Vaja 3 – Reševanje sistemov linearnih enačb (Gaussova eliminacija)

1. Splošna predstavitev problema

V inženirskih aplikacijah je eden najpogostejših problemov reševanje sistemov n linearnih enačb z n neznankami. Ker se ti problemi niso pojavili šele z nastankom računalnika, ne preseneča, da obstaja kar nekaj matematičnih pristopov k reševanju teh problemov. Eden najbolj osnovnih je Gaussova eliminacija, ki je vgrajena tudi v veliko matematičnih programov. Gaussova eliminacija je v svoji ideji zelo enostavna. Sistem linearnih enačb zapišemo v matrično obliko, v tako imenovano razširjeno matriko, nato pa z odštevanjem vrstic dosežemo, da dobimo zgornje-trikotno matriko. V kolikor je sistem linearnih enačb rešljiv, rešitev dobimo s substitucijo nazaj. Kot primer poglejmo naslednji sistem linearnih enačb:

$$9x_1+3x_2+4x_3=7,$$
$$4x_1+3x_2+4x_3=8,$$
$$x_1+x_2+x_3=3.$$

Sistem zapišemo v razširjeno matriko:

$$\left[\begin{array}{ccc|c} 9 & 3 & 4 & 7 \\ 4 & 3 & 4 & 8 \\ 1 & 1 & 1 & 3 \end{array}\right].$$

Gradnjo zgornje trikotne matrike pričnemo z zamenjavo prve in tretje vrstice, saj je v tretji vrstici vrednost v prvem stolpcu najmanjša. Tako dobimo matriko:

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 3 \\ 4 & 3 & 4 & 8 \\ 9 & 3 & 4 & 7 \end{array}\right],$$

nato od druge vrstice odštejemo štirikratnik prve vrstice, od tretje pa devetkratnik prve vrstice. Tako dobimo matriko:

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 3 \\ 0 & -1 & 0 & -4 \\ 0 & -6 & -5 & -20 \end{array}\right].$$

Kot lahko vidimo, smo tako v prvem stolpcu druge in tretje vrstici pridelali ničlo in tako naredili prvi korak proti želeni zgornje-trikotni matriki. Do te nam manjka samo še en korak in sicer tudi v drugem stolpcu tretje vrstice moramo dobiti ničlo. To dobimo tako, da od tretje vrstice odštejemo šestkratnik druge vrstice in dobimo matriko:

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 3 \\ 0 & -1 & 0 & -4 \\ 0 & 0 & -5 & 4 \end{array}\right].$$

Iz tako dobljene matrike lahko brez težav izračunamo, da je spremenljivka x_3 enaka $-4/5$, z enako malo napora pa še, da je x_2 enak 4 in nazadnje še iz prve vrstice, da je spremenljivka x_1 enaka $-1/5$.

Zgornji primer je bil specifičen, saj je bila prva neničelna vrednost v stolpcu v katerem smo želeli pridelati ničlo, to vrednost imenujemo tudi pivot, enaka ena. V kolikor temu ne bi bilo tako bi morali celotno vrstico deliti z ustrezno vrednostjo in tako pridelati vrednost ena v pivotu. Seveda je mogoče tudi to, da sistem linearnih enačb rešitve nima, oziroma je lahko rešitev neskončno mnogo. Oba primera sta v inženirskih aplikacijah redka, a jih vseeno ne smemo zanemariti.

2. Pomoč pri implementaciji

Opisani postopek lahko zapišemo v obliki psevdokoda, ki je prikazan v izpisu 1.

```
function GAUSS_ELIMINATION(A, n)
 begin
   for k:=0 to n-2 do
     begin
       Poišči najmanjši člen |a_jk| ≠ 0 (j ≥ k);
       if iskani člen a_jk ne obstaja then
         return ERROR;
       else
         begin
           Zamenjaj vrstici j in k;
           Deli celotno vrstico s prejšnjo vrednostjo a_jk;
         end
       for l:=k+1 to n-1 do
         Odštej od vrstice l vrstico k, pomnoženo z a_lk;
     end

   if a_n-1,n-1 = 0 then
     return ERROR;

   x[n-1] := a_n-1,n/a_n-1,n-1;
   for i:=n-2 to 0 do
     x[i] := 1/a_ii(a_in − Σ_{j=i+1}^{n-1} a_ij x[j]);

   return x;
end
```

Izpis 1: Reševanje sistema linearnih enačb z Gaussovo eliminacijo

Funkcija `GAUSS_ELIMINATION` kot vhodni parameter vzame razširjeno matriko **A** ter število linearnih enačb v sistemu. Kot izhod vrne vektor rešitve, če le-ta obstaja.

3. Zahteve naloge

Implementirati je potrebno aplikacijo, ki bo sposobna rešiti poljubno velik sistem *n* linearnih enačb z *n* neznankami. Če rešitev ne obstaja ali pa je rešitev neskončno mnogo naj aplikacija vrne napako z ustreznim opisom, saj tovrstni sistemi niso zanimivi za inženirske aplikacije. Linearni sistem enačb bo predstavljen v obliki razširjene matrike, ki bo shranjena v tekstovni datoteki z imenom *sistem.txt*. Struktura datoteke *sistem.txt* je za primer iz prvega poglavja naslednja:

```
3
9 3 4 7
4 3 4 8
1 1 1 3
```

Aplikacija naj vrne rešitev v naslednji obliki:

```
x1:  -0.20
x2: 4
x3: -0.80
```

V kolikor rešitev ne obstaja, mora program to izpisati.