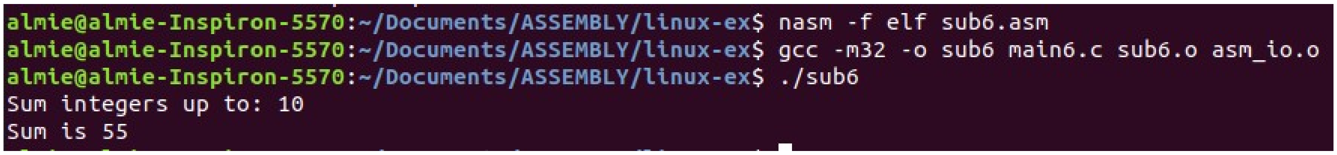
Laboratory Exercise #9

Reading

- Read Section 4.8 of Paul Carter's PC Assembly Book

Practice Exercise:

- Execute “sub6.asm” and interface it with “main6.c”.



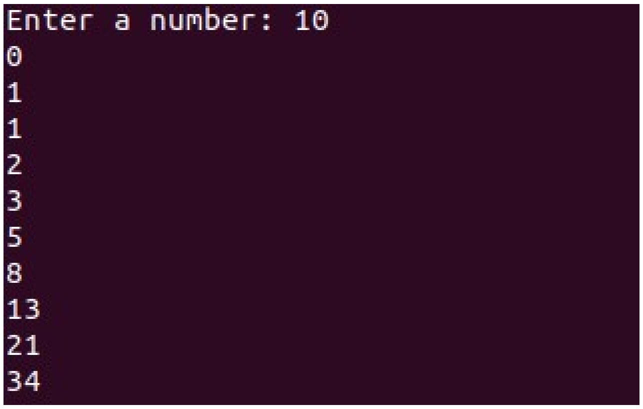
```
almie@almie-Inspiron-5570:~/Documents/ASSEMBLY/linux-ex$ nasm -f elf sub6.asm
almie@almie-Inspiron-5570:~/Documents/ASSEMBLY/linux-ex$ gcc -m32 -o sub6 main6.c sub6.o asm_io.o
almie@almie-Inspiron-5570:~/Documents/ASSEMBLY/linux-ex$ ./sub6
Sum integers up to: 10
Sum is 55
```

- Analyze the sample codes (sub6.asm and main6.c). Reflective questions:

```
   What is the function of sub6.asm? What is/are the stack
register(s) used in the program? What is the purpose of main6.c?
Explain the output of sub6.asm implementing stack.
```

Problem #9.

- Write an assembly program that prints the fibonacci series.
- Create a “fibo.asm” (which computes for each fibonacci numbers in the series) that interface with “main.c” that calls an assembly subprogram (named **fibonnaci**) in “fibo.asm”. (*See the practice exercise as an example.*)
- The output of your program is something like this:



```
Enter a number: 10
0
1
1
2
3
5
8
13
21
34
```

- A good programming practice is to write comments on important line of codes for readability and documentation.

Note: Take a screen record of your working code and make sure to record a video explaining each line of your code as well as showing the correct output of your code. Use screen recorder application in Ubuntu (https://itsfoss.com/best-linux-screen-recorders/) or Windows (https://atomisystems.com/screencasting/record-screen-windows-10/)

Deadline :____

Rubric for Programming Exercises				
Program (50 pts)	**Excellent**	**Good**	**Fair**	**Poor**
Program Execution	Program executes correctly with no syntax or runtime errors (9-10)	Program executes with minor (easily fixed) error (4-8)	Program executes with a major (not easily fixed) error (2-3)	Program does not execute (0-1)
Correct Output	Program displays correct output with no errors (9-10)	Output has minor errors (6-8)	Output has multiple errors (3-5)	Output is incorrect (0-2)
Design of Output	Program displays more than expected (7-8)	Program displays minimally expected output (5-6)	Program does not display the required output (3-4)	Output is poorly designed (0-2)
Design of Logic	Program is logically well-designed (9-10)	Program has slight logic errors that do not significantly affect the results (6-8)	Program has significant logic errors (3-5)	Program is incorrect (0-2)
Standards	Program is stylistically well designed (6-7)	Few inappropriate design choices (i.e., poor variable names, improper indentation) (4-5)	Several inappropriate design choices (i.e., poor variable names, improper indentation) (2-3)	Program is poorly written (0-1)
Documentation	Program is well-documented (5)	Missing one required comment (4)	Missing two or more required comments (2-3)	Most or all documentation missing (0-1)