INFORMATIKA LABORATÓRIUM I.

Pointerek

Pointerek

Az eredményes C nyelvű programozáshoz elengedhetetlen a pointerek megértése és felhasználása.

Okai:

- Pointerek segítségével lehet a függvényeken belül a hívási argumentumokat módosítani.
- Ezeket használják a C dinamikus allokáló rutinjai.
- A pointerek bizonyos rutinok eredményességét javíthatják.

A C legjellemzőbb tulajdonságainak egyike a pointerek alkalmazása, de ez egyúttal a legveszélyesebb vonása is.

Például inicializálatlan vagy vad pointerek képesek a rendszert tönkretenni. A C nyelv megengedi, hogy bárminek a címét előállítsuk. Például: változók, tömbök, struktúrák és függvények címét.

Pointer változók - példa

```
char   *p;
int   *count_addr,  *start;
count_addr = &count;       /* (a count változó címe) */
val = *count_addr;         /* (a count értéke val-ba kerül)*/
main ( )
{
      int *count_addr, count, val;
      count = 100;
      count_addr = &count;
      val = *count_addr;
      printf ( "%d", val )       /*megjelenik a 100*/
}
```

Pointerek

- Egy pointer változó típusa ugyanaz kell, hogy legyen, mint annak a változónak a típusa, amelynek a pointere.
- Például, ha egy változó egész típusú, akkor a pointere is egész kell, hogy legyen.
- Eltérő deklarálás esetén hibaüzenetet nem kapunk, de bizonyos műveletek hibásan mennek végbe.

Például:

```
main ( )
{
        float   x,y;
        int  *p;
        p = &x;
        y = *p;                 /* y-ba csak 2 bájtos információ kerül át */
}
```

- Nagyon ügyeljünk arra, hogy egy pointert mielőtt felhasználnánk, inicializálnunk kell.
- Egy pointer definiálása még nem jelenti annak inicializálását.

Pointer kifejezések

A pointer kifejezések hasonlóak más C kifejezésekhez. Néhány speciális aspektust fogunk vizsgálni.

Pointer átadás

Egy pointert használhatunk egy értékadó kifejezés jobb oldalán, egy másik pointernek így adunk értéket.

```
main()
{
   int x;
   int *p1; *p2;
   p1 = &x;
   p2 = p1;      /* értékadás */
}
```

Pointer kiíratása

Egy pointer tartalmát kiírhatjuk:

```
//---------------------------------------------------------------------
main()
{
    int x;
    int *p1; *p2:;
    p1 = &x;
    p2 = p1;              /* értékadás! */
    printf ("%p",p2);        /* x címe kerül kiírásra */
    printf ( "%u", p2 );         /* x címe decimálisan kerül kiírásra */
}
```

Pointer aritmetika

Pointerekre a négy aritmetikai operátor (+ , - , ++ , --) használható a következő értelmezéssel:

a. p1++
ha p1=2000 volt, akkor 2002 lesz nem 2001!

Az inkrementált pointer a következő egészre mutat.

b. p1--
ha p1=2000 volt, akkor 1998 lesz.

c. p1=p1+9
char *ch=3000
int *i =3000

ch	3000	i
ch+1	3001	
ch+2	3002	i+1
ch+3	3003	
ch+4	3004	i+2
ch+5	3005	

Pointerek és tömbök

Egy tömbnév index nélkül a tömb kezdetének a címét adja.

```
char str[80];
char *p1;
p1 = str;
```

Hivatkozás a tömb elemeire index és pointer segítségével is történhet.
Példa:

```
char i,  j, str [80 ], -*p1 ;

i = str [4]
j = *( p1+4 )
```

Mindkét esetben a sztring ötödik elemére hivatkozunk.