Assembly Programozás
Nagy ZH (valahanyadik). csoport

Általános információk

- A ZH-ra **45 perc + 5 perc** van.
- **FIGYELMESEN OLVASD EL A FELADATOT!**
- A feladat megoldása során választható, hogy az eljárás **ARM vagy Intel x86** assembly-ben kerül implementálásra.
- Az eljárást célszerű egy **megoldas.S** nevű fájlba elkészíteni.
- A **minta.zip**-ben található a C keret alkalmazás amivel lehetőség van tesztelni az eljárást.
- A biro-ra csak az assembly (.S) fájlt kell feltölteni.
- Az assembly-ben írt eljárás neve a feladatleírásban található. Ha nem karakterre pontosan egyezik a megoldás az elvárt névvel akkor 0 pont.
- Figyelni kell arra, hogy a tömbök milyen hosszúak! Ne legyen tömb túlírás!
- Figyelni kell az adatok méretére és előjelességére!
- A gyakorlati és előadás anyagok a `https://biro.inf.u-szeged.hu/kozos/assembly/assembly_anyag.zip` linkel elérhetőek

Fordítási és futtatási segédlet

Intel x86 Linux:

```
$ gcc -m32 -static -g feladat.c megoldas.S -o program
$ ./program
```

ARM:

```
$ arm-linux-gnueabihf-gcc -marm -mcpu=cortex-a7 ↩
   -static -g feladat.c megoldas.S -o program
$ qemu-arm-static ./program
```

Gyakori exit kódok:

- **-8** SIGFPE: floating point exception, aritmetikai számítás hiba.
- **-11** SIGSEG: segmentation fault, tipikusan rossz címzés vagy eljáráshívási konvenciók be nem tartása (helló cdecl!).

Feladat leírás: A következő oldalon!

1. (15 pont) Készítsünk egy **feladatDiver** nevű eljárást aminek három paramétere van. Ezek a paraméterek az alábbiak:

    1. **input**: egy bemeneti 32 bites előjeles egész értékeket tartalmazó tömb. A **0** érték a tömbben az elemek/tömb végét jelöli.
    2. **divisor**: egy 32 bites előjeles egész szám, amivel oszt az eljárás.
    3. **output**: a kimeneti tömb, mely 32 bites előjeles egész értékeket tárol. Előre lefoglalt, nem kell allokálni neki helyet.

    Az eljárás visszatérési értéke egy 32 bites előjeles egész érték ami megadja, hogy hány elemet másolt az eljárás a kimeneti tömbbe.
    Az eljárás végig iterál a bemeneti tömb (**input**) elemein. Amennyiben az adott vizsgált elem **0** akkor az iteráció befejeződött. Minden egyéb esetben megvizsgálja, hogy az adott elem a **divisor** paraméterrel elosztva nulla maradékot eredményez-e. Amennyiben a maradék nem nulla(!), akkor az adott elemet a kimeneti tömbbe másolja.

    A kimeneti tömbbe a feltételnek megfelelő értékek egymás után kell lenniük kihagyás nélkül.

    Az elkészítendő assembly eljárás C-s fejléce:

```
int feladatDiver(int input[], int divisor, int output[]);
```

    A megalósítandó eljárás peszeudó kódja

```
int feladatDiver(int input[], int divisor, int output[]) {
    count = 0;
    for (idx = 0; input[idx] != 0; idx++) {
        value = input[idx];

        if (value % divisor != 0) {
            output[count] = value;
            count++;
        }
    }

    return count;
}
```