Name:\_\_\_Joe Baskin\_\_\_\_\_\_\_\_\_\_

**CMIS 310**

**HOMEWORK #7 – Week #7**

This homework is worth 10% of your course grade.

**Read each problem carefully. Failure to follow the instructions for a problem will result in a zero score for that problem.**

Submit the completed Homework via Assignment in LEO.

CHAPTER 8

1. Do Exercise 4 in Chapter 8 (System Software) of Null and Lobur
   1. What is the difference between multiprogramming and multiprocessing? Multiprogramming and multithreading? **Multiprogramming, multiprocessing, and multithreading are all different ways to utilize main memory and CPUs. In Multiprogramming, multiple programs are tasked to a single CPU simultaneously to keep the CPU busy. In Multiprocessing, at least 2 CPUs are used to distribute process execution through completion of the task. Lastly, in multithreading, programs or processes are split in to multiple threads (or chunks) and assigned to the CPU and executed in parallel by the different cores of the CPU.**
2. Do Exercise 11 in Chapter 8 (System Software) of Null and Lobur
   1. Why should assembly language be avoided for general application development? Under what circumstances is assembly language preferred or required? **Assembly language is difficult to learn and understand as is prone to errors. Assembly language is also difficult to maintain when not performed by the original author. Assembly language is and should be used only when there is a specific need to do so, such as writing code to access a specialized device on a computer that there is no operating system service for.**
3. Do Exercise 12 in Chapter 8 (System Software) of Null and Lobur
   1. Under what circumstances would you argue in favor of using assembly language code for developing an application program?  **I would argue to use assembly language if I were to be the sole author of a program that used non-standard protocols in accessing some device that I developed. If, and only if, I was in control of the entire end-to-end process *and* I could not accomplish the task through other programming languages, would I use assembly language.**
4. Do Exercise 13 in Chapter 8 (System Software) of Null and Lobur
   1. What are the advantages of using a compiled language over an interpreted one? Under what circumstances would you choose to use an interpreted language? **Compiled languages convert the source code to machine code directly, when compiled, and can be read by the machine off of the compiled source code whereas interpreted languages are read line for line in the source code. Compiled codes can increase execution speed and have better memory management when executed. Selection of which language to use depends on the application in development. Running simple programs to change files names in bulk can be done simply through an interpreted language, such as Python.**

CHAPTER 9

1. Do Exercise 6 in Chapter 9 (Alternative Architectures) of Null and Lobur
   1. A RISC processor has 152 total registers, with 12 designated as global registers. The register windows each have 6 input registers and 6 output registers. How many local registers are in each register window set? **This question is incomplete is what it is asking for, missing the number of register windows (n). With that, we can calculate the number of local registers (r) as:**

**Therefore, is *n =* 10, then the number of local registers would be *r = 2*.**

1. Do Exercise 33 in Chapter 9 (Alternative Architectures) of Null and Lobur
   1. Indicate whether each of the following applies to CISC or RISC by placing either a C (for CISC) or an R (for RISC) in the blank:
      1. **R** Simple instructions averaging one clock cycle to execute
      2. **C** Single register set
      3. **R** Complexity is in the compiler
      4. **R** Highly pipelined
      5. **C** Any instruction can reference memory
      6. **C** Instructions are interpreted by the microprogram
      7. **R** Fixed length, easily decoded instruction format
      8. **C** Highly specialized, infrequently used instructions
      9. **R** Use of overlapping register windows
      10. **R** Relatively few addressing modes

For problems 7 through 9, consider a CPU that implements **two parallel fetch-execute pipelines for superscalar processing**. Show the performance improvement over scalar pipeline processing and no-pipeline processing, assuming an instruction cycle similar to **figure 4.1 in the Section I B of “Advanced Systems Concepts”**, i.e.:

        a one clock cycle fetch

        a two clock cycle decode

        a three clock cycle execute

and a 50 instruction sequence:

*Show your work.*

1. **o pipelining** would require \_300\_ clock cycles:
   1. Each instruction would require (1 + 2 + 3) = 6 clock cycles to execute, therefore 50 instructions x 6 clock cycles = 300 clock cycles to execute.
2. **A scalar pipeline** would require \_55\_\_ clock cycles:
   1. (1 + 2 + 3) = 6 clock cycles to fill pipeline + 49 (each additional instruction sequence) clock cycles = 55 clock cycles total.
3. **A superscalar pipeline** with two parallel units would require \_\_30\_\_ clock cycles:
   1. (1 + 2 + 3) = 6 clock cycles to fill pipelines in each unit + (48 instructions remaining / 2 parallel units = 24 clock cycles) = 30 clock cycles.

10. a. What is the essential characteristic of the superscalar approach to processor design?

The ability to complete instruction sets faster using multiple parallel processors are the essential characteristic of a superscalar processor.

b. What is the difference between the superscalar and super-pipelined approaches? Superscalar pipelines use parallel pipelines to complete instructions faster while super-pipelined approaches create large pipelines to fill in order to complete instructions faster.