Tartalomjegyzék

Elmélet

Google Drive

https://docs.google.com/document/d/1k8hpm1o9429d-Gs_OFAJMtkUWACGdYxvr0baZCnq6b8/edit

2013 el?adás

- Mivel az el?adás el?tt járunk, itt a múlt év el?adásának anyaga: http://wiki.math.bme.hu/view/Informatika2-2013/Eloadas

Mutatók

- El?ször tekintsünk egy egyszer? modellt a számítógép memóriájára, illetve az ebben történ? adattárolásra.
- A memóriát képzeljük el diszkrét adattároló egységeknek. Minden egységnek külön tároljuk a címét a memóriában.
- Az egységeket külön-külön, vagy tömbösítve tudjuk kezelni.
- Ezek alapján értlemezzük a nyelv absztrakt szintjén a tömböket és pointereket.
- Memóriacím egy egész szám, amely kijelöli a memória egy bájtját.
- Az **&** operátor hívása adja meg az adott változó **címét** a memóriában.
- **Pointernek** nevezzük azt a változót, amely egy másik változó memóriacímét tartalmazza.
- Pointer típusa: milyen típusú adatra vagy függvényre mutat a pointer.
- Ha ***** operátort (melynek neve indirekció) egy pointerre hattatjuk, akkor visszaadja az adott pointer által megcímzett értéket.
- Pointerek deklarációja:

```
int *p;
```

- NULL pointer: olyan, külön erre a célra fentartott érték? pointer, mely nem mutat semmilyen értékre:

```
int *p=NULL;
```

Tömbök

- Tömbök esetén "http://wiki.math.bme.hutömbösítve"http://wiki.math.bme.hu foglaljuk le a változók helyét a memóriában.
- Példa deklarációra:

```
int a[10];
```

- Példa inicializálásra:

```
int t[10]={6,8,-2,6,123,-8,3,4,2,1};
```

- A tömb elemei 0-tól indexel?dnek.
- Deklaráláskor tehát **le kell rögzítenünk egy konstanssal a tömb méretét**
- Fontos, hogy a méretet csak konstanssal adhatjuk meg. Néhány fordító támogatja a tömb méretének változóval való megadását, de ezt általánosan nem alkalmazhatjuk.
- Ha fordítási id?ben nem ismert, mekkora tömbre lesz szükség, használjuk dinamikus tömböt. (lsd. kés?bb)
- Mindig kötelez? a tömbnek méretet adni.
- A tömb nem másolható az = operátorral, végig kell iterálnunk elemenként a két tömbön másoláskor:
- Sztringeket karaktertömbökként tárolunk.
- A sztring utolsó eleme mindig a "http://wiki.math.bme.huszöveg végét"http://wiki.math.bme.hu jelöl? "http://wiki.math.bme.hu\0"http://wiki.math.bme.hu karakter. Tehát a sztringben lév? karakterek számánál mindig egyel nagyobb a tároló karaktertömb mérete.
- az **a** és &a[0] jelölések ekvivalensek, és a tömb els? elemére mutatnak. A következ? kód végén a pa változó pedig az a tömb 6. (indexe 5) elemére mutat:

```
int *pa;
pa = &a[0];
pa = &a;
*(pa + 5);
```

- Lehet?ségünk van többdimenziós tömbök deklarálására.
- Többdimenziós tömbök sorfolytonosan tárolódnak a memóriában, tehát a következ? két inicializálás (mely egyben példa is) ekvivalens:

```
int t[3][4][2]={34,-5,3,20,12,5,-1,0,4,77,12,-3,-14,3,1,23,75,2,10,24,78,1,2,7};

int t[3][4][2]={{{34,-5},{3,20},{12,5},{-1,0}},{{4,77},{12,-3},{-14,3},{1,23}},{{75,2},{10,24},{78
```

Dinamikus memória foglalás

- Függvények:
  - **malloc** megadott byte-ot foglal le a memóriában,
  - **realloc** a már lefoglalt memória-tömb méretét változtatja meg,
  - **calloc** memóriafoglalás és a lefoglalt memória byte-jainek 0-ra állítása egyben,
  - **free** üríti a megadott memória-részt.
- Példa a **malloc** függvény használatára

- Példa egydimenziós, dinamikusan foglalt tömb lefoglalására és használatára:

```
  ...
  int m;
```

```
scanf("%d",&M);
int *vec = (int *) malloc(M * sizeof (int));
...
int i;
for(i=0; i<M; i++){
  vec[i]=1/(double)i;
}
...
```

- Példa kétdimenziós, dinamikusan foglalt tömb lefoglalására és használatára:

```
...
int M,N;
scanf("%d",&M);
scanf("%d",&N);
int **matrix = (int **) malloc(M * sizeof (int *));
int i;
for (i = 0; i < M; i++){
  matrix[i] = (int *) malloc(N * sizeof (int));
}
...
int j;
for(i=0; i<M; i++){
  for(j=0; j<N; j++){
    matrix[i][j]=(i+1)*(j+1);
  }
}
```

Feladatok

El?z? gyakorlat feladatai

Összeg

- Írj függvényt, mely összeadja egy tömb elemeit.

Rendezés

- Írj függvényt, mely nagyság szerint rendezi egy tömb elemeit.

Sakktábla

- Rajzolj ki egy *NxN*-es sakktábla mintát, ahol X-szel jelöljük a fekete mez?ket, és üresen hagyjuk (egy szóköz) a fehéreket. Nem kell keretet adni a táblának.
- A sakktábla méretét (*N*) a felhasználótól kérd be.

Mátrixok, 2 dimenziós tömbök

- Írd ki egy 2 dimenziós tömb elemeit a képerny?re. Próbáld ki a függvényt úgy, hogy egy main függvényen belül hozol létre egy példa tömböt.
- Hasonlóan számold ki és jelenítsd meg a mátrix transzponáltját és négyzetét is.
- Írj függvényt, mely összead / kivon egymásból / összeszoroz 2 mátrixot és a megoldást megjeleníti a képerny?n.

Prímtényezõ keresés

- Írj programot, ami megkeresi egy a felhasználó által adott szám prímtényezõit és sorban kiírja azokat.
- Meg lehet oldani úgy is, hogy egyesével megpróbáljuk elosztani az adott számunkat 2-tõl kezdve egyesével haladva egész számokkal, amíg 1-hez nem jutunk.
- A maradék képzés (modulo) jele C-ben is a %

Gyakoriság, hisztogram

- Írj függvényt, mely megszámolja egy egész számokból álló tömb elemeinek a gyakoriságát.
- A függvény paraméterként kapja meg a vizsgált tömböt és annak méretét. Feltételezzük, hogy a tömbben 20-nál kisebb, vagy egyebl? értékek szerepelnek (miért van erre szükség?).