Pokazivači i višedimenzionalni nizovi

Uvod

VIŠEDIMENZIONI NIZ ELEMENATA

- Prilikom programiranja često se javlja potreba za smeštanjem i manipulisanjem podacima u dvodimenzionim strukturama kao što su matrice i tabele.
- Deklaracija promenljive čiji tip je višedimenzioni niz elemenata.
- tip_elemenata ime_promenljive[BR_VRSTA][BR_KOLONA]
- **int A[3][4]**

Primeri

- Za deklarisanje matrice dimenzija 3x4 u koju ćemo da smeštamo cele brojeve pišemo: **int A[3][4]**
- Inicijalizacija višedimenzionog niza prilikom deklarisanja: **int A[3][2] = { {1, 2}, {3, 4}, {5, 6}}** **ili int A[3][2] = {1, 2, 3, 4, 5, 6}**
- Pristupanje elementu i-te vrste i j-te kolone: **A[i][j]**

Pristup članovima

- U memoriji se elementi smeštaju redom iz svake vrste. Prvo redom elementi iz prve vrste. Odmah za njima redom elementi iz druge vrste i tako redom, vrsta po vrsta.
- Kao i kod jednodimenzionih nizova, ime niza je adresa prvog elementa.
- Tako je: A adresa elementa A[0][0]
- A + 4 adresa elementa A[0][1], jer se redom smestaju u memoriju.
- Svaki je int zauzima 4B
- A + 8 adresa elementa A[1][0],
- posle prve vrste se smeštaju elementi iz druge vrste
- A + 12 -> A[1][1]
- A + 16 -> A[2][0]
- A + 20 -> A[2][1]...

Nizovi nizova = matrice

- Višedimenzioni niz se može posmatrati kao matrica, ali i kao niz nizova.
- Stoga možemo reći:

int A[3][2]; je deklaracija niza od 3 elementa.

- Svaki element niza A je tipa niz od 2 cela broja. A[0] je niz od 2 cela broja iz prve vrste
- A[1] je niz od 2 cela broja iz druge vrste
- A[2] je niz od 2 cela broja iz treće vrste ...

Adrese

- Iz ove perspektive A kao ime niza je adresa prvog elementa, tj. A[0].
- Kako je A[0] tipa niz od 2 cela broja, on je kao ime tog niza i adresa svog prvog elementa, tj. elementa A[0][0].

Zadaci

- Napisati funkciju koja ucitava matricu datih dimenzija
- Napisati program koji koristi ovu funkciju

Dinamicka alokacija matrice

- Deklarišemo pokazivač na pokazivač na double:

  double **p;

- Ovaj pokazivač će nam služiti da čuva adresu prvog u nizu dinamički alociranih pokazivača na double element.

Dinamicka alokacija matrica

- Dinamički alociramo niz od n pokazivača na double-ove, i adresu prvog u nizu smeštamo u promenljivu p:

  p = malloc(sizeof(double*) * n);

Primetimo da je ime tipa – double * -- pokazivač na double.

Ovih n pokazivača će nam služiti da pokazuju na prve elemente nizova double-ova koji predstavljaju dinamički alocirane vrste.

- Svaku od n vrsta dinamički alociramo posebnim pozivom malloc f-je.
- Povratnu adresu smeštamo redom u malopre alocirani niz pokazivača.
- Za i-tu vrstu povratnu vrednost smeštamo u p[i].

for(i = 0; i < n; i++) p[i] = malloc(sizeof(double) * m);

Primetimo da nizovi double-ova koji se alociraju nisu morali biti svi jednakih dimenzija. Nisu morali svi imati m elemenata kao što je u primeru.

Ovo može biti korisno u situacijama kada znamo da će u nekoj vrsti biti korišćeno samo nekoliko prvih elemenata. Tada možemo alocirati manje prostora za tu vrstu i tako uštedeti prostor.

Pristup elementima dinamicki alocirane matrice

- Elementima dinamički aocirane matrice pristupamo isto kao i kod automatski alocirane matrice, sa **p[i][j]**.
- **p[i]** je i-ti pokazivač u nizu, a kako on pokazuje na početni element u i-toj vrsti, tada je **p[i][j]** upravo j-ti elementi u i-toj vrsti.

Dealokacija dinamicki alocirane matrice

- Dealokacija se sastoji u oslobađanju dinamički alociranih vrsta, nakon čega se oslobađa niz pokazivača.
- Dakle, dealokacija ide u suprotnom redosledu od redosleda alokacije.

for(i=0 ; i< n; i++) free(p[i]); free(p);

Zadatak

- Napisati program kojim učitava sa standarnog ulaza realnu matricu dimenzija 10x10, a zatim računa:
- trag matrice,
- euklidsku normu i
- gornju vandijagonalnu normu.

Resenje

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main(){
 int i,j;
 double** A; /* pokazivac na dinamicki alociran niz pokazivaca na vrste
     matrice */
   int n; /* broj vrsta */
   int m; /* broj kolona */
   double trag; /* trag matice */
   double norma; /* Eukliska norma */
  double vdnorma; /* Vandijagonalna norma */
```

```
/* Unosimo dimenzije matrice*/
   printf("Unesite broj vrsta i broj kolona matrice.\n");
    scanf("%d%d",&n,&m);

/* dinamicki alociramo prostor za n pokazivaca na double-
    ove */
   A = malloc(sizeof(double*) * n);
/* proveramo da li je doslo do greske pri alokaciji */

if(A == NULL){
    fprintf(stderr,"Malloc() greska!\n"); exit(EXIT_FAILURE);
  }
```

```
/* dinamicki alociramo prostore za elemente u vrstama */
for(i = 0; i < n; i++){
     A[i] = malloc(sizeof(double) * m);

    if(A[i] == NULL){
      /* Dalje ne mozemo da nastavimo.
         Trebalo bi da napustimo program,
       ali pre toga moramo da oslobodimo svih i-1 prethodno
    alociranih vrsta,   i alociran niz pokazivaca. */
      for( j=0; j<i; j++)  free(A[j]);
           free(A);
       /* sada mozemo da napustimo program */
      exit( EXIT_FAILURE); }
    }
```

```
/* Unosimo sa standardnog ulaza brojeve u
    matricu. Popunjavamo vrstu po vrstu.
   i - brojacka promenjiva za vrste matrice
   j - brojacka promenjiva za kolone matrice */
printf("Unesite elemente matrice:\n");


  for( i = 0; i < n; i++ )
/* ucitavamo elemente i-te vrste */
      for(j = 0; j < m; j++ ){
         printf("A[%d][%d] = ", i, j);
         scanf("%lf", &A[i][j]);
}
```

```
/* ispisujemo elemente matrice */
for(i = 0; i < n; i++){
   for( j = 0; j < m; j++)
       printf("%.2f ", A[i][j]);
  printf("\n");
}
```

```
/* Racunamo trag matrice, tj. sumu
   elemenata na glavnoj dijagonali. */
   trag = 0.0;
  for(i=0; i<n; i++)  trag += A[i][i];

  printf("Trag matrice je %.2f.\n",trag);
```

```
• /* Racunamo euklidsku normu matrice,
   tj. koren sume kvadrata svih elemenata. */
norma = 0.0;
for( i=0; i<n; i++)
   for( j=0; j < m; j++) norma += A[i][j] * A[i][j];

norma = sqrt(norma);
printf("Euklidska norma matrice je %.2f.\n",\
norma);
```

```
/* Racunamo gornju vandijagonalnu normu
   matrice, tj. sumu apsolutnih vrednosti
   elemenata iznad glavne dijagonale */
vdnorma = 0.0;
for( i=0; i< n; i++)
   for( j= i+1; j<m; j++)
     vdnorma += fabs(A[i][j]);

printf("Gornja vandijagonalna norma \
   matrice  je %.2f./\n", vdnorma);
```

```
/* Oslobadjamo prostor */
for( j=0; j<n; j++) free(A[j]);

free(A);

return 0;
}
```