Programozási Paradigmák - Zárthelyi Dolgozat - 2012. 12. 22

Feladat

Írjon **assembly** nyelvű programot, amely kiszámítja és kiírja 5 derékszögű háromszög átfogójának a hosszát. Használja a mellékelt *exam.asm* állományt belátása szerint.

Részletek

A fő programrész elején a sys_read rendszerhívás segítségével olvasson be a konzolról egy stringet, célszerűen a stri változóba. Ezt az alfeladatot megoldhatja függvényként is, ha úgy kézenfekvőbb, vagy utasítások sorozataként a fő programrészben. Ezután Írja meg, majd használja a convert függvényt, amely átalakítja a beolvasott stringet egy pozitív egész számmá: ez a szám lesz n. Ehhez semmiféle ellenőrzésre nincs szükség, csak valósítsa meg az alábbi algoritmust:

```
n = 0;
for (int i = 0; i < strlen – 1; i++) {
      n = n * 10;
      n = n + (stri[i] – '0');
}
```

A sys_read rendszerhívás visszaadja a beolvasott string hosszát, amelyben az utolsó karakter az ENTER, emiatt tart eggyel rövidebb ideig a ciklus. Használhat while-t is, ebben az esetben a kilépési feltétel `stri[i] == 13`, ahol a 13 az ENTER karakterkódja. Ezek után írjon egy ciklust a fő programrészben, amely ötször fut le. Minden iterációban hívja meg a következő függvényt:

```
printf(stro, i, a, b, hyp);
```

ahol:

stro a kiíratandó string, ''Triangle %d (a: %.2f mm, b: %.2f mm) hypotenuse: %.2f mm''

i a háromszög sorszáma, 1-től 5-ig, integer

a a háromszög a oldala, i + 3, double

b a háromszög b oldala, n / i, double

hyp az átfogó hossza, double $c = \sqrt{a^2 + b^2}$

Természetesen az aktuális iterációhoz tartozó a, b és hyp értékeket a programnak kell kiszámítania!

A programnak nem kell követnie a hívási konvenciót, kivéve, ahol ez elengedhetetlen.

A mellékelt exam.asm állományt tetszés szerint módosíthatja, bármilyen változót, regisztert utasítást használhat. A feladathoz igény szerint alakíthat ki saját függvényeket.

Fordítás: `nasm exam.asm -felf64`

Link: `gcc exam.o -oexec -no-pie`

Tipp

Haladjon kis lépésekben, csak akkor menjen tovább, ha az adott programrész megfelelően működik!

Egy példa erre: először a ciklus a printf függvénnyel, ahol a paramétereket még nem számítja ki, csak átadja azokat tetszőleges értékekkel. Ha ez működik, akkor kiszámítja a és b paramétereket egy fix n értékkel. Ezután következhet az átfogót számító függvény. Ezután a string átalakító függvény egy fix stringgel, végül pedig a string bekérése. Így feltételezhetően mindig látja, hogy melyik résszel van a gond.