Tartalomjegyzék

Ismétlés

- Beolvasás scanf-el:

```
int z;
printf("Add meg z értékét: ");
scanf("%d", &z);
```

Az &-et ne hagyjuk ki.

- Ciklusok: for, while, do-while

- Megírtunk egy egyszerû programot, ami egyesével összeadta a pozitív egész számokat növekvõ sorrendben amíg el nem ért egy küszöböt. Tanulság:
  - For ciklus feltétele nem feltétlen függ közvetlenül a ciklusváltozótól.
  - Ha elakadnánk lehetséges, hogy csak egy új változót kellene felvenni.

- Írtunk programot amivel akárhány számot beolvashattunk és akkor állt le ha egymás után azonosat adtunk neki. Tanulság:
  - Ha elõre nem ismert számú adatot akarunk beolvasni, hasznos lehet egy while vagy do-while

ciklus.
    - Korábbi adatokat eltárolhatunk egy változóban, ha szükségünk lehet még rájuk.

- Írtnunk minimum és/vagy maximum keresõ programot. Tanulság:
    - Még ha elõre ismert számú adatot kell beolvasnunk akkor se feltétlen praktikus mindet eltárolni.
    - Tudunk olyan algoritmusokat írni amik "http://wiki.math.bme.huon the fly"http://wiki.math.bme.hu mûködnek, miközben a beolvasás történik már fut a számolás, nem csak a beolvasás után.

Feladatok 1

Miért nem fut végtelen ciklusba a következ? kód?

```
#include<stdio.h>
int main(void){
    char i;
    for(i = 1; i != 0; i++){}
    return 0;
}
```

1. 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!

Tipp: a ciklusokat egymásba is ágyazhatjuk, de ilyenkor nagyon kell figyelni a ciklusváltozókra!

Egymásba ágyazott ciklus példa:

```
#include<stdio.h>

int main(void) {
    int i;
    int j;

    for(i = 0; i < 10; i++) {
        printf("i: %d \n", i);
        for(j = 0; j < i; j++) {
            printf(" (%d, %d) ", i, j);
        }
        printf("\n");
    }
    printf("\n");
}
```

2. Prímtényezõ keresés

Írj programot, ami megkeresi egy a felhasználó által adott szám prímtényezõit és sorban kiírja azokat. (A hiba elkerülése végett elõször vizsgáljuk meg, hogy nem 0-t vagy 1-et kaptunk-e.)

Nem kell bonyolultra gondolni azonnal, meg lehet oldani úgy is, hogy egyesével megpróbáljuk elosztani az adott számunkat 2-tõl kezdve egyesével haladva egész számokkal, amíg 1-hez nem jutunk.

Emlékezzünk, hogy a maradék képzés (modulo) jele C-ben is a %

A gyakorlat anyaga

Kulcsszavak

Minden programnyelvnek vannak védett kulcsszavai, ezek speciális jelentéssel bírnak, és nem használhatjuk õket változónévként.

Adat típusok

	Formátum megadó	Rövid leírás
void	nincs	Típus nélküli.
int	%d, %i, %u, %o, %x	Egész szám, leggyorsabban tud vele számolni a processzor.
char	%c	Karakter, kisebb méretû mint az int, de valójában õ is szám, pl 'm' == 109.
float	%f, %e, %g, %a	Lebegõpontos szám, fordítótól függ a mérete.
double	%lf	Lebegõpontos szám, elméletileg 2-szer olyan pontos mint a float, gyakorlatilag fordítótól függ.

Léteznek még módosítók, ezek a következõk: short, long, signed, unsigned.

Vezérlési szerkezetek

Ezeket már láttuk, for, if, else, while, do, ami eddig még nem jelent meg: switch, case.

Ide lehetne sorolni még a következõket:

	Rövid leírás
break	Kilép a legbelsõbb ciklusból.
continue	Továbblép a legbelsõbb ciklus következõ lépésére.
goto	Egyik futási részrõl a másikra ugorhatunk. Soha ne használjuk.
return	Visszatér az adott függvény, esetleg értéket is ad vissza.

További kulcsszavak is léteznek, ezekrõl majd késõbb lesz szó.

Operátorok

Már volt róluk szó, a következõ táblázat mindent leír.

Feladatok 2

3. Adatlap

Kérj be adatokat a felhasználótól egy emberr?l, mégpedig a következ?ket:

- születési év
- magasság méterben
- tömeg kilogrammban

Majd írd ki a képerny?re ugyanezeket, szépen formázva, egymás alá rendezve, így nézzen ki (használd a printf-ben a "http://wiki.math.bme.hu\t"http://wiki.math.bme.hu-t arra hogy tab-ot írj ki):

```
Kora:       19     ev
Magassaga:  1.75   m
```

```
 Tomege:     69     kg
```

Válaszd meg jól a változók típusait amiket használsz!

A tört érték kiírásának formázásához segítség: A printf-ben "http://wiki.math.bme.hu%1.8f"http://wiki.math.bme.hu -fel jelölheted hogy egy float típusú változót úgy akarsz kiírni hogy az egész részt egy karakterrel írod ki, a tört részt pedig 8 karakterrel.

4. Tömbök

A tömbökre lehet úgy gondolni, mint a listákra Sage-bõl, de a kettõ közt rengeteg különbség van. Jelenleg csak fix méretû tömbökkel fogunk dolgozni, változó méretûekre késõbb lesz példa.

Egy **n** hosszú tömböt 0-tól **n**-1-ig indexelünk. Az elemeit egyenként lehet módosítani, vagy lekérdezni.

A következõ kód belerakja egy tömbbe 0-tól 19-ig a számok négyzetét:

```
#include<stdio.h>

int main(void) {
    int t[20];     // Itt hozzuk letre a tombot, elore megadjuk a meretet
    int i;
    t[5] = 5 * 5;             // Igy tudunk egy elemenek erteket adni
    t[6] = t[5] + 11;         // Vagy felhasznalni ertekkent
    for(i = 0; i < 20; i++) { // Ez a ciklus vegzi el a munkankat
        t[i] = i * i;
    }
    return 0;
}
```

Írhattunk volna még egy kiírást a végére külön ciklusba, de talán a tömbök használatának alapja már ebbõl átvehetõ.

Kiegészítendõ feladat

Egészítsd ki a kódot a megadott helyeken, hogy a Fibonacci számokat tárolja a 20. tagig, majd a beadott indexû tagot írja ki.

```
#include<stdio.h>

int main(void) {
    /* Hozz létre egy 20 hosszú int tömböt */
    int i;
    int index;

    /* Állítsd be a 0. és 1. indexû tag értékét */
    for(i = 2; i < 20; i++){
        /* Számoljátok ki az i. tagot a korábbi tagokból */
    }

    scanf("%d", &index);
    /* Írd ki a konzolra az indexedik tagot */

    return 0;
}
```

Mi történik ha túl nagy indexet adtok meg?

5. Pi közelítés (bónusz)

Közelítsd a Pi-t, a négyzetszámok reciprokösszege segítségével (ez ugye Pi négyzet per 6). Ha úgy mint az stdio.h-t betöltöd a math.h-t akkor mûködni fog az sqrt függvény, mellyel a gyökvonást megoldhatod.

Valamint, ha már sikerült közelíteni, akkor a 4 * atan(1) kifejezéssel ellenõrizheted magad (ennek elég jól kell becsülnie a Pi-t).