ŘEŠENÍ SOUSTAVY A X = B ITERAČNÍMI METODAMI

Upozornění: cílem je zde názorná ilustrace metod, nikoliv efektivní program

obecná metoda prosté iterace:

řešíme rovnici X = F(X) jako $X_{i+1} = F(X_i)$, i = 1, 2, ... (X může být číslo, vektor, matice ...)

příklad pro F(X) = sin(X) + 1:

```
>> X = 6;           % volba pocatecni aproximace
>> X = sin(X) + 1   % opakujeme, dokud se X neustali
```

metoda prosté iterace pro rovnici X = U X + V:

```
>> U = [ 0.5  0.1 ; 0.2  0.3 ];  % dana matice
>> V = [ 2 ; 3 ];                % dany vektor
>> X = [ 6 ; 4 ];                % volba pocatecni aproximace
>> X = U * X + V                 % opakujeme, dokud se X neustali
```

převedení rovnice A X = B na rovnici X = U X + V

```
>> A = [ 2 -1  0 ; -1 3 1 ; 0 1 2 ];  % dana matice
>> B = [ 1 ; 3 ; 3 ];                 % dany vektor prave strany
```

Richardsonova metoda:

```
>> E = eye(3);            % jednotkova matice
>> c = 0.2;               % vhodna konstanta (zavisla na matici)
>> U = E - c * A
>> sp = max(abs(eig(U)))  % spektralni polomer matice U - kriterium konvergence
>> V = c * B;
>> X = V;                 % volba pocatecni aproximace
>> X =  U * X + V         % opakujeme, dokud se X neustali
>> B - A * X              % residuum - melo by byt blizko nuly
```

Zkuste různé volby konstanty c.
Přesvědčte se, že pro sym. A metoda konverguje, právě když max(|1 - c emin|, |1 - c emax|) je menší než 1 (emin a emax jsou minim. a max. vlastní čísla A).

Jacobiova metoda:

```
>> D = diag(diag(A));       % diagonala A
>> L = tril(A, -1);         % dolni trojuhelnik A (bez diagonaly)
>> P = triu(A, 1);          % horni trojuhelnik A (bez diagonaly)
>> UJ = - inv(D) * (L + P)  % Jacobiova iteracni matice
>> sp = max(abs(eig(UJ)))   % spektralni polomer matice UJ - kriterium konvergence
>> VJ = inv(D) * B;         % upraveny vektor prave strany
>> X = VJ;                  % volba pocatecni aproximace
>> X = UJ * X + VJ          % opakujeme, dokud se X neustali
>> B - A * X                % residuum - melo by byt blizko nuly
```

Gauss-Seidelova metoda:

```
>> D = diag(diag(A));       % diagonala A
>> L = tril(A, -1);         % dolni trojuhelnik A (bez diagonaly)
>> P = triu(A, 1);          % horni trojuhelnik A (bez diagonaly)
>> UG = - inv(D + L) * P    % Gauss-Seidelova iteracni matice
>> sp = max(abs(eig(UG)))   % spektralni polomer matice UG - kriterium konvergence
>> VG = inv(D + L) * B;     % upraveny vektor prave strany
>> X = VG;                  % volba pocatecni aproximace
>> X = UG * X + VG          % opakujeme, dokud se X neustali
>> B - A * X                % residuum - melo by byt blizko nuly
```

Metoda SOR:

```
>> om = 1.25;                   % volba relax. parametru omega
>> D = diag(diag(A));           % diagonala A
>> L = tril(A, -1);             % dolni trojuhelnik A (bez diagonaly)
>> P = triu(A, 1);              % horni trojuhelnik A (bez diagonaly)
>> US = - inv(D + om*L) * (om*P + (om-1)*D);    % SOR iteracni matice
>> sp = max(abs(eig(US)))       % spektralni polomer matice US - kriterium konvergence
>> VS = om * inv(D + om*L) * B;  % upraveny vektor prave strany
>> X = VS;                      % volba pocatecni aproximace
>> X = US * X + VS              % opakujeme, dokud se X neustali
>> A * X                        % zkouska
```

Vyzkoušejte metody na dalších maticích, například

```
>> A = [ 2 -1  0 ; -1 -3 1 ; 0 1 2 ];
```

(změna znamínka u prostředního prvku - Richardsonova metoda přestane fungovat).