Piotr Borsuk TP 4.0

WO/WEAIiIB

Sprawozdanie Laboratorium 4

W szczególności bedziemy omawiać pętle for i while, instrukcje warunkowe if oraz wykorzystanie funkcji i wektorów do manipulowania danymi. Dodatkowo, na przykładach omawialiśmy jak korzystać z funkcji wbudowanych w MATLAB, takich jak input() i disp(), oraz jak definiować i wykorzystywać zmienne. Omawialiśmy również przykłady zastosowań tych elementów w praktycznych problemach, takich jak obliczenia matematyczne czy manipulowanie wektorami i macierzami.

Zad 1 Stosując pętle for wypisz liczby od 1 do N, gdzie N=20.

```
N=20;
for i=1:N
    disp('liczba')
i
end
```

```
liczba

i =

     1

liczba

i =

     2

liczba

i =

     3

liczba

i =

     4
```

Kod powyżej ustala wartość N na 20, a następnie wykorzystuje pętlę for do wypisania kolejnych liczb całkowitych od 1 do 20. Każda liczba jest wypisywana w nowej linii za pomocą funkcji disp(). Dla zaoszczędzenia miejsca, pokazuje tylko 4 początkowe rozwiązania.

Zad 2 Stosując pętle for wypisz liczby od N do 1, gdzie N=20.

```
>> for i = 20:-1:1
    disp(i)
end
    20

    19

    18

    17

    16

    15

    14

    13

    12

    11

    10

     9

    (...)

    3

    2

    1
```

Kod powyżej ustala wartość N na 20, a następnie wykorzystuje pętlę for do wypisania kolejnych liczb całkowitych od 20 do 1. W tym celu wykorzystano składnię N:-1:1, która generuje wektor liczb od N do 1 z krokiem -1. Każda liczba jest wypisywana w nowej linii za pomocą funkcji disp(). Tak samo jak w powyższym zadaniu skróciłem rozwiązanie w celu zaoszczędzeniu miejsca.

Zad 3 Stosując pętle for przemnóż dwa wektory X=[1,2,3] oraz t=[1,2,3]', Wyniki zapisz w wektorze A.

```
Zadanie 3
X = [1, 2, 3];   % wektor X
t = [1; 2; 3];   % wektor t
A = zeros(length(X), 1);  % inicjalizacja wektora A jako wektora zerowego o długości równej długości wektora X

for i = 1:length(X)
    A(i) = X(i) * t(i);
end
A
```

Rozwiązanie:

```
A =

     1
     4
     9
```

Zad 4 Wykorzystując 2 pętle for jedna wewnątrz drugiej utwórz macierz A będącą iloczynem numeru kolumny i wiersza. Przyjmij liczbę wierszy i kolumn od 1 do 10. Macierz A powinna przypominać tabliczkę mnożenia o wymiarach 10x10.

```
Zadanie 4
n = 10;     % liczba wierszy i kolumn
A = zeros(n);   % inicjalizacja macierzy A jako macierzy zerowej o wymiarach n x n

for i = 1:n    % iteracja przez wiersze
    for j = 1:n    % iteracja przez kolumny
        A(i,j) = i * j;   % przypisanie wartości iloczynu numeru kolumny i wiersza do odpowiedniej komórki macierzy A
    end
end

disp(A)     % wyświetlenie macierzy A na ekranie
```

```
disp(A)      % wyświetlenie macierzy A na ekranie
     1     2     3     4     5     6     7     8     9    10
     2     4     6     8    10    12    14    16    18    20
     3     6     9    12    15    18    21    24    27    30
     4     8    12    16    20    24    28    32    36    40
     5    10    15    20    25    30    35    40    45    50
     6    12    18    24    30    36    42    48    54    60
     7    14    21    28    35    42    49    56    63    70
     8    16    24    32    40    48    56    64    72    80
     9    18    27    36    45    54    63    72    81    90
    10    20    30    40    50    60    70    80    90   100
```

Zad 5 Mając dany wektor x=[1 15 7 12 -1 0 -15 9 5 2], napisz polecenia wykonujące następujące operacje: a) podmień dodatnie elementy wektora x na zera b) pomnożyć parzyste elementy wektora x przez 100 c) utworzyć wektor y złożony z elementów x pomnożonych przez 20 d) wymienić zerowe elementy na liczbę pseudolosową e) policzyć sumę elementów wektora x wykorzystując pętle. Polecenie sum(x) użyj do sprawdzenia swojego wyniku

```
% dany wektor
x = [1 15 7 12 -1 0 -15 9 5 2];

% a) podmiana dodatnich elementów na zera
x(x > 0) = 0;

% b) pomnożenie parzystych elementów przez 100
x(mod(x, 2) == 0) = x(mod(x, 2) == 0) * 100;

% c) utworzenie wektora y z elementami x pomnożonymi przez 20
y = x * 20;

% d) wymiana zerowych elementów na liczbę pseudolosową
x(x == 0) = randi([-10 10], 1, sum(x == 0));

% e) policzenie sumy elementów wektora x wykorzystując pętlę
suma = 0;
for i = 1:length(x)
    suma = suma + x(i);
end

% wyświetlenie wyników
disp('Wektor x: ');
disp(x);


disp('Wektor y: ');
disp(y);

disp('Suma elementów wektora x obliczona pętlą: ');
disp(suma);

disp('Suma elementów wektora x obliczona funkcją sum: ');
disp(sum(x));
```

Rozwiązanie:

```
Wektor x:
     7     9    -8     9    -1     3   -15    -8    -5     1

Wektor y:
     0     0     0     0   -20     0  -300     0     0     0

Suma elementów wektora x obliczona pętlą:
    -8

Suma elementów wektora x obliczona funkcją sum:
    -8
```

Zad 6 Proszę użyć konstrukcji while() do wyświetlenia 1 2 3 4 5 6 7 8 9 10(w pionie)

```
>> i = 1;
while i <= 10
    disp(i);
    i = i + 1;
end
     1

     2

     3

     4

     5

     6

     7

     8

     9

    10
```

Zad 7 Oblicz n! dla n=4 z zastosowaniem pętli while().

```
>> n = 4;
f = n;
while n > 1
    n = n - 1;
    f = f * n;
end
disp(['n! = ' num2str(f)])
n! = 24
```

Zad 8 Oblicz „Pierwszą liczbę większą niż 60” dla wektora a = [2 5 6 3 66 33 22] z zastosowaniem pętli while().

```
>> a = [2 5 6 3 66 33 22];
y = 1;
while a(y) <= 60 && y < length(a)
    y = y + 1;
end
if a(y) > 60
    disp('Pierwsza liczba większa niż 60 to:');
    disp(a(y));
else
    disp('Brak liczb większych niż 60 w wektorze a.');
end
Pierwsza liczba większa niż 60 to:
    66
```

Zad 9 Użyj funkcji input() i switch() aby móc wyświetlać odpowiednie przypadki.

```
>> x = input('Wprowadz liczbe: x=');
switch x
    case 1
        disp('jedynka')
    case 2
        disp('dwojka')
    case 3
        disp('trojka')
    otherwise
        disp('cos innego')
end
Wprowadz liczbe: x=3
trojka
```

```
>> x = input('Wprowadz liczbe: x=');
switch x
    case 1
        disp('jedynka')
    case 2
        disp('dwojka')
    case 3
        disp('trojka')
    otherwise
        disp('cos innego')
end
Wprowadz liczbe: x=100
cos innego
```

Po wprowadzeniu wartości 3, program wyświetli trójka. W przypadku wprowadzenia wartości 100 wyświetli cos innego.

Odpowiedzi na pytania:

1. Podstawowe operacje warunkowe to: równość (==), nierówność (~=), mniejsze niż (<), większe niż (>), mniejsze bądź równe (<=), większe bądź równe (>=), koniunkcja (&&) oraz alternatywa (||).
2. Operacje warunkowe pozwalają na tworzenie instrukcji warunkowych, które pozwalają programowi na wykonanie różnych akcji w zależności od spełnienia określonego warunku. Pozwalają one na tworzenie bardziej elastycznych i mniej przewidywalnych programów.
3. Mamy trzy instrukcje iteracyjne (pętle): for, while i do-while.
4. Pętle pozwalają na powtarzanie bloku kodu określoną ilość razy lub do momentu spełnienia określonego warunku. Pozwalają one na bardziej efektywne i elastyczne programowanie, gdyż pozwalają na powtarzanie bloków kodu bez potrzeby pisania ich wielokrotnie.

Podsumowanie

Podczas rozwiązywania zadań skupialiśmy się na praktycznym wykorzystaniu podstawowych elementów języka MATLAB, takich jak pętle for i while, instrukcje warunkowe if, funkcje wbudowane oraz wektory i macierze. Zadania dotyczyły różnych zagadnień, takich jak operacje na wektorach, obliczenia matematyczne, tworzenie macierzy, a także wykorzystanie funkcji input() i disp() do interakcji z użytkownikiem. Poprzez wykorzystanie przykładów praktycznych, zobaczyliśmy, jak w praktyce wykorzystać podstawowe elementy języka MATLAB do rozwiązywania różnych problemów naukowych i inżynieryjnych.