Tartalomjegyzék

C fordítás konzolban

Els? fordítás gcc-vel

Nyiss egy szövegszerkeszt?t, és másold be egy új fájlba a "http://wiki.math.bme.huhello.c"http://wiki.math.bme.hu kódját:

```
#include <stdio.h>
int main() {
    int x = 2;
    printf("hello world! %d\n", x);
    return 0;
}
```

Nyiss egy konzolt és menj abba a könyvtárba (cd paranccsal) ahová elmentetted a "http://wiki.math.bme.huhello.c"http://wiki.math.bme.hu-t.

Fordítsd le, készíts "http://wiki.math.bme.huhello"http://wiki.math.bme.hu nev? futtatható fájlt:

```
gcc -o hello hello.c
```

Futtasd le a kapott programot!

Második fordítás gcc-vel

Mostantól használjuk a gcc warning-kapcsolóit, hogy lássuk a kisebb hibákat, figyelmeztetéseket is!

```
gcc -W -Wall -o hello hello.c
```

Rontsd el a "http://wiki.math.bme.huhello.c"http://wiki.math.bme.hu kódot a következ? módokon, majd mindig mentsd el és fordítsd le újra (és a következ? elrontás el?tt mindig állítsd vissza a helyes verziót):

- törölsz egy pontosvessz?t a sor végér?l
- a printf-es sorban x helyett valami más bet?t írsz
- létrehozol egy y nev? egész típusú változót az x után, de nem használod a kódban
- kitörlöd az utolsó } -t
- törlöd a "http://wiki.math.bme.hureturn 0;"http://wiki.math.bme.hu sort

CodeLite használata

Beállítások

- Indítsd el a CodeLite fejleszt?i környezetet!
- Állítsd át az indentálást (tabulálást/beljebbezést) a *Settings -> Global Editor Preferences* menüpont alatt a *General -> Indentaiton* résznél hogy ne használjon tab karaktereket
- Hozz létre egy új Workspace-et és benne egy új projektet (mindenképp külön könyvtárban: *Create the project under a separate directory* legyen bepipálva), segítség itt az oldal vége felé

Használat

- Minden feladat megoldását új projektbe kell tenni, hogy a *main()* függvények ne ütközzenek (egy projekten belül nem lehet több bel?le).
- A narancssárga az aktív projekt a workspace-en belül. Ha menüb?l kiválasztod a fordítást (*Build -> Build project* vagy F7 billenty?) akkor ez mindig az aktív projektet fogja lefordítani, ha épp másik file van megnyitva a szerkeszt?dben, akkor is! Ugyanígy a futtatásnál is (menüb?l *Build -> Run* vagy Ctrl-F7) az aktív projekt *main()* függvénye fog lefutni!

Feladatok

Minden feladathoz nyiss új projektet a CodeLite-ban!

1. Feltételes print

Egészítsd ki a következ? kódot a megjegyzések helyén!

```
#include<stdio.h>

int main(void) {
    /* x és y változók deklarálása, értékadás */

    printf("X erteke: %d\n", x);
    printf("Y erteke: %d\n", y);
```

```
    printf("\n");

    /* egy feltételes utasítással (if, else if, else) érd el hogy mindig igaz állítás kerüljön a k
    printf("%d nagyobb mint %d!\n", x, y);
    printf("%d nagyobb mint %d!\n", y, x);
    printf("%d ugyanannyi mint %d!\n", x, x);

    return 0;
}
```

Ha kész, módosítsd úgy a kódot, hogy ne legyen beleírva az elején az *x* és az *y* értéke, hanem a program a felhasználótól kérjen be egész számokat, a *scanf()* függvény segítségével! Példa egy *z* egész típusú változó bekérésére:

```
int z;
    printf("Add meg z értékét: ");
    scanf("%d", &z);
    /* innent?l z-nek van értéke (ha a felhasználó tényleg egész számot adott meg) */
```

2. For ciklus

Segítségnek itt egy for ciklus példa: 0-tól 9-ig kiírjuk a számokat, vagyis a ciklusváltozó értékét (általában *i*-nek vagy *j*-nek nevezzük a ciklusváltozót, ami tipikusan minden körben növekszik eggyel, de persze máshogy is lehetne):

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

Egészítsd ki a következ? kódot a megjegyzések helyén! Egy-egy külön változóban legyen a ciklusváltozó kezd?értéke (*start_i*) és utolsó értéke (*end_i*), adj nekik értéket!

A for ciklusban *i* legyen a ciklusváltozó, és az értéke hármasával növekedjen vagy csökkenjen (attól függ?en hogy *start_i* vagy *end_i* a nagyobb).

```
#include<stdio.h>

int main(void) {
    /* start_i és end_i változók deklarálása, értékadás */
    /* i-t is deklarálni kell ! */

    for (/* inic. */ ; /* feltétel */; /* minden ciklusmag végén */) {
        printf("%d\n", i);
    }
    return 0;
}
```

3. Do / while ciklusok

Írj C kódot, ami a felhasználótól egy ciklusban egész számokat kér be addig, amíg 0 értéket nem kap. Ekkor pedig írja ki a képerny?re a kapott nemnulla számok átlagát!

Segítség a nem egész érték? változó (*atlag*) kiírásához:

```
printf("%f", atlag);
```

Oldd meg a feladatot *do-while* (hátultesztel?) és *while* (elöltesztel?) ciklussal is!

```
do {
    /* utasítások amik minden körben lefutnak */
} while (/* feltétel */);

while ( /* feltétel */ ) {
    /* utasítások amik minden körben lefutnak */
}
```

4. Egyszer? gyökkeresés

Egészítsd ki a következ? kódot, amely megkeresi az 5329 négyzetgyökét (lesz neki, és egész)!
(Segítség: nem kell hogy haténkony legyen a kód, elég ha egyesével végigpróbálgatjuk a számokat hogy az-e a gyöke.)

```
#include<stdio.h>

int main(void) {
    int szam = 5329;
    int gyok;
    /* szamolj valamilyen ciklusban, valoszinuleg egy feltetel is kell majd a cikluson belul */

    printf("A %d gyoke %d\n", szam, gyok);
    return 0;
}
```

5. 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");
}
```