UNIVERSITY OF THE PHILIPPINES VISAYAS
COLLEGE OF ARTS AND SCIENCES
DIVISION OF PHYSICAL SCIENCES AND MATHEMATICS

CMSC 131
Introduction to Computer Organization and Machine Level Computing
A.Y. 2022 - 2023

Assignment Guide

Prepared by:

Jayvee B. Castañeda
Instructor

ACADEMIC INTEGRITY

As a student of the University of the Philippines, I pledge to act ethically and uphold the value of honor and excellence. I understand that suspected misconduct on given assignments/examinations will be reported to the appropriate office and if established, will result in disciplinary action in accordance with University rules, policies and procedures. I may work with others only to the extent allowed by the Instructor.

Laboratory Exercise #7

Reading

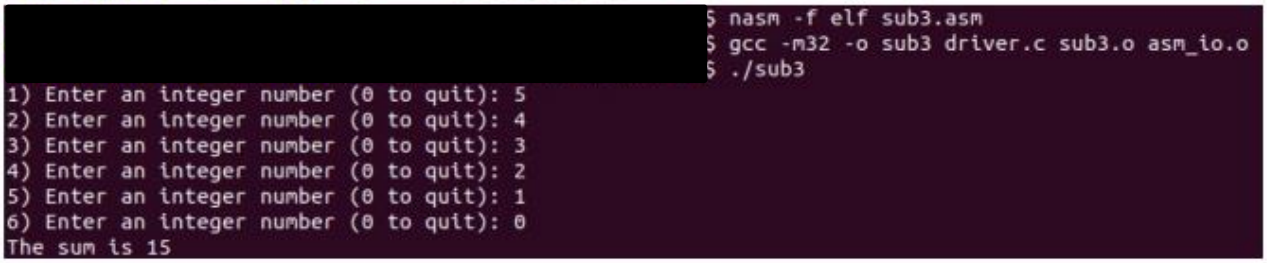
- Read Section 4.6 of Paul Carter's PC Assembly Book

Practice Exercise:

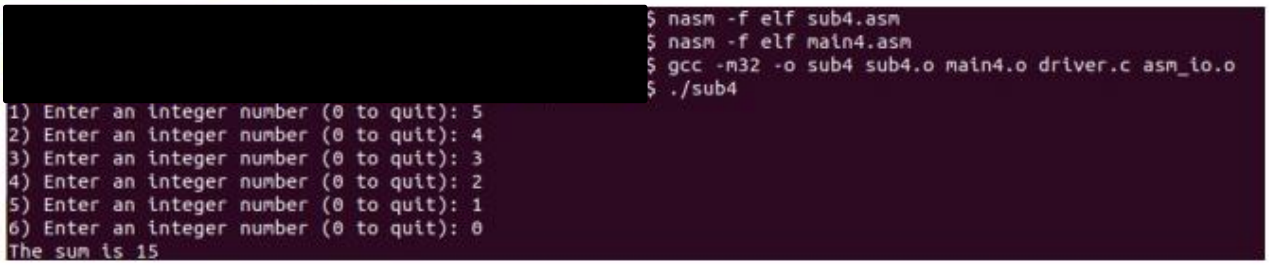
- Compare and contrast **"sub3.asm"** and **"sub4.asm"**. *What is the purpose of sub3.asm? What is the purpose of sub4.asm? Explain the relationship between them.*

- Explore **"main4.asm".** Analyze the sample codes sub3.asm and sub4.asm. *What is the subprogram in the two assembly programs? What is the major difference between them? What are multi-module programs?*

sub3.asm



sub4.asm



Problem #7.

Person 1: After extensive research, I have concluded that 10 is smaller than 5!
Person 2: No, it's not?!
Person 1: It is.

Person 2: Huh? But it's 5, 6, 7, 8, 9, 10. 5 is CLEARLY smaller since it comes before 10 when counting. Huh? Suddenly, I feel a De ja vu!
Person 1: Yes, 5 is smaller than 10, but 10 is smaller than 5!

Person 2: 10 is smaller than 5? I feel like I've said that before.
Person 1: No!
Person 2: …, but 10 is smaller than 5!!! This is too weird.
Person 1: That's too much even.

Person 2: ??????
Person 1: !!!!!!

Person 2: Why does this feel the same? I think I got it, but I'm way too caught up with this repeating time loop instead.

- Write an assembly program that ***computes the factorial of a number,*** however, this time you must **implement a multi-module program** for this problem where you will have **"factorial.asm"** that includes the subprograms **get_int** (responsible for getting an integer from the user) and **factorial** (responsible for solving the factorial of the given number), and another file **"main.asm"** which consists of external subprograms **get_int** and **factorial**.

- Below is an example of a C program of finding a factorial

```
// C program to find factorial of given number
#include <stdio.h>

// Function to find factorial of given number

unsigned int factorial(unsigned int n)
{
    if (n == 0)
        return 1;
    return n * factorial(n - 1);
}

// Driver code

int main()
{
    int num;
    printf("Enter a number to calculate its factorial:",
    scanf("%d", &num);

    printf("Factorial of %d is %d", num, factorial(num));
    return 0;
}
```

- Run the two programs **"factorial.asm"** and **"main.asm"** implementing multi-module programs. You may use your previous lab exercise as a reference.

- The output of your program should be something like this:

```
Enter a number to calculate its factorial: 6

Factorial of 6 is 720
```

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

- Save your program in a file called ***SurnameFirstLetterOfFirstName_lab7.asm*** in camel case. For instance, if your surname is "Juan Dela Cruz", submit it as follows:

DelaCruzJ_lab7.asm

- Take a screen recording 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/)

Submission Requirements:

1. **Program Code ('.asm' file)**
2. **Screen Recorded Defense Video**

DEADLINE: November 28, 2022, 11:59 PM

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)