ĆWICZENIE 6: SCILAB: zadanie

Utworzyć skrypt obliczający pierwiastki funkcji kwadratowej ($y=a*x^2+b*x+c$) oraz umożliwiający rysowanie wykresu funkcji z możliwością wyboru zakresu argumentów dla których odbywa się rysowania wykresu.

ALGORYTM:

Uwaga: fragmenty zapisane czcionką pochyloną (kursywa) mogą być zastosowane do utworzenia skryptu.

Skrypt tworzymy w oknie edytora (polecenie: *edit*), i zapisujemy pod nazwą *fkskrypt.*

1. Wprowadzić współczynniki (a, b, c) jednoznacznie określające postać analizowanej funkcji. Uzyskanie wartości tych współczynników powinno być efektem interakcji skryptu z użytkownikiem („odpytanie" i przypisanie odpowiedzi do określonych zmiennych).
   Zastosować instrukcje wejścia: **input**

```
a=input('wprowadź wartość a:  ');          //odpytanie użytkownika o wartość wsp. a
b=input('wprowadź wartość b:  ');          //odpytanie użytkownika o wartość wsp. b
c=input('wprowadź wartość c:  ');          //odpytanie użytkownika o wartość wsp. c
```

2. Obliczyć „**deltę"** analizowanego równania kwadratowego:

```
delta=b.^2-4*a*c;
```

3. W zależności od wartość i znaku „delty" obliczyć pierwiastki równania i/lub wyświetlić odpowiednie komunikaty.

Jeżeli **delta** jest większa od zera:

```
if delta > 0                    // początek instrukcji warunkowej if,
                                sprawdzenie warunku czy jest  delta > 0
```

to równanie ma dwa pierwiastki rzeczywiste obliczane wg wzorów:

```
x(1)=(-b-sqrt(delta))/(2*a);
x(2)=(-b+sqrt(delta))/(2*a);    // obliczenie dwóch pierwiastków równania
                                kwadratowego, w przypadku gdy spełniony jest
                                warunek:  delta > 0
```

Obliczone pierwiastki zostaną przypisane do zmiennej **x**, która jest wektorem: **x=[x(1), x(2)].**
Po obliczeniu pierwiastków należy ich wartości wyświetlić na ekranie. Znając (mając wyświetlone) wartości pierwiastków będzie można odpowiednio dobrać zakres rysowania wykresu, tak aby oba obliczone pierwiastki były widoczne na wykresie.

```
disp('Delta >0 ')                              //wyświetlenie komunikatu
disp('Funkcja posiada dwa miejsca zerowe:')    //wyświetlenie komunikatu
disp(x)                                        //wyświetlenie wartości zmiennej x
```

Jeżeli **delta** jest równa zero :

```
elseif delta == 0               // kontynuacja instrukcji warunkowej if
                                sprawdzenie warunku czy jest  delta == 0
```

to równanie ma jeden pierwiastek rzeczywisty obliczany wg wzoru:

```
x=-b/(2*a);
```

Informacje wyświetlane na ekranie:

```
disp('Delta =0 ')                              //wyświetlenie komunikatu
disp('Funkcja posiada jedno miejsce zerowe:')  //wyświetlenie komunikatu
disp(x)                                        //wyświetlenie wartości zmiennej x
```

Jeżeli **delta** jest mniejsza od zera

else *// kontynuacja instrukcji warunkowej **if.** Po wyrażeniu **else** nie wpisujemy warunku, który byłby sprawdzany, ponieważ działania zapisane po **else** będą realizowane gdy żaden z wcześniejszych warunków (po **if** oraz **elseif**) nie był spełniony. W naszym przypadku oznacza to sytuację, w której delta **nie** jest większa od zera i **nie** jest równa zero.*

to równanie nie ma pierwiastków rzeczywistych.
Należy obliczyć współrzędną wierzchołka paraboli, aby umożliwić późniejsze właściwe dobranie zakresu rysowania wykresu:

*xw=-b/(2*a); //wartość współrzędnej przypisujemy do nowej zmiennej: xw*

Informacje wyświetlane na ekranie:

***disp**('Delta <0 ') //wyświetlenie komunikatu*
***disp**('Funkcja nie posiada miejsc zerowych:') //wyświetlenie komunikatu*
***disp**('Współrzędna wierzchołka paraboli:') //wyświetlenie komunikatu*
***disp**(xw)*
end *//zakończenie instrukcji warunkowej **if***

Powyższy zapis skryptu powinien umożliwić wyliczenie i wyświetlenie wartości miejsc zerowych (gdy delta>=0), lub współrzędnej wierzchołka paraboli (gdy delta<0). Zapisać plik skryptu i przetestować działanie.

4. W dalszej części skryptu należy zapisać możliwość rysowania wykresu.
   Zastosować instrukcje wejścia: **input** w celu podjęcia decyzji o rysowaniu wykresu:
   w=***input**('Czy narysować wykres? T-tak, N-nie ','s') ;*

Odpowiedź na powyższe pytanie będzie miała charakter tekstowy (znak) i zostanie przypisana do zmiennej tekstowej ***w***. Oczekiwane odpowiedzi to **T**-jeśli chcemy narysować wykres lub **N** - jeśli nie chcemy narysować wykresu. Należy wprowadzić zabezpieczenie przed wybraniem innego znaku niż **T/N**. Algorytm takiego zabezpieczenia jest następujący:
dopóki nie zostanie podana właściwa wartość T/N ponawiaj pytanie o rysowanie wykresu.
Powyższe można zrealizować stosując polecenie **while:**

while** (w~='T' & w~='N') //początek instrukcji **while

wyrażenie logiczne (w~='T' & w~='N') oznacza w jest różne niż T i w jest różne niż N

w=***input**('Czy narysować wykres? Należy wybrać: T-tak, N-nie ','s') ;*
end *//koniec instrukcji **while.** Warunkiem zakończenia pętli **while** jest podanie przez użytkownika jednej z wymaganych postaci odpowiedzi T/N. Każda inna odpowiedź spowoduje ponowne wykonanie pętli i pojawienie się monitu z zapytaniem.*

powyższa pętla spowoduje ponawianie pytania o rysowanie wykresu aż do uzyskania odpowiedzi T/N.
Jeśli odpowiedź będzie: T

if** w=='T' // początek instrukcji warunkowej **if

algorytm odpytuje o zakres w jakim ma być narysowany wykres.
Stosujemy odpowiednio instrukcje wejścia: **input:**

*Xstart=**input**('Podaj początek zakresu rysowania wykresu')*
*Xend=**input**('Podaj koniec zakresu rysowania wykresu')*

Tworzymy wektor 100-argumentów rysowanej funkcji stosując polecenie **linspace**:

*xwykres=**linspace**(Xstart, Xend, 100);*

Tworzymy wektor wartości rysowanej funkcji

```
ywykres=a*xwykres.^2+b*xwykres+c;
plot(xwykres,ywykres);
```

Kolejnym krokiem jest zaznaczenie na wykresie miejsc zerowych.

```
if delta > 0                          // początek instrukcji warunkowej if
    plot(x,[0,0],'or')
elseif delta = = 0                    // kontynuacja instrukcji warunkowej if
    plot(x,0,'or')
end                                   // zakończenie instrukcji warunkowej if
```

W tym miejscu można zastosować dodatkowe polecenia dotyczące wyglądu wykresu np.:

```
set(gca(),'grid',[1,1])       // rysowanie siatki na wykresie
end                           // koniec instrukcji warunkowej if
```

Dopisać dodatkowe instrukcje/polecenia dodające inne elementy wykresu:

- tytuły osi
- tytuł wykresu
- legendę.

Zapisać uzyskany plik skryptu i przetestować działanie.

Zadania:

1. Zmienić sposób wprowadzania współczynników a, b, c tak aby były wprowadzane jednocześnie jako wektor złożony z trzech liczb.

2. Zmodyfikować sposób wyświetlania komunikatów z punktu 3-go algorytmu, stosując polecenie ***write*** zamiast polecenia ***disp***. Komunikaty powinny być wyświetlane w dwóch wierszach:

- w pierwszym: informacja o tym czy delta jest większa, równa czy mniejsza od zera oraz gdy delta <0 dodatkowa informacja o braku pierwiastków,
- w drugim: informacja o ilości pierwiastków i ich wartościach, lub o współrzędnej wierzchołka paraboli (gdy delta<0).

3. Jeżeli argumenty do rysowania wykresu (wektor *xwykres*) zawierają zero, to osie układu współrzędnych powinny przecinać się w punkcie (0,0).

Zmodyfikowany skrypt zapisać pod nową nazwą *fkskrypt_2.*