

TABLICE

1. Deklaracja tablicy.



Tablica to struktura danych zawierająca wiele elementów tego samego typu identyfikowanych za pomocą indeksu.

Tablicę tworzymy za pomocą słowa kluczowego **ARRAY** (z ang. tablica).

Deklaracja tablicy w języku Pascal ma postać:

var zmienna: array [1..n] of typ elementów tablicy;

gdzie:

zmienna – oznacza nazwę zmiennej tablicowej,

n – to rozmiar tablicy, przy czym liczba elementów tablicy musi być określona wcześniej, np. wpisana w deklaracji tablicy jako konkretna wartość lub podana jako stała (const n=20).

2. Tablica jednowymiarowa i dwuwymiarowa.

Rozróżniamy tablice:

▪ Jednowymiarowe

Najprostsza tablicą jednowymiarową jest ciąg liczb czyli tzw. wektor.

| | | | | |
|-------|-------|-------|-----|-------|
| X_1 | X_2 | X_3 | ... | X_n |
|-------|-------|-------|-----|-------|

Obraz graficzny tablicy jednowymiarowej

Przykład w Pascalu:

Deklaracja tablicy jednowymiarowej składającej się z 20 liczb rzeczywistych.

```
var tab: array [1..20] of real;
```

▪ Dwuwymiarowe

Tablice dwuwymiarowe zwane są często **macierzami**.

| | | | |
|----|-----|-----|----|
| 31 | 22 | 17 | 43 |
| 12 | 78 | 30 | 1 |
| 52 | 132 | 214 | 88 |

Obraz graficzny tablicy dwuwymiarowej

Deklaracja tablicy dwuwymiarowej w języku Pascal ma postać:

var zmienna: array [1..n, 1..m] of typ elementów tablicy;

gdzie:

zmienna – oznacza nazwę zmiennej tablicowej,

n – liczba wierszy tablicy,

m – liczba kolumn tablicy.

Przykład w Pascalu:

Deklaracja tablicy dwuwymiarowej składającej się z 5 wierszy i 7 kolumn z elementami w postaci liczb rzeczywistych (tablica ta składa się z 35 elementów).

```
var dane: array [1..5, 1..7] of real;
```

3. Rozmiar tablicy – indeksy

W nawiasach kwadratowych zapisane są graniczne wartości indeksów w postaci *od..do* (z dwiema kropkami). Indeks oznacza liczbę elementów. Zawiera on kres dolny i górny, czyli zakres tablicy.



- **Rozmiar tablicy musi być ustalony w chwili jej deklaracji i nie może być zmieniony w czasie realizacji programu.**
- Indeksy nie muszą być numerowane od 1, mogą zaczynać się np. od 5 (var tab: array [5..20] of real);
- **Indeksy w deklaracji tablicy muszą być stałymi typu porządkowego (całkowity – integer, word, byte; znakowy – char; logiczny – boolean). Indeks nie może być typu rzeczywistego (real).**

Przykłady deklaracji tablic:

| Dobrze | Źle |
|--|--|
| Var wpisaneLiczby: array [1..200] of Real; Temperatura: array [1..31] of real; Macierz: array [1..3, 1..3] of byte; NazwyDniTygodnia: array [1..7] of string; LicznikZnakow: array ['a'..'z'] of integer; | var Liczby: array [1.5..200.6] of Real; |

4. Odwoływanie się do elementów tablicy

Do elementów tablicy odwołujemy się poprzez zmienną tablicową i indeks.

Przykłady:

1. Deklaracja tablicy składającej się z 20 liczb rzeczywistych.
var a: array [1..20] of real;
Do elementów tej tablicy odwołujemy się poprzez: a[1], a[2], ... a[20].
2. Deklaracja tablicy składającej się z 15 znaków.
var znak: array [0..14] of char;
Do elementów tej tablicy odwołujemy się poprzez: znak[0], znak[1], ... znak[14].
3. Deklaracja tablicy dwuwymiarowej o 12 wierszach i 15 kolumnach.
var tab: array [1..12, 1..15] of integer;
Do elementów tej tablicy odwołujemy się poprzez zmienne z dwoma indeksami tab[i,j]: tab[1,1], tab[1,2], ... tab[5,1], tab[5,2], ... tab [12,15]..

Przykład graficzny:

Na rysunku poniżej przedstawiono dwie tabele (tabela A i B). Aby odwołać się do konkretnego elementu tablicy musimy podać nazwę tablicy (tzw. zmienną tablicową) oraz indeksy informujące o pozycji na której dany element jest w tablicy zapisany.

| A | | | | | B | | | | |
|---|----|----|----|----|---|----|-----|-----|----|
| 1 | 2 | 3 | 4 | 5 | 1 | 2 | 3 | 4 | |
| 5 | 82 | 10 | 17 | 23 | 1 | 31 | 20 | 17 | 43 |
| | | | | | 2 | 12 | 78 | 30 | 1 |
| | | | | | 3 | 52 | 132 | 214 | 98 |

Np. A[1] oznacza element, który ma wartość 5,
A[3] – element o wartości 10,
B[1,2] – element o wartości 20,
B[2,3] – element o wartości 30 itd.

5. Nadawanie wartości elementom tablicy

Nadawanie wartości elementom tablicy odbywa się poprzez przypisanie im pewnych wartości, np.

Liczby[3,5]:=25; {przypisanie elementowi tablicy Liczby znajdującemu się na przecięciu 3 wiersza i 5 kolumny wartości 25}

Nadawanie wartości elementom tablicy może również odbywać się poprzez wczytywanie tych wartości z klawiatury, za pomocą instrukcji powtórzeń – **FOR**.

PROGRAMY - PRZYKŁADY

CW 1 a (bez procedur). Napisz program realizujący wprowadzanie liczb całkowitych do tablicy jednowymiarowej o rozmiarze max 20 elementów.

```
program Tablica;
uses crt;
var tab: array [1..20] of integer; {deklaracja tablicy jednowymiarowej o rozmiarze 20
elementow}
var i : integer; {i – zmienna wykorzystywana jako licznik w pętli FOR oraz jako
indeks tablicy}

begin
  clrscr;
  writeln ('Podaj 20 elementow tablicy');
  for i:=1 to 20 do
  begin
    write ('Podaj ',i , ' element tablicy: ');
    readln (tab[i]); {wczytywanie elementow do tablicy}
```

```

end;

writeln ('Wyprowadzanie elementow tablicy na ekran');
for i:=1 to 20 do
  begin
    write (tab[i], ' '); {wyswietlanie elementow tablicy na ekranie monitora}
  end;
readln;
end.

```

CW 1 b (z procedurami). Napisz program realizujący wprowadzanie liczb całkowitych do tablicy jednowymiarowej o rozmiarze max 20 elementów. Do wprowadzania i drukowania danych z tablicy zastosuj procedury.

```

program Tablica;
uses crt;
var tab: array [1..20] of integer; {deklaracja tablicy jednowymiarowej o rozmiarze 20 elementow}
var i : integer; {deklaracja licznika petli}

procedure Czytaj; {deklaracja procedury, która odpowiada za wczytywanie liczb do tablicy}
begin
  writeln ('Podaj 20 elementow tablicy');
  for i:=1 to 20 do
    begin
      writeln ('Podaj ',i , ' element tablicy');
      readln (tab[i]); {wczytywanie elementow do tablicy}
    end;
  end;
end;
{*****}
procedure Wyswietl; {deklaracja procedury, która wyprowadza dane wczytane do tablicy na ekran monitora}
begin
  writeln ('Wyprowadzanie elementow tablicy na ekran');
  for i:=1 to 20 do
    begin
      write (tab[i], ' '); {wyswietlanie elementow tablicy na ekranie monitora}
    end;
  end;
end;
{*****}

begin {tu zaczyna się program główny}
  clrscr;
  Czytaj; {wywołanie procedury Czytaj, która umożliwia wczytanie liczb do tablicy}
  Wyswietl; {wywołanie procedury Wyswietl, która umożliwia wyprowadzenie na ekran elementow tablicy}
  readln;
end. {koniec programu}

```



CW 2 Napisz program realizujący wprowadzanie liczb całkowitych do tablicy jednowymiarowej o 10 elementach, a następnie wydrukowanie jej zawartości w odwrotnej kolejności. Do wprowadzania i drukowania danych z tablicy zastosuj procedury.

```
program Tablica;
uses crt;
var tab: array [1..10] of integer; {deklaracja tablicy jednowymiarowej o rozmiarze r_max}
var i : integer;

procedure Czytaj;
begin
  writeln ('Podaj 10 elementow tablicy ');
  for i:=1 to 10 do
    begin
      writeln ('Podaj ',i , ' element tablicy');
      readln (tab[i]); {wczytywanie elementow do tablicy}
    end;
end;
{*****}
procedure Wyświetl_odwrotnie;
begin
  writeln;
  writeln ('Wyprowadzanie elementow tablicy w odwrotnej kolejności');
  for i:=10 downto 1 do
    begin
      write (tab[i], ' '); {wyswietlanie elementow tablicy na ekranie monitora}
    end;
end;
{*****}

begin
  clrscr;
  Czytaj;
  Wyświetl_odwrotnie;
  readln;
end.
```



Funkcja losowa RANDOM

Do wypełniania tablicy przypadkowymi wartościami służy funkcja **Random**.



- Przed pierwszym użyciem funkcji Random, należy koniecznie wywołać procedurę **Randomize**.
- W przypadku podania argumentu **Zakres** wynikiem działania funkcji będzie liczba całkowita z przedziału **0..Zakres -1**. Jeżeli argument ten został pominięty, wynikiem będzie liczba rzeczywista z przedziału **0..1**. (patrz przykład poniżej).

Przykładowy zapis funkcji RANDOM:

```
randomize;  
x:=random (6); {pod zmienną x podstaw losowo wybraną liczbę z przedziału 0..5}
```

CW 3 Napisz program, który zapełni tablice dwudziestoma przypadkowymi liczbami z zakresu od 1 do 100.

```
program Tablica;  
uses crt;  
var tab: array [1..20] of integer; {deklaracja tablicy jednowymiarowej o rozmiarze 20  
elementow}  
var i: integer;  
  
procedure Losowanie;  
  
begin  
    randomize; {zainicjalizuj generator liczb pseudolosowych}  
    for i:=1 to 20 do  
        begin  
            tab[i]:=random (101);  
        end;  
end;  
{*****}  
procedure Wyświetl;  
  
begin  
    write(' Elementy tablicy          : ');  
    for i:=1 to 20 do  
        begin  
            write (tab[i], ' '); {wyswietlanie elementow tablicy na ekranie monitora}  
        end;  
end;  
{*****}  
  
begin  
    clrscr;  
    Losowanie;  
    Wyświetl;
```

```
Maksimum_i_Minimum;  
readln;  
end.
```



CW 4 Napisz program, który zapełni tablicę jednowymiarową o 10 elementach, przypadkowymi liczbami z zakresu od 1 do 100, a następnie wyświetli największą i najmniejszą z tych liczb.

Uwaga!

Algorytm wyboru minimum (lub maksimum) z n liczb.

Opis algorytmu:

- Wskazujemy dowolny element jako najmniejszy (w programie wskazałem element pierwszy tablicy czyli *tab[1]*), i porównujemy z drugim. Jeśli drugi jest mniejszy, to teraz on jest traktowany jako minimum i porównuję go z trzecim. W przeciwnym wypadku porównuje element aktualnie najmniejszy z trzecim itd. – aż do końca ciągu elementów.
- Bieżąca wartość minimum przechowywana jest w zmiennej *min*.
- Powtarzana jest w pętli operacja porównania i podstawiania.

```
program Tablica;
```

```
uses crt;
```

```
var tab: array [1..10] of integer; {deklaracja tablicy jednowymiarowej o rozmiarze  
max 10 elementow}
```

```
var i: integer;
```

```
procedure Losowanie;
```

```
begin
```

```
  randomize;
```

```
  for i:=1 to 10 do
```

```
    begin
```

```
      tab[i]:=random (101);
```

```
    end;
```

```
end;
```

```
{*****}
```

```
procedure Wyświetl;
```

```
begin
```

```
  write(' Elementy tablicy           : ');
```

```
  for i:=1 to 10 do
```

```
    begin
```

```
      write (tab[i], ' '); {wyswietlanie elementow tablicy na ekranie monitora}
```

```
    end;
```

```
end;
```

```
{*****}
```

```
procedure Maksimum_i_minimum;
```

```
var max,min: integer; {min – wartosc elementu najmniejszego, max – wartosc  
elementu największego}
```

```

begin
  max:=tab[1];
  min:=tab[1];
  writeln;
  for i:=1 to 10 do
    begin
      if tab[i]>max then
        max:=tab[i];
      end;
      writeln(' Największy elementy tablicy      : ',max);
    } Wyszukaj maximum

  for i:=1 to 10 do
    begin
      if tab[i]<min then
        min:=tab[i];
      end;
      writeln(' Najmniejszy elementy tablicy    : ',min);
    } Wyszukaj minimum

end;
{*****}
begin
  clrscr;
  Losowanie;
  Wyświetl;
  Maksimum_i_Minimum;
  readln;
end.

```



CW 5. Napisz program realizujący wprowadzanie liczb całkowitych do tablicy jednowymiarowej o n elementach. W programie zdefiniuj procedurę bez parametrów, która będzie obliczała sumę elementów tablicy oraz średnią arytmetyczną.

Uwaga!

Aby rozpocząć działania na elementach tablicy, w tym przypadku aby zsumować je, musimy najpierw te elementy wczytać do tablicy. Stąd w każdym programie pojawia się procedura *Czytaj*.

```

program Tablica;
uses crt;
var tab: array [1..10] of integer; {deklaracja tablicy jednowymiarowej o rozmiarze
max 10 elementow}
var n: integer; {n – rozmiar tablicy, podawany w programie; n<=10}
var i: integer;

procedure Czytaj;

begin
  writeln ('Podaj liczbe elementow tablicy (n<=10)');
  readln (n);

```

```

for i:=1 to n do
  begin
    write ('Podaj ',i , ' element tablicy ');
    readln (tab[i]); {wczytywanie elementow do tablicy}
  end;
end;
{*****}
procedure Wyświetl;
begin
  write(' Elementy tablicy          : ');
  for i:=1 to n do
    begin
      write (tab[i], ' '); {wyswietlanie elementow tablicy na ekranie monitora}
    end;
end;
{*****}
procedure Suma_I_Srednia;
var suma: integer; {wartosc sumy wpisywanych liczb}
var srednia: real; {wartosc sredniej arytmetycznej}

begin
  suma:=0; {pamietaj o wyzerowaniu sumy przed petla}
  writeln;
  for i:=1 to n do
    begin
      suma:=suma+tab[i];
    end;
  srednia:=suma/n; {oblicz srednia}
  writeln(' Suma elementow tablicy wynosi      : ',suma);
  write(' Srednia arytmetyczna wynosi      : ',srednia:5:2);
end;
{*****}
begin
  clrscr;
  Czytaj;
  Wyświetl;
  Suma_I_Srednia;
  readln;
end.

```



CW 6 Napisz program realizujący wprowadzanie liczb całkowitych do tablicy jednowymiarowej o 10 elementach. W programie zdefiniuj procedurę bez parametrów, która będzie wyświetlała na ekranie monitora elementy parzyste i nieparzyste tablicy.

Uwaga!

Operator **mod** podaje resztę z dzielenia liczb całkowitych. Można go również wykorzystać aby sprawdzić, czy podana liczba jest parzysta lub nie.

Podana liczba jest parzysta gdy jest ona podzielna przez 2, a więc gdy reszta z dzielenia przez 2 będzie równa zero.

```
program Tablica;
uses crt;
var tab: array [1..10] of integer; {deklaracja tablicy jednowymiarowej o rozmiarze
max 10 elementow}
var i: integer;

procedure Czytaj;
begin
  writeln ('Podaj 10 elementow tablicy');
  for i:=1 to 10 do
    begin
      write ('Podaj ',i , ' element tablicy ');
      readln (tab[i]); {wczytywanie elementow do tablicy}
    end;
end;
{*****}
procedure Wyswietl;
begin
  write(' Elementy tablicy          : ');
  for i:=1 to 10 do
    begin
      write (tab[i], ' '); {wyswietlanie elementow tablicy na ekranie monitora}
    end;
end;
{*****}
procedure Parzyste_I_Nieparzyste;
begin
  writeln;
  write(' Elementy parzyste tablicy to      : ');
  for i:=1 to 10 do
    begin
      if (tab[i] mod 2)=0 then {sprawdz czy podana liczba jest podzielna przez 2}
        write(tab[i], ' ');
    end;
  writeln;
  write(' Elementy nieparzyste tablicy to    : ');
  for i:=1 to 10 do
    begin
      if (tab[i] mod 2)<>0 then {sprawdz czy podana liczba nie jest podzielna przez
        2}
        write(tab[i], ' ');
    end;
end;
{*****}
begin
  clrscr;
  Czytaj;
  Wyswietl;
```

```

    Parzyste_I_Nieparzyste;
    readln;
end.

```



CW 7 Napisz program realizujący wprowadzanie liczb całkowitych do tablicy jednowymiarowej o 5 elementach. W programie zdefiniuj procedurę bez parametrów, która będzie wyświetlała na ekranie monitora elementy dodatnie, ujemne.

```

program Tablica;
uses crt;
var tab: array [1..5] of integer; {deklaracja tablicy jednowymiarowej o rozmiarze max
5 elementow}
var i: integer;

procedure Czytaj;
begin
    writeln ('Podaj 5 elementow tablicy');
    for i:=1 to 5 do
        begin
            write ('Podaj ',i , ' element tablicy ');
            readln (tab[i]); {wczytywanie elementow do tablicy}
        end;
    end;
end;
{*****}
procedure Wyszwietl;
begin
    write(' Elementy tablicy          : ');
    for i:=1 to 5 do
        begin
            write (tab[i], ' '); {wyszwietlanie elementow tablicy na ekranie monitora}
        end;
    end;
end;
{*****}
procedure Operacje;
begin
    writeln;
    write(' Elementy tablicy dodatnie      : ');
    for i:=1 to 5 do
        begin
            if tab[i] > 0 then
                write(tab[i], ' ');
        end;
    end;

    writeln;
    write(' Elementy tablicy ujemne        : ');
    for i:=1 to 5 do
        begin
            if tab[i]< 0 then

```

```

    write(tab[i], ' ');
end;

end;
{*****}
begin
  clrscr;
  Czytaj;
  Wswietl;
  Operacje;
  readln;
end.

```



CW 8 Napisz program, który zapełni tablicę jednowymiarową dziesięcioma przypadkowymi liczbami z zakresu od 1 do 20, a następnie uporządkuje je rosnąco i malejąco.

Uwagi!

- Porządkowanie tablicy nosi nazwę **sortowania**. Możemy stosować różne algorytmy sortowania danych. W programie zostanie użyta metoda sortowania bąbelkowego.

Sortowanie - Ustawianie danych lub informacji według określonego kryterium, zwykle w kolejności rosnącej (tj. od najmniejszego elementu do największego) lub malejącej (odwrotnie). Istnieje wiele sposobów (algorytmów) sortowania, z których najpopularniejsze są:

- sortowanie przez wybór,
- sortowanie bąbelkowe,
- sortowanie kulekowe,
- sortowanie przez wstawianie.



Sortowanie bąbelkowe polega na porównywaniu parami kolejnych liczb i przestawianiu ich, jeśli występują w niewłaściwej kolejności.

Opis algorytmu sortowania bąbelkowego

W poniższym programie sortowanie odbywać się będzie w następujący sposób: w pętli przeszukiwać będziemy tablicę i sprawdzać, czy bieżący element ($tab[i]$) nie jest większy od następnego ($tab[i+1]$). Jeżeli tak to zamienimy je ze sobą. Wykorzystamy przy tym zmienną pomocniczą (pom – wcześniej ją zadeklaruj), która będzie służyć do zapamiętania elementu, który chcemy zmienić miejscami. Operacje porównywania i zamiany par liczb będziemy wykonywać, aż do chwili, gdy okaże się, że nie trzeba już wykonywać żadnych zamian.



Ciekawe strony – tu możesz poczytać o innych algorytmach sortowania:
<http://www.i-lo.tarnow.pl/edu/inf/alg/algsort/index.html>

```

program Tablica;
uses crt;
const n=10; {okresla rozmiar tablicy}
var tab: array [1..n] of byte; {deklaracja tablicy jednowymiarowej o rozmiarze 10
elementow}
var pom: byte; {zmienna pomocnicza, która przechowuje chwilowo wartości
elementow tablicy, które będą ulegac porządkowaniu - sortowaniu}
var i,j: integer;

procedure Losowanie;
begin
  randomize;
  for i:=1 to n do
    begin
      tab[i]:=random (21);
    end;
end;
{*****}
procedure Wyszwietl;
begin
  write(' Elementy tablicy           : ');
  for i:=1 to n do
    begin
      write (tab[i], ' '); {wyszwietlanie elementow tablicy na ekranie monitora}
    end;
end;
{*****}
procedure Sortowanie;
begin
  writeln;
  for j:=1 to n do    {przeszukiwanie tablicy nieuporządkowanej – całości, gdzie
n – rozmiar tablicy}
    begin
      for i:=1 to n-1 do    {porównujemy zawsze dwa sąsiednie elementy, ale pod
koniec interesuje nas tylko indeks przedostatni stąd n-1}
        begin
          if tab[i]>tab[i+1] then    {jeżeli porządek niewłaściwy, zamień liczby
miejscami}
            begin
              pom:=tab[i]; {pom - zmienna pomocnicza, która przechowuje
chwilowo wartości elementów tablicy, które będą ulegać
porządkowaniu - sortowaniu}
              tab[i]:=tab[i+1];
              tab[i+1]:=pom;
            end;
          end;
        end;
      end;
    end;
end;
{*****}

```

```

procedure Odwrotnie;
begin
  writeln ('Tablica po sortowaniu malejaco');
  for i:=n downto 1 do
    begin
      write (tab[i], ' '); {wyswietlanie elementow tablicy na ekranie monitora}
    end;
end;
{*****}

```

```

begin
  clrscr;
  Losowanie;
  Wyswietl;
  Sortowanie;
  writeln;
  writeln ('Tablica po sorotwaniu rosnaco');
  Wyswietl;
  Odwrotnie;
  readln;
end.

```



OPERACJE NA TABLICACH

DODAWANIE DWÓCH TABLIC JEDNOWYMIAROWYCH

CW 9 Napisz program, który będzie dodawał elementy tablicy jednowymiarowej 10 – elementowej i drugiej tablicy jednowymiarowej też 10 - elementowej.

```

program dodawanie_tablic_jednowymiarowych;
uses crt;
const n=10; {okresla rozmiar tablicy}
var tab1: array [1..n] of integer;
var tab2: array [1..n] of integer;
var tab3: array [1..n] of integer;
var i: integer;

procedure Wprowadzanie;
begin
  writeln('Wprowadzanie elementow do tablicy 1 ');
  for i:=1 to n do
    begin
      write('Podaj ',i,' element pierwszej tablicy : ');
      readln(tab1[i]);
    end;
  writeln;

  writeln('Wprowadzanie elementow do tablicy 2 ');

```

```

    for i:=1 to n do
        begin
            write('Podaj ',i,' element drugiej tablicy : ');
            readln(tab2[i]);
        end;
    end;
{*****}
procedure Wyswietl;
begin
    write(' Elementy tablicy pierwszej          : ');
    for i:=1 to n do
        begin
            write (tab1[i], ' '); {wyswietlanie elementow tablicy na ekranie monitora}
        end;
    writeln;
    write(' Elementy tablicy drugiej          : ');
    for i:=1 to n do
        begin
            write (tab2[i], ' '); {wyswietlanie elementow tablicy na ekranie monitora}
        end;
    end;
{*****}
procedure Dodaj;
begin
    for i:=1 to n do
        begin
            tab3[i]:=tab1[i]+tab2[i];
        end;
    writeln;
    writeln;
    write(' **** Dodawanie dwoch tablic ****          ');
    writeln;
    write(' Elementy tablicy po dodaniu          : ');
    for i:=1 to n do
        begin
            write (tab3[i], ' '); {wyswietlanie elementow tablicy na ekranie monitora}
        end;
    end;
{*****}

begin
    clrscr;
    Wprowadzanie;
    Wyswietl;
    Dodaj;
    readln;
end.

```



TABLICE DWUWYMIAROWE – PROGRAMY PRZYKŁADY

W przypadku tablicy dwuwymiarowej potrzebne nam są dwie pętle, jedna do wierszy, a druga – do kolumn. Pętla należy umieścić jedną w drugiej czyli zagnieździć. Pętle muszą też być sterowane różnymi licznikami (np. i pierwsza pętla, a druga j).

CW 10 Napisz program realizujący wprowadzanie liczb całkowitych do tablicy o wymiarach 3×3 oraz wydrukowanie jej zawartości. Do wprowadzania i drukowania danych z tablicy zastosuj procedury.

```
program Tablica_dwuwymiarowa;
uses crt;
var tab: array [1..3, 1..3] of integer; {deklaracja tablicy  $3 \times 3$  – 9 elementowa tablica}
var i,j : integer; {i,j - zmienne wykorzystywane w petli FOR jako licznik oraz jako indeks w tablicy odpowiednio i – wiersze, j - kolumny}

procedure Czytaj;
begin
  writeln ('Wprowadzanie danych do tablicy');
  for i:=1 to 3 do {wczytywanie elementow do tablicy w kolejności wiersz po wierszu}
    begin
      for j:=1 to 3 do
        begin
          writeln ('Podaj element na przecieciu ',i,' wiersza i ',j,' kolumny');
          readln (tab[i,j]);
        end;
      end;
    end;
end;
{*****}
procedure Wyszwietl;
begin
  writeln;
  writeln ('Odczytywanie elementow z tablicy');
  for i:=1 to 3 do
    begin
      for j:=1 to 3 do
        begin
          write('w=',i, ' k=',j, ' tab['',i,'',j,'] = ');
          writeln (tab[i,j]);
        end;
      end;
    end;
end;
{*****}
begin
  clrscr;
  Czytaj;
  Wyszwietl;
```

```
readln;  
end.
```



CW 11 Napisz program realizujący wprowadzanie liczb całkowitych do tablicy o wymiarach $m \times n$ (m – liczba wierszy, n – liczba kolumn) oraz wydrukowanie jej zawartości. Przyjmij dla tablicy następujące rozmiary: maksymalna liczba wierszy 10 i maksymalna liczba kolumn 10. Do wprowadzania i drukowania danych z tablicy zastosuj procedury.

```
program Tablica_dwuwymiarowa;  
uses crt;  
const w_max=10; {maksymalna liczba wierszy}  
const k_max=10; {maksymalna liczba kolumn}  
var tab: array [1..w_max, 1..k_max] of integer;  
var m,n: integer; {m,n – zmienne wykorzystywane do wprowadzenia liczby wierszy i kolumn;  $m$  i  $n \leq 10$ ; wartości wprowadzane sa w programie}  
var i,j : integer;
```

```
procedure Czytaj;  
begin
```

```
  writeln ('Wprowadzanie danych do tablicy');  
  writeln ('Podaj liczbe wierszy (m<=10) :');  
  readln (m);  
  writeln ('Podaj liczbe kolumn (n<=10) :');  
  readln (n);
```

```
  for i:=1 to m do
```

```
    begin
```

```
      for j:=1 to n do
```

```
        begin
```

```
          writeln ('Podaj element na przecieciu ',i,' wiersza i ',j,' kolumny');  
          readln (tab[i,j]);
```

```
        end;
```

```
      end;
```

```
    writeln;
```

```
end;
```

```
{*****}
```

```
procedure Wyszwietl;
```

```
begin
```

```
  writeln;
```

```
  writeln ('Odczytywanie elementow z tablicy - wiersz po wierszu');
```

```
  for i:=1 to m do
```

```
    begin
```

```
      for j:=1 to n do
```

```
        begin
```

```
          write('w=',i, ' k=',j, ' tab['',i,',',j,'] = ');
```

```
          writeln (tab[i,j]);
```

```
        end;
```

```

        end;
    end;
    {*****}
begin
    clrscr;
    Czytaj;
    Wyświetl;
    readln;
end.

```



CW 12 Napisz program, który zapełni tablicę o wymiarach 3×3 przypadkowymi liczbami z zakresu od 1 do 100, a następnie wyświetli największą i najmniejszą z tych liczb.

```

program Tablica_dwuwymiarowa;
uses crt;
var tab: array [1..3, 1..3] of integer; {deklaracja tablicy dwuwymiarowej o rozmiarze
9 elementow}
var max,min: integer;
var i,j: integer;

procedure Czytaj;
begin
    randomize;
    writeln('Wprowadzanie danych do tablicy');

    for i:=1 to 3 do
        begin
            for j:=1 to 3 do
                begin
                    tab[i,j]:=random (101);
                end;
            end;
        end;
end;
    {*****}
procedure Wyświetl;
begin
    writeln;
    writeln('Elementy tablicy : ');
    writeln;
    for i:=1 to 3 do
        begin
            for j:=1 to 3 do
                begin
                    write (tab[i,j], ' ');
                end;
            writeln;
            writeln;
        end;
    end;

```

```

end;
{*****}
procedure Maksimum;
begin
    max:=tab[1,1];
    for i:=1 to 3 do
        begin
            for j:=1 to 3 do
                begin
                    if tab[i,j]>max then
                        max:=tab[i,j];
                end;
            end;
        end;
    writeln ('Element maksymalny tablicy wynosi: ', max);
end;
{*****}
procedure Minimum;
begin
    min:=tab[1,1];
    for i:=1 to 3 do
        begin
            for j:=1 to 3 do
                begin
                    if tab[i,j]<min then
                        min:=tab[i,j];
                end;
            end;
        end;
    writeln ('Element minimalny tablicy wynosi: ', min);
end;
{*****}

begin
    clrscr;
    Czytaj;
    Wyświetl;
    Maksimum;
    Minimum;
    readln;
end.

```



CW 13 Napisz program, sumujący dwie macierze. Elementy obu macierzy mają być generowane przez program.



Dodawanie macierzy jest możliwe tylko dla dwóch macierzy o takich samych wymiarach !!! Wynikiem dodawania macierzy jest macierz o takich samych wymiarach jak składniki. Elementy macierzy wynikowej są sumą odpowiednich elementów składników.

Przykład:

Dodawanie macierzy. Pamiętaj -> Dwie macierze możemy dodać wtedy, gdy są tego samego wymiaru.

Przykład: Niech $A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix}$, $B = \begin{bmatrix} -1 & -2 & 2 & 1 \\ 0 & 1 & -7 & -7 \end{bmatrix}$.

wówczas

$$\begin{aligned} A + B &= \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix} + \begin{bmatrix} -1 & -2 & 2 & 1 \\ 0 & 1 & -7 & -7 \end{bmatrix} = \begin{bmatrix} 1+(-1) & 2+(-2) & 3+2 & 4+1 \\ 5+0 & 6+1 & 7+(-7) & 8+(-7) \end{bmatrix} = \\ &= \begin{bmatrix} 0 & 0 & 5 & 5 \\ 5 & 7 & 0 & 1 \end{bmatrix}. \end{aligned}$$

```
program Tablica_dwuwymiarowa;
uses crt;
var tab1:array[1..5, 1..5] of integer;
var tab2:array[1..5, 1..5] of integer;
var suma:array[1..5, 1..5] of integer;
var i,j: integer;
```

```
procedure wprowadz;
```

```
begin
```

```
  randomize;
```

```
  for i:=1 to 5 do
```

```
    begin
```

```
      for j:=1 to 5 do
```

```
        begin
```

```
          tab1[i,j]:=random(6);
```

```
          tab2[i,j]:=random(6);
```

```
        end;
```

```
      end;
```

```
end;
```

```
{*****}
```

```
procedure wyswietl;
```

```
begin
```

```
  writeln ('Elementy tablicy pierwszej: ');
```

```
  for i:=1 to 5 do
```

```
    begin
```

```
      for j:=1 to 5 do
```

```
        begin
```

```
          write(tab1[i,j], ' ');
```

```
        end;
```

```
      writeln;
```

```
    end;
```

```
  writeln;
```

```
  writeln ('Elementy tablicy drugiej: ');
```

```

for i:=1 to 5 do
begin
for j:=1 to 5 do
begin
write(tab2[i,j], ' ');
end;
writeln;
end;
end;
{*****}
procedure sumowanie;
begin
for i:=1 to 5 do
begin
for j:=1 to 5 do
begin
suma[i,j]:=tab1[i,j]+tab2[i,j];
end;
end;
writeln;
writeln('Tablice po ZSUMOWANIU');
for i:=1 to 5 do
begin
for j:=1 to 5 do
begin
write(suma[i,j], ' ');
end;
writeln;
end;
end;
{*****}
Begin
clrscr;
wprowadz;
wyswietl;
writeln;
sumowanie;
readln;
End.

```



W tablicach można przechowywać dane innych typów, np. teksty.

CW14 Napisz program, który zapełni tablice pięcioma nazwiskami, a następnie je wyświetli.

```

program cw14;
uses crt;

```

```

var nazwiska: array[1..5] of string;
var i: integer;

procedure wprowadz;
begin
  for i:=1 to 5 do
    begin
      writeln ('Podaj ',i, ' nazwisko:');
      readln (nazwiska[i]);
    end;
end;
{*****}
procedure wyswietl;
begin
  writeln;
  writeln ('Tablica z nazwiskami:');
  for i:=1 to 5 do
    begin
      writeln (nazwiska[i]);
    end;
end;
{*****}

begin
  clrscr;
  wprowadz;
  wyswietl;
  readln;
end.

```



6. DZIAŁANIA NA TEKSTACH – praca z danymi tekstowymi

6.1. Deklaracja zmiennej typu łańcuchowego

Do przedstawienia wszelkich napisów, czyli łańcuchów w Pascalu służy typ **STRING**. Łańcuch znaków może być traktowany jako specyficzna tablica, której typem jest char.

ARRAY[0..255] OF CHAR

Deklaracja zmiennych typu tekstowego ma postać:

var zmienna: string[n];

gdzie:

n oznacza długość łańcucha (maksymalnie 255 znaków).

np. *var nazwa: string [15];* {deklaracja łańcucha o długości 15 znaków}

Do całego napisu odwołujemy się przez zmienną *nazwa*, a do pojedynczych znaków – np. przez zmienne: *nazwa[1]*, *nazwa[5]*.

Łańcuch znaków można traktować jako całość i tym różnią się od tablic, do których zawsze odwołujemy się poprzez pojedyncze elementy. Wartością łańcucha jest dowolny ciąg znaków ujęty w apostrofy (').

CW15 Napisz program w wyniku którego dla wprowadzonego znaku „I” program wyprowadza twoje imię, dla „N” – twoje nazwisko, a dla znaku innego niż „I” oraz „N” – komunikat „zły znak”.

```
program cw15;
uses crt;
var wybor: char; {wprowadzany znak I lub N}
var imie: string[20];
var nazwisko: string[25];
begin
  clrscr;
  writeln ('Wybierz I lub N ');
  writeln ('I - wyswietlenie imienia');
  writeln ('N - wyswietlenie nazwiska');
  readln (wybor);

  case wybor of
    'I':begin
      imie:='Jan';
      writeln ('Masz na imie ',imie);
    end;
    'N':begin
      nazwisko:='Kowalski';
      writeln ('Masz na nazwisko ',nazwisko);
    end
  else
    writeln ('Zły znak');
  end;
  readln;
end.
```



6.2. Funkcja LENGTH

Funkcja length podaje (zwraca) długość danego łańcucha. Zapis ogólny funkcji ma postać:

Length (s);

Wynikiem jest długość łańcucha znaków s, czyli liczba przedstawiająca liczbę znaków, z których on się składa.

CW 16 Napisz program, sprawdzający długość wprowadzanego przez użytkownika nazwiska i imienia.

```
program dlugosc_wyrazow;
uses crt;
var imie: string[20];
var nazwisko: string[25];

begin
  clrscr;
  write ('Podaj imie :');
  readln (imie);
  write ('Podaj nazwisko :');
  readln (nazwisko);
  writeln;
  writeln ('Imie ',imie,' zawiera ',length(imie),' znakow');
  writeln ('Nazwisko ',nazwisko,' zawiera ',length(nazwisko),' znakow');
  readln;
end.
```



6.3. Odwoływanie się do elementów łańcucha

Turbo Pascal umożliwia operacje na pojedynczych znakach danego łańcucha tekstowego. Do pojedynczego znaku odwołujemy się podobnie jak w przypadku tablic poprzez nazwę zmiennej i indeks: np. *wyraz[1]*, *wyraz[5]*.

CW 17 Napisz program zamieniający we wprowadzonym tekście znak s na p.

```
program zamiana;
uses crt;
var tekst: string[20];
var i : integer;

begin
  clrscr;
  writeln ('Napisz krotki tekst i naciśnij Enter');
  readln (tekst);
  writeln;
  writeln ('Lancuch przed zamiana: ',tekst);

  for i:=1 to length(tekst) do {length - wyliczenie liczby znakow}
  begin
    if (tekst[i]='s') then {sprawdz czy znak w lancuchu jest litera s}
    begin
      tekst[i]='p'; {jezeli tak to zamien znak s na p}
    end;
  end;
  writeln('Lancuch po zamianie liter: ',tekst);
end;
```

```
readln;  
end.
```



CW18 Napisz program sprawdzający liczbę małych liter „a” w tekście.

```
program zamiana;  
uses crt;  
var tekst: string[20];  
var i, dlugosc, ile : integer; {i - licznik petli oraz indeks w lancuchu tekstowym}  
                                {dlugosc - zmienna, ktora bedzie przechowywac dlugosc lancucha}  
                                {ile - zmienna na ilosc liter a}  
begin  
  clrscr;  
  ile:=0;  
  writeln ('Napisz krotki tekst i naciśnij Enter');  
  readln (tekst);  
  dlugosc:=length(tekst); {wylicz liczbę znaków lancucha i przypisz ją do zmiennej  
                           dlugosc}  
  
  writeln;  
  for i:=1 to dlugosc do  
    begin  
      if (tekst[i]='a') then  
        begin  
          ile:=ile+1;  
        end;  
    end;  
  writeln('Liczba małych liter "a" w tekście: ',ile);  
  readln;  
end.
```

Wskazówki do powyższego programu:

- Zmienna *ile* została zadeklarowana wartością początkową równą zero. Należy tak uczynić gdyż zliczanie małych liter „a” odbywa się w pętli. W ten sposób unikniemy błędów które mogą pojawić się przy pierwszym sumowaniu, gdzie może nastąpić sumowanie z zupełnie przypadkową wartością.

