Exercise 0 – Intro to x86 assembly

README.1ST:

1. **Please make sure to fill out the course questionnaire as soon as possible.**
2. **Please make sure you have gotten access to your Box.com folder well before you intend to submit your exercise.**
3. **Please read the exercise submission guidelines before submitting your solution.**
4. Please provide any tools, scripts, code, IDA idb files or reversing notes.
5. If you use code/data/etc. from an external source – please be clear in mentioning that source in comments / readme file accompanying your code.

Your task, should you choose to accept it:

1. VM Setup:
    a. Download the VirtualBox software
    b. Go to http://course.cs.tau.ac.il/infosec14/instructions and download the virtual
    c. machine binary.
    d. Unpack it using 'tar zxvf infosec13_VM.tar.gz'.
    e. Open the virtual machine binary using VirtualBox.
    f. Login to your 'student' account with the password 'do or do not there is no try'
    g. Use 'passwd' to change your password. (Now write it down so you don't forget).
    h. Use 'ls –lrt' to view your home directory.
    i. Make sure there is no directory named ~/ex00/. If it exists, get rid of it somehow.
    j. (commands: mv/rm/rmdir).
    k. Download ex00.png from the website
    l. Run the following cmd:
    m. ex_unpack ex00.bin ~/ex00/
    n. There should now be an ~/ex00 directory.
    o. Go into it and find the example.c file (more info in the next segment).

Notes:

For the next questions you will need to write and debug x86 Assembly code.
You should definitely test your code before submitting, but we highly recommend developing code with the compiler and debugger to verify what works and what doesn't along the way.
Please use the example.c file as a template to start your journey a little more quickly.
Helpful commands:
"gcc -ansi -Wall -pedantic -g -masm=intel example.c -o example"
"gdb ./example"

GDB commands can be understood by using the built-in help menus or any GDB cheat-sheets (i.e.: http://goo.gl/9ZZ7AP).

2. Write a small assembly program that factors composite (non-prime) numbers.
   (Number to factor is stored in ebx at the start of execution, result should be placed in eax at the end of execution. Result should be the first (lowest positive) factor or the number itself if no factors were found. All invalid input should return 0).

3. Write a small assembly program that calculates the n-th Fibonacci number
   (N is stored in ebx at the start of execution, result should be stored in eax at the end of execution. All invalid input should return 0).
   Write a non-recursive solution (+10 bonus points if you supply a working recursive solution in addition a working non-recursive solution).

4. Read the following assembly code:

```
MOV ECX, 0
XOR EDX, EDX
label_1:
CMP [ESI], DL
JZ label_2
INC ECX
INC ESI
JMP label_1
label_2:
NOP
```

   a. What are its input arguments? What is its output?
   b. What is the purpose of this small program? (Hint: string representation)