## LEKCJA 3 – ZMIENNE

```
procedure nazwa procedury
        (parametry);

const
  Nazwisko = 'Jan Nowak';
  Wiek = '30';
  Waga = '75.5';
var
  Nazwisko: string;
  Wiek: integer;
  Waga: real;

begin
  …
end;
```

Posługiwanie się liczbami zwłaszcza dużymi i jeśli trzeba wykonywać na nich obliczenia jest niewygodne. Lepiej przechowywać je pod określoną nazwą w ZMIENNYCH lub STAŁYCH. Jak sama nazwa wskazuje, wartości przechowywane w zmiennych mogą się zmieniać, a w stałych nie mogą – w trakcie działania programu.

**Deklarowanie zmiennych i typy zmiennych**

Zmienne trzeba ZADEKLAROWAĆ, tzn. nadać nazwę i opisać TYP. Typ określa, jakie wartości można PRZYPISAĆ do zmiennej, czyli co może być w niej przechowywane.

Podstawowymi i najczęściej używanymi typami są:

- INTEGER – liczby całkowite (-32767..32766),
- REAL – liczby rzeczywiste,
- STRING – łańcuchy tekstu, np. ‘Ala ma kota’,
- CHAR – pojedyncze znaki tekstu, np. ‘a’, ‘A’, ‘1’, ‘ ‘.

Jeśli zależy nam rzeczywiście na tym, aby coś (przez przypadek) nie zmieniło się w programie – zdefiniujmy to za pomocą stałej. We wszystkich innych przypadkach korzystamy ze zmiennych.

**Nazwy stałych i zmiennych:**

- nie mogą zawierać polskich znaków z „ogonkami” (diakrytycznych)
- można pisać dużymi lub małymi literami (nierozróżnialne)
- muszą być pojedynczymi wyrazami; stosuje się konstrukcje typu: BokFigury lub bok_figury
- nazwy powinny kojarzyć się z przechowywaną w nich wartością – łatwiej czytać program

**Zmienne lokalne i globalne**

Stałe i zmienne deklarujemy po słowie VAR, pod nagłówkiem procedury, a przed słowem BEGIN – oznaczającym początek części głównej procedury i są to tzw. **ZMIENNE LOKALNE**. Zmienne lokalne istnieją tylko w trakcie, gdy wykonywana jest ta procedura. Jeśli program wykonuje procedurę, „nie widać” zmiennych zadeklarowanych w innych procedurach, nie można z nich korzystać, nawet, jeśli w innej procedurze będą zadeklarowane identyczne zmienne, o identycznych nazwach.

Zmienne można również deklarować (po słowie VAR) przed słowem IMPLEMENTATION i są to tzw. **ZMIENNE GLOBALNE** – „widzą je” wtedy **wszystkie procedury**.

Cechą dobrego programisty jest tak napisać program, aby było zadeklarowane jak najmniej zmiennych globalnych (program jest mniej podatny na błędy), choć oczywiście nie da się zawsze uniknąć takiej sytuacji.

**Operacja przypisania**

Stała po zadeklarowaniu ma przypisaną konkretną wartość, niezmienną w trakcie działania procedury. Do zmiennej przypisujemy wartość (nadajemy zmiennej wartość) w części wykonawczej procedury – pomiędzy słowami begin i end. Do przypisywania wartości służy specjalny znak := (dwukropek i znak równości pisany razem).

```
x := 3;                 {X zawiera liczbę 3}
y := (2 - 3) * 3;       {Y zawiera liczbę -3}
x := x + 4;             {X zawiera liczbę 7 - poprzednie X, czyli 3 +4}
x := y;                 (w Y była liczba -1, to X też równe -1)
```

```
const
  w=200;  s=50;  o=20;
var
  x,y:integer;
begin
  x:=100;  y:=100;
```

**PRZYKŁAD - LEKCJA031 - literki**

Schemat liter, jak na poprzedniej lekcji, tym razem wykorzystujemy stałe i zmienne. W przykładzie (górna ramka) zadeklarowano trzy stałe i dwie zmienne. Gdybyśmy mieli zamiar zmieniać w programie wysokość, szerokość lub odległość pomiędzy literami (w, s, o) lepiej od razu zdefiniować je jako zmienne (ramka dolna). Kolejne instrukcje można pisać w jednym wierszu - tracimy jednak na czytelności programu.

```
var x,y,w,s,o:integer;
begin
  x:=100;  y:=100;
  w:=200; s:=50; o:=20;
```

- na zakładce właściwości formatki klikamy w pole ONRESIZE, tworzy się procedura FORMRESIZE, do której wpisujemy deklaracje i instrukcje z okienka na następnej stronie.

**Nowa wartość X**
Zamiast instrukcji **moveto(x+w+o,y+w);** gdzie występuje dodawanie trzech wartości (przy kolejnej literze rachunek skomplikował by się jeszcze bardziej), dołożono instrukcję przypisania **x:=x+s+o;** Wyliczamy nową wartość X – początek kolejnej litery i od tej chwili nie musimy w instrukcjach pisać sumowania, lecz tylko nową wartość X. Przy kolejnej literze możemy postąpić w podobny sposób i przestawić X.

**Wyliczenie połowy odcinka**
W drugiej literze należy narysować linię w połowie wysokości. Rachunek jest prosty: 100+200/2. W związku z tym, że wykonujemy dzielenie (wynikiem jest zawsze liczba rzeczywista, a posługujemy się zmiennymi typu całkowitego), należy dzielić wysokość za pomocą operatora DIV: **y+w div 2** albo **y+(w div 2)**.

```
const
  w=200;  s=50;  o=20;
var
  x,y:integer;
begin
  x:=100;  y:=100;
  with canvas do
  begin
    moveto(x,y);
    lineto(x,y+w);
    lineto(x+s,y+w);
    lineto(x+s,y);
    lineto(x,y);

    moveto(x+s+o,y+w);
    lineto(x+s+o,y);
    lineto(x+s+o+s,y);
    lineto(x+s+o+s,y+w);
    lineto(x+s+o+s,y+w div 2);
    lineto(x+s+o,y+w div 2);
…
```

```
const
  w=200;  s=50;  o=20;
var
  x,y:integer;
begin
  x:=100;  y:=100;
  with canvas do
  begin
    moveto(x,y);
    lineto(x,y+w);
    lineto(x+s,y+w);
    lineto(x+s,y);
    lineto(x,y);

    x:=x+s+o;
    moveto(x,y+w);
    lineto(x,y);
    lineto(x+s,y);
    lineto(x+s,y+w);
    lineto(x+s,y+w div 2);
    lineto(x,y+w div 2);
…
```

Równie skutecznym sposobem byłoby zdefiniowanie nowej zmiennej, np. o nazwie PW (połowa wysokości) i wyliczenie jej na początku programu, ale po instrukcji w:=200; Rysowanie od połowy litery byłoby realizowane w następujący sposób: **lineto(x+s,y+pw);**

```
w:=200;
pw:=w div 2;
```

Pierwsza instrukcja **lineto(x+s,y+w div 2);** rysuje odcinek na narysowanym wcześniej – tak też można, choć nic nie stoi na przeszkodzie by wykonać to za pomocą przesunięcia **moveto(x+s,y+w div 2);**

UWAGA – jeśli poprawnie zostały zdefiniowane wszystkie instrukcje, wystarczy zmienić zawartość zmiennych w,s,o,x,y, aby litery rysowały się w innym miejscu i miały inną wysokość, szerokość lub odstęp między nimi – to jest też jedna z zalet stosowania zmiennych i stałych

**PRZYKŁAD - LEKCJA 32 - przyciski**
Korzystamy z poprzedniego programu LEKCJA031. Literki będą rysowane po naciśnięciu przycisku RYSUJ, a nie jak do tej pory automatycznie po starcie.
- na formatce narysuj kolejny przycisk BUTTON
- zmień nazwę przycisku na RYSUJ (pole CAPTION na zakładce właściwości)
- kliknij podwójnie w przycisk RYSUJ (nowa procedura BUTTON2CLICK) można też kliknąć w pole ONCLICK w zakładce zdarzeń przycisku
- przenieś kod źródłowy z procedury FORMRESIZE do procedury BUTTON2CLICK

Nowy przycisk CZYŚĆ – do wymazywania rysunku
- na formatce narysuj kolejny przycisk BUTTON
- zmień nazwę przycisku na CZYŚĆ
- kliknij podwójnie w przycisk
- dopisz do procedury BUTTON3CLICK instrukcję REFRESH – oczyszczenie formatki

Rysowanie literek po kliknięciu myszą w obszar formatki
- wybierz obiekt FORM1
- w zakładce zdarzeń (EVENTS) wybierz pole ONCLICK
- za pomocą trójkącika wybierz gotową procedurę BUTTON2CLICK – rysowanie liter

**SPRAWDZIAN**
Zadeklarować zmienne i stałe (lub same zmienne).
Narysować 6 liter stosując tylko zadeklarowane zmienne (dane: lewy górny róg, wysokość, szerokość, odstęp).
Przypisać rysowanie do zdarzenia ONRESIZE. Czas 25 minut. Jedna litera = jeden stopień oceny.