Laboratory Exercise #1

Reading

- Read Section 2.1 of Paul Carter's PC Assembly Book

Practice Exercise:

- Assemble the assembly code (**math.asm)**. This will create an object file (**math.o**) for math.asm.

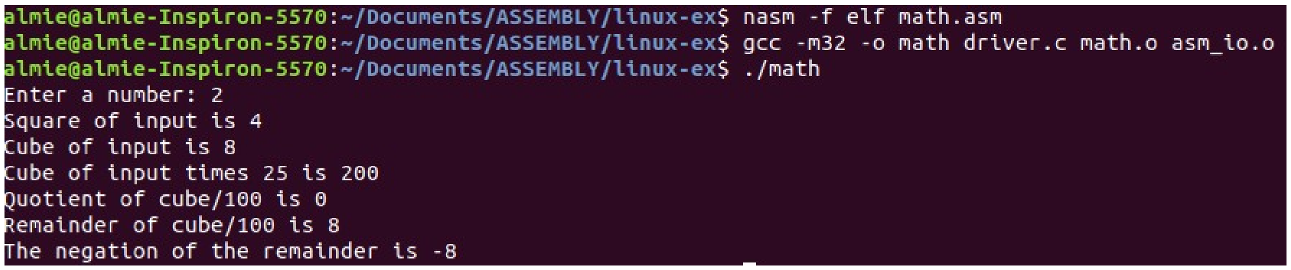
```
   nasm -f elf math.asm
```

- Compile and link the assembly code with the C program (**driver.c**). In our machine, we will be using 32-bit registers thus we specify “-m32”.

```
   gcc -m32 -o math driver.c math.o asm_io
```

- Execute the assembly code.

```
   ./math
```



```
almie@almie-Inspiron-5570:~/Documents/ASSEMBLY/linux-ex$ nasm -f elf math.asm
almie@almie-Inspiron-5570:~/Documents/ASSEMBLY/linux-ex$ gcc -m32 -o math driver.c math.o asm_io.o
almie@almie-Inspiron-5570:~/Documents/ASSEMBLY/linux-ex$ ./math
Enter a number: 2
Square of input is 4
Cube of input is 8
Cube of input times 25 is 200
Quotient of cube/100 is 0
Remainder of cube/100 is 8
The negation of the remainder is -8
```

- Analyze the sample code (math.asm). Reflective questions:

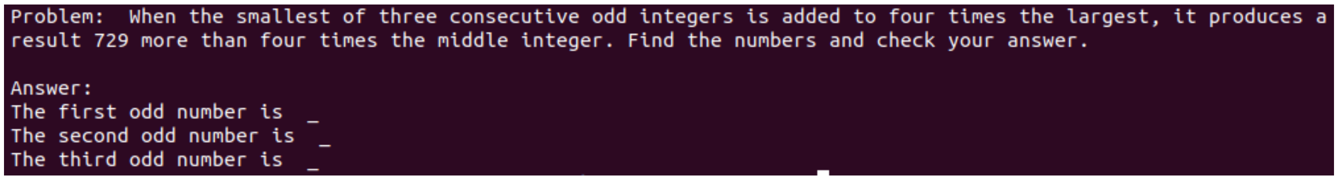
```
   What does the math.asm do? How does “mul” and “imul”
instructions differ from each other?  How does “div” and “idiv”
instructions differ from each other?
```

Problem #1.

- Write an assembly program that solves this problem:

> When the smallest of three consecutive odd integers is added to four times the largest, it produces a result 729 more than four times the middle integer. Find the numbers and check your answer.

- Use different variables to represent these three consecutive odd integers.
- Your assembly program should solve the above problem using arithmetic operations in assembly (*add, sub, mul, div etc.*)
- The output of your program is something like this:



```
Problem:  When the smallest of three consecutive odd integers is added to four times the largest, it produces a
result 729 more than four times the middle integer. Find the numbers and check your answer.

Answer:
The first odd number is  _
The second odd number is  _
The third odd number is  _
```

- The underscore "_" above should be replaced by the correct answer that your program outputs.
- A good programming practice is to write comments on important line of codes for readability and documentation.
- Save your program in a file called surname_lab1.asm. For instance if your surname is "Dela Cruz", submit it as follows:

```
delacruz_lab1.asm
```

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)