**256 Exam #2 Content:**

**- Language Abstractions**

 - High level / Low level languages

 - Program Execution

 - Translated / Compiled

 - Interpreted

 - Hybrid

 - Assembly Programming

 - Labels

 - Type directives (.word, .space)

 - Opcodes / Operands

 - Register to Register Instructions

 - Direct Addressing Mode

 - Immediate Addressing Mode (#)

 - Implementing Control Structures

 - if, if/else, for, while

 - Indirect Addressing Mode

 - Arrays and References

 - Stack Instructions (PUSH/POP)

 - Function Calling

 - Passing Arguments

 - Return Values

 - Function Implementation

 - Saving / Restoring Registers

 - Accessing passed parameters

 - Calling a function in a function

**- Operating Systems Abstractions** - Definitions

 - Bootstrap process

 - Multiprogramming

 - Mechanisms

 - Time Sharing

 - Timer Interrupts

 - Input/Output Time

 - Synchronization Waits

 - Programs vs Processes

 - Process context

 - Context Switching

 - Process Control Blocks

 - Process States

 - New, Ready, Running, Waiting, Terminated

 - Interrupts and System Calls

 - Sharing & Protection

 - Processor Modes (Kernel, User)

 - Interrupts

 - Device Interrupts

 - System Calls

 - Traps

 - Interrupt Handling Mechanisms

 - Interrupt Service Routines

 - Interrupt Vector

 - System Call Mechanisms

 - System Call Vector

 - Logical Memory

 - Offset register

 - Code / Data / Stack /Heap Segments

 - What goes where.

 - Paged Virtual Memory

 - Pages and Page Frames

 - Page / Offset Addresses

 - Page Table

 - Page Faults

 - Page Replacement

 - Threads

 - Threads vs processes

 - Parallelization

 - Specialization

 - Join

 - Race Conditions

 - Definition

 - Shared variables and critical sections

 - Mutually exclusive execution

 - Locks

 - Threads in Java

 - Thread class and run method

 - start and join

 - synchronized blocks and locks

**Reference:**

*Arithmetic and Logic Instructions:*



*Data Movement Instructions:*



*Branching Instructions:*



*Function Calling Instructions:*

