Tudnivalók
az otthon kidolgozandó feladatokról

- Otthon kidolgozandó feladat megoldásának beküldése csak azok számára kötelező, akik fölvették az Assembly programozás konzultáció kurzust. Minden hallgató, aki az általa választott vagy a számára kiosztott otthon kidolgozandó feladat megoldását beküldi, a megoldás minőségétől függően minimum 0, maximum 10 pontot kap.
- A feladat választani az ETR CooSpace moduljának segítségével lehet a http://www.coosp.etr.u-szeged.hu címen **2011. március 10. 8:00-tól**. Bejelentkezni ugyanazzal a felhasználónévvel és jelszóval kell, mint amellyel az ETR-be. Bejelentkezés után az „IB676g-XX Assembly programozás gyakorlat" színteret választva a „Döntés" fejléccel ellátott „Otthon kidolgozandó feladatok – jelentkezés" linkre kattintva érhető el az a felület, melynek segítségével kiválasztható a megoldani kívánt feladat. A választás határideje **2011. április 3. éjfél**.
- Egy programot maximum 2 hallgató választhat gyakorlati csoportonként. A feladatok kiosztása a választás sorrendjében történik automatikusan. Ha már mindkét hely elkelt egy adott feladatra, akkor a CooSpace már nem enged több jelentkezést.
- Amennyiben valaki nem jelöl meg semmit a CooSpace-en, annak a határidő lejártával a gyakorlatvezető fog feladatot kijelölni a még szabad feladatok közül.
- Az otthon kidolgozandó feladat programját Intel 8086/8088 assembly nyelven kell elkészíteni, úgy hogy a kabinetben lévő MASM assemblerrel fordítható legyen.
  Hallgatók közötti kooperáció, külső források (internet, szakirodalom) felhasználása megengedett, de a beadott végeredményt a hallgatónak ismernie kell. Nem elfogadható az a program, amelynek a működését a készítője nem ismeri, nem tudja elmagyarázni annak főbb részeit!
- Az otthon kidolgozandó feladat beküldési határideje: **2011. április 28. (csütörtök) 23:59:59**. A feladatokat szintén a CooSpace-en kell majd beadni az „Időterv" fejléccel ellátott „Otthon kidolgozandó feladat" linken. A CooSpace lehetővé teszi a több alkalommal történő feltöltést is, de a gyakorlatvezetők mindig csak az **utoljára feltöltött** fájlt tudják figyelembe venni!!!

- A választható programok pontos működése helyenként nem teljesen van megfogalmazva. Amennyiben valami nincs konkrétan kikötve, bármilyen technikai megoldás alkalmazható, lényeg hogy a program a várt funkcionalitást biztosítsa.
- A program kódjának tartalmaznia kell a hallgató nevét, csoportját és EHA kódját komment formájában. Szintén komment formájában tartalmaznia kell a program input/output specifikációját (azaz hogy pontosan hogyan használható) és rövid, pár soros leírását.
- Értékelés: a program elfogadásának alapfeltétele, hogy fordítani és futtatni lehessen a kabinetes környezetben MASM assemblerrel. Amennyiben ez nem teljesíthető, az otthon kidolgozandó feladat – és amennyiben a hallgató fölvette az Assembly programozás konzultációt, a kurzus teljesítése is sikertelen. A programot a gyakorlatvezetővel egyeztetett időben és helyen be kell mutatni. A bemutatás során demonstrálni kell a programot futás közben, és röviden el kell tudni mondani annak működését, főbb részeit, szükség szerint a program megvalósításával kapcsolatos kérdésekre válaszolni.
- Kötelező program bemutatás utáni javítására, határidőn túli pótlására nincs lehetőség.

Segítség az otthon kidolgozandó feladatok elkészítéséhez

A http://www.inf.u-szeged.hu/~mate/programok/ címen elérhető PROG2.ASM tartalmaz egy

- a klaviatúráról egy karaktert beolvasó eljárást, amely nem írja ki a karaktert a képernyőre (OLVAS),
- egy karaktert a képernyőre kiíró eljárást (IR),
- egy 0 értékű bájtra végződő szöveget kiíró eljárást (KIIRO),
- 32 bites előjel nélküli szám decimális formában történő kiírását végző eljárást (KI_32).

Beolvasáskor a következőkre kell ügyelni:

- csak a megengedett karaktereket szabad kiírni (pl. szám beolvasásakor csak számjegyeket),
- Backspace (a kódja 08H) esetén az utoljára beírt karaktert törölni kell.

Otthoni kidolgozásra választható feladatok listája

1. Számolás megvalósítása – Összeadó gép
    - Input: a felhasználó billentyűzetről maximum 20 jegyű egész számokat és műveleteket ír be. Sorban egy szám, majd egy művelet. A műveletek: + (összeadás) és – (kivonás), Enter jelzi a formula végét. A program csak az éppen következő inputnak megfelelő karaktereket engedje beírni (vagy számjegyet, vagy műveleti jelet).
    - Output: a beírt számokkal a műveleteket elvégzi, és az eredményt kiírja a képernyőre. A kapott eredménnyel számoljunk tovább egészen addig, amíg a felhasználó ki nem lép.
    - Megjegyzés: Legegyszerűbb, ha úgy dolgozunk, mintha papíron végeznénk a számolást.
2. Számolás megvalósítása nagy számokkal (input, kiírás, megvalósítás) – Szorzás
    - Input: két, maximum 10 jegyű egész számot olvassunk be billentyűzetről és írjuk is ki a képernyőre. A számok végét Enter leütésével jelezheti a felhasználó, de 10 számjegy után automatikusan lépjünk a következő számra.
    - Output: ha megvan a két szám, szorozzuk őket össze és írjuk ki az eredményt a képernyőre.
    - Megjegyzés: Legegyszerűbb, ha úgy dolgozunk, mintha papíron végeznénk a számolást.
3. Anagramma ellenőrzés
    - Input: olvassunk be a billentyűzetről két szót. A szavak végét szóköz jelezze.
    - Output: írjuk ki a két szó alá, hogy anagramma, ha az egyik szó a másik szó betűinek permutációjával előállítható, különben azt írjuk ki, hogy nem anagramma.
4. Hamming kódolás
    - Olvassunk be a billentyűzetről egy négyjegyű hexadecimális számot, és írjuk ki a páros paritású Hamming kódolását binárisan!
5. Bit hibajavítás
    - Olvassunk be a billentyűzetről egy 16 bites bináris szám páros paritású Hamming kódját (21 bites szám)! Állapítsuk meg, hogy hibás-e! Ha hibás, akkor írjuk alá a helyes számot, ha helyes, akkor írjuk ki, hogy helyes!

6. Akasztófajáték
   - Szókitalálós játék előre megadott szókészlettel. A szavakat az adat szegmensben tároljuk, változó hosszúságú 0 értékű byte-ra végződő sztringeket használva. Induláskor a gép sorsolja ki az egyik szót és írjon ki annyi „-” jelet a képernyőre ahány betűből a szó áll. A játékos betűk lenyomásával tippelhet. Ha a szóban van olyan betű, akkor írjuk ki a megfelelő helyre a „-” jel helyére. Ha nincs, akkor növeljük a rossz tippek számát. Ügyeljünk arra, hogy csak betűket vegyünk tippnek, és a kis/nagybetűket ne különböztessük meg! Adott számú rossz tipp után, vagy amennyiben kitalálták a szót, a program írjon ki ennek megfelelő üzenetet és lépjen ki.
7. Prímszámkereső
   - Keressük meg és írjuk ki a képernyőre az összes prímszámot 1 és 10 000 között.
8. Buborékrendezés (sztringek)
   - Input: egy 5 elemű, max. 10 karakter hosszú sztringre mutató pointerből álló tömb az adatszegmensben + a hozzá tartozó sztringek. A sztringek hosszúsága változó és 0 értékű byte jelzi a végüket.
   - Output: készítsük el a tömb rendezett változatát a buborékrendezés algoritmusa szerint. Az eredményt írjuk ki a képernyőre.
9. Gyorsrendezés (számok)
   - Input: egy maximum 10 elemű, 32 bites előjeles egész számokat tartalmazó tömb az adatszegmensben.
   - Output: készítsük el a tömb rendezett változatát a gyorsrendezés algoritmusa szerint. A rendezést ne helyben végezzük el, hanem készítsünk a tömbről másolatot előtte. Az eredményt írjuk ki a képernyőre, vesszővel elválasztva.
10. Betű hisztogramm
   - Számoljuk meg, hogy az input karaktersorozatban melyik betű hányszor fordul elő.
   - Input: A klaviatúráról beolvasott legfeljebb 80 karakter hosszú karaktersorozat.
   - Output: a képernyőre kiírt táblázat tartalmazza az előforduló betűket és az előfordulások számát.

11. Buborékrendezés (sztringek)
    - Input: egy 5 elemű, max. 10 karakter hosszú sztringre mutató pointerből álló tömb az adatszegmensben + a hozzá tartozó sztringek. A sztringek hosszúsága változó, és 0 értékű byte jelzi a végüket.
    - Output: készítsük el a tömb rendezett változatát a buborékrendezés algoritmusa szerint. A rendezést helyben végezzük el. Az eredményt írjuk ki a képernyőre.
12. Törtszámok megvalósítása
    - Valósítsuk meg a 4 alapműveletet valódi törtszámoláshoz. A törteket számláló/nevező alakban tároljuk, egyszerűsítve minden esetben. Egész számokat is törtként x/1 alakban kezelhetjük. Elég, ha számláló és a nevező belefér egy-egy regiszterbe.
    - A műveletek bemutatásához készítsünk kis programot, ami 1-1 példaszámolást végez mindegyik műveletre. Az eredményt számláló/nevező alakban kell kiírni.
13. Számátváltó 2-16 rendszerek között, oda-vissza
    - Készítsünk számrendszerek közötti átváltó programot.
    - Input: számrendszer típusa (2-16 közötti szám), majd egy szám beírása billentyűzetről és végül újból egy számrendszer típusa. Az adatok beírását Enter zárja.
    - Output: az elsőnek megadott számrendszerből váltsuk át a beírt számot a másodikba. Ügyeljünk arra, hogy csak az elsőnek beírt számrendszernek megfelelő számjegyeket engedjünk beírni! Az eredményt írjuk ki a képernyőre.
14. Tökéletes számok
    - Keressük meg és írjuk ki a képernyőre az összes dupla szóban elférő tökéletes számot!
15. Barátságos számok
    - Keressük meg, és írjuk ki a képernyőre az összes dupla szóban elférő barátságos számpárt!
16. Szó ismétlések száma
    - Az input szavait szóközök választják el egymástól. Meg kell számolni, hogy hányszor fordul elő az, hogy egy szó megismétlődik az inputban.
    - Input: A klaviatúráról beolvasott legfeljebb 80 karakter hosszú karakter sorozat.
    - Output: a szó ismétlések száma.

17. Mértani sorozat.
- Állapítsuk meg, hogy egy számsorozat mértani sorozatot alkot-e.
- Input: a billentyűzetről beolvasott előjel nélküli számok sorozata.
- Output: "Mértani sorozat", ha a számok mértani sorozatot alkotnak. "Nem mértani sorozat" egyébként.

18. Rész-string keresés
- Input: A klaviatúráról beolvasott két, legfeljebb 80 karakter hosszú karaktersorozat.
- Output: „Nincs egyezés.” kiírása a képernyőre, ha az első karaktersorozat nem tartalmazza a másodikat, különben egy egész szám, az a pozíció ahonnan kezdve az első sorozatban szerepel a második.

19. Hanoi tornyai
- Input: A klaviatúráról beolvasott három egész szám, az első 1 és 10, a többi 1 és 3 közötti.
- Output: A Hanoi tornyai játék képernyőre kiírt megoldása feltételezve, hogy az első szám a torony magasságát jelöli, a másik kettő pedig a kezdő és a cél pozíciókat.

20. Növekvő részsorozat hossza:
- Határozzuk meg egy számsorozat leghosszabb növekvő részsorozatának a hosszát.
- Input: számsorozat.
- Output: a leghosszabb növekvő részsorozat hossza.