

Laboratory Exercise #4

Reading

- Read [Chapter 3 of Paul Carter's PC Assembly Book](#)
-

Practice Exercise:

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

```
nasm -f elf max.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 max driver.c max.o asm_io
```

- Execute the assembly code.

```
./max
```

```
almie@almie-Inspiron-5570:~/Documents/ASSEMBLY/linux-ex$ nasm -f elf max.asm
almie@almie-Inspiron-5570:~/Documents/ASSEMBLY/linux-ex$ gcc -m32 -o max driver.c max.o asm_io.o
almie@almie-Inspiron-5570:~/Documents/ASSEMBLY/linux-ex$ ./max
Enter a number: 10
Enter another number: 4
The larger number is: 10
```

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

```
What does the max.asm do? How does “OR”, “AND” , “XOR”, “NOT” instruction works?
```

Problem #4.

- Write an assembly program that implements the left shift and right shift operations.

<< (Left shift): Takes two numbers, left shifts the bits of the first operand, the second operand decides the number of places to shift. Or in other words left shifting an integer “x” with an integer “y” ($x \ll y$) is equivalent to multiplying x with 2^y (2 raise to power y).

>> (Right shift): Takes two numbers, right shifts the bits of the first operand, the second operand decides the number of places to shift. Similarly right shifting ($x \gg y$) is equivalent to dividing x with 2^y .

- Design a high level programming logic for the above problem and translate it to assembly language
- Use bit operations (*like shifts and bitwise operations*) to solve the problem.
- The output of your program is something like this:

```
Enter a number: 5
Enter the number of places to shift: 1
5 << 1 is 10
5 >> 1 is 2
```

- 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_lab4.asm. For instance if your surname is “Dela Cruz”, submit it as follows:

```
delacruz_lab4.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)

