1. Bits, Data Types, and Operations
   a. What is a datatype?
      i. A classification of data that tells the compiler/interpreter how the programmer intends to use it
      ii. The data held by the compiler/interpreter.
   b. What is 0101 AND 0110?
      i. 0100
      ii. 0101
      iii. 0110
      iv. 1010
   c. What is 0101 OR 0110?
      i. 0111
   d. What is NOT 0101?
      i. 1010
   e. What is 0101 XOR 0110?
      i. 0011
      ii. 1011
   f. What is 0101 NAND 0110?
      i. 0110
      ii. 1011
      iii. 1100
   g. What is 0101 NOR 0110?
      i. 0110
      ii. 1000
      iii. 1100
   h. How would you negate using bitwise operators and the plus sign?
      i. ~x+1
   i. What does this do? (x >> (p+1-n)) & ~(~0 << n);
      i. Gets n bits from position p
      ii. Gets p bits from position n
2. Digital Logic Structures

   a. If you had a decoder with a 5 bit input, how many outputs would it have?
      i. 5
   b. A full adder should consist of
      i. 2 inputs, 2 outputs
      ii. 2 inputs, 3 outputs
      iii. 3 inputs, 2 outputs
      iv. 3 inputs, 3 outputs
   c. Which is a correct sequence according to Gray Code?
      i. 00, 01, 11, 10
      ii. 00, 01, 10, 11
   d. Which is false about simplifying K-Maps
      i. Rows/Columns must be arranged in Gray Code

ii. You must group in multiples of 2

iii. If including a “don’t care”, they must be in multiples of 2

iv. You cannot have a group entirely of “don’t cares”

e. Which is true regarding Combinational Logic?

i. Output depends on an input and a state.

ii. It is used to build memory.

iii. It can use only a combination of the AND, OR, NOT gates.

iv. The same inputs always produce the same output.

3. The Von Neumann Model

a. The Program Counter (PC) holds

i. The count of instructions executed

ii. The address of the next instruction to be executed

iii. The instruction currently being executed

iv. Number of programs completed since the last boot

b. The Instruction Register (IR) holds

i. All legal op codes

ii. The address of the next instruction to be executed

iii. The instruction currently being executed

iv. All data needed to execute the current instruction

4. Introduction to a simple microprocessor, the LC-3

a. Answer True/False - The PC holds the address of the instruction being executed.

i. False, the PC holds the address of the instruction being executed next.

b. Answer True/False - The IR holds the next instruction to execute.

i. False, the IR holds the address of the current instruction being executed.

c. What would be a problem with using an unconditional BR instruction to call a subroutine?

i. It wouldn’t know where to go after subroutine has been called. Using JSR, we can know that the PC value has been stored so we know where the next instruction is from.

ii. It would go directly to HALT

iii. It would start back from first instruction

iv. None of the above

d. What is the function of the LDI instruction?

i. Puts the address represented by some label into memory

ii. Puts the content of some label into memory

iii. Puts the contents of memory location whose address is stored in memory at some label into a register

iv. Puts the content of some label into a register

e. What can we do with a JMP instruction?

i. Branch long distances

ii. Add two numbers

iii. For loop

iv. Multiply two values

f. What does a TRAP instruction perform?

i. IN, OUT, HALT
ii. IN, JMP, HALT
iii. HALT, IN
iv. JMP, IN, HALT

g. What happens if there is no HALT instruction?
i. Nothing, the processor will stop when it runs out of instructions
ii. A memory boundary trap will occur
iii. The processor will start to execute your data
iv. Zombies will try and kill you

h. How would you try to subtract two numbers in LC3?
i. NOT r1, r1; ADD r1, r1, 1; ADD r5, r0, r1
ii. NOT r2, r1; ADD r1, r1, 1; ADD r5, r1, r1

5. Programming in Assembly Language

a. What does the assembler directive .FILL do?
i. Where to put the program
ii. Initialize 7 locations
iii. Initialize one location
iv. Initialize 12 locations

b. What is the point of the labels?
i. Labels are used to symbolically indicate the first address of a particular program
ii. Labels are used to symbolically indicate the address of a particular instruction
iii. Labels are used to symbolically indicate the address of the next instruction
iv. Labels are used to symbolically indicate the content of the next instruction

c. What is true about comments in LC3?
i. Anything on a line after two slashes (//) is considered a comment
ii. Anything on a line after a colon is considered a comment
iii. Anything on a line after a semicolon is considered a comment
iv. Anything on a line after a (#) is considered a comment

d. Why is the LC3 a two pass assembler?
i. First pass - generates a symbol table; Second pass - converts the instructions to machine language
ii. First pass - converts to machine language; Second Pass - scan program file
iii. First pass - converts to machine language; Second Pass - final all labels and calculate the corresponding addresses
iv. First pass - scan program file; Second Pass - converts to machine language

e. Which instruction would ends the program?
i. HALT
ii. .END
iii. STOP

        iv. The program automatically ends

6. I/O 9. TRAP Routines and Subroutines
7. Stacks, programming examples
    a. What is an Interrupt Driven I/O
        i. Modifications to the software to allow for an external device to cause the CPU to stop the current execution a “service” routine
        ii. Modifications to the hardware of the datapath and I/O system and additional software to allow an external device to cause the CPU to stop the current execution and execute a “service” routine
        iii. I don’t know. It sounds too complicated.
    b. When is the usage of polling most appropriate?
        i. When there is a high likelihood of success
        ii. When interrupts aren’t appropriate
        iii. When there are numerous devices
        iv. When there is a high likelihood of failure
8. Introduction to Programming in C
    a. What is the advantage of using a macro instead of a function that performs the same operation?
        i. Macros are shorter and usually require a line of code, whereas a function usually requires more than that. Also, macro executes faster than a function.
        ii. Functions are easier to use and code than macros
        iii. Macros execute slower than a function
        iv. Macros are longer and usually require more lines of code function than a function.
    b. Header files typically contain:
        i. Definitions,declarations,function prototypes
        ii. Definitions, coding statements, functions, prototypes
        iii. Definitions, coding statements, functions, variables
        iv. Functions, variables, prototypes
9. Variables and Operators
    a. What thing(s) does a variable associate?
        i. A symbolic name and a value
        ii. A symbolic name and semicolon
        iii. A symbolic name
        iv. A value
    b. Global variable is visible to?
        i. All functions within a file
        ii. Selected functions within a file
        iii. Only within a function
        iv. Within all files
    c. Static variable is
        i. Local variable inside a function whose value persists throughout the life of the program

ii. Local variable inside a function whose value changes through the life of the program

iii. Local variable inside a function whose value persists throughout that function

iv. None of the above

10. Control Structures

    a. What is true about enumeration constants?

    i. There is no new type created

    ii. There is a new type created

    iii. There is a new value created

    iv. B and C

11. Functions

    a. What does the callee do?

    i. Puts argument onto the stack

    ii. Takes argument form the stack

    iii. Add twos values

    iv. None of the above

12. Debugging

13. Recursion

    a. What do you do to make a call?

    i. Pass parameters, save some caller-saved registers, JMP

    ii. Pass parameters, save some caller-saved registers, JSR

    iii. Pass parameters, save some callee-saved registers, JSR

    b. What do you do when call returns?

    i. Do nothing

    ii. Add values of two registers

    iii. Restore registers, examine return value

    iv. Add to registers

    c. What are caller-saved registers?

    i. Are used to hold temporary values that should be preserved across calls.

    ii. Are used to hold temporary quantities that need not be preserved across calls.

    iii. Are used to hold long-lived values that should be preserved across calls.

    iv. Are used to hold some values that should be preserved across calls.

14. Pointers and Arrays

    a. What is a pointer?

    i. Variable that contains the address and value of another variable.

    ii. Variable that contains the address of another variable.

    iii. Variable that contains the value of another variable.

    iv. Variable that contains the address of itself

15. I/O in C

    a. What is a file pointer?

i. Value returned by fopen and used whenever a stream is called for by a c function
ii. Value and address returned by fopen and used whenever a stream is called for by a c function
iii. Address returned by fopen and used whenever a stream is called for by a C function
iv. None of the above

16. Data Structures in C