ONCLICK dla przycisku wpisać instrukcję CLOSE

```
var kx,ky:integer;
begin
  kx:=random(3)-1;
  ky:=random(3)-1;
  Button1.Left:=Button1.Left+10*kx;
  Button1.Top:=Button1.Top+10*ky;
end;
```

Gdy myszka znajdzie się nad przyciskiem, losujemy kierunki przesuwania z przedziału -1..1 i odpowiednio przesuwamy przycisk o 10 pikseli. Jeśli uda się kliknąć na przycisku – program się zakończy – Powodzenia.

Ćwiczenia (na podstawie programu LEKCJA071)

- Zmień program w ten sposób, aby kwadrat poruszał się nie po całym ekranie, ale w prostokącie, którego lewy górny róg znajduje się w punkcie 100,100 a prawy dolny w punkcie 500,400.
- Zmień program tak, aby po każdym odbiciu od ściany kwadrat zmieniał kolor na inny, może być losowo wybrany.
- Zmień program tak, aby po ekranie poruszał się prostokąt, trójkąt, twoje inicjały.
- Zmień program tak, aby po 10 odbiciach od ścianki dolnej program się automatycznie zakończył

SPRAWDZIAN