The final exam will be comprehensive. You should begin by studying any problems you missed on the midterm exam. If I feel that a question was not well understood at the time of the midterm, there is a good chance that the topic will be revisited on the final!

You should study the end-of-chapter problems from the text – particularly the ones that I list at the end of each section of slides.

Below are just a few questions on some of the most important concepts that we have covered. You can be sure that some of these will appear in some form or another on the final exam. These questions are by no means comprehensive! Other material will be covered – be sure to understand the end of chapter problems and lecture notes. Solutions will be posted shortly.

1. **Subroutines, Functions and the Stack.** Consider the following C/C++ program fragment:

```c
int sub1(int a, int b) {
    int c;
    int d;
    ...
    d = sub2(b, c);
    return d;
}

int sub2(int &x, int &y) {
    int e, f, g;
    ...
    x = y + g;
    return f;
}
```

a. Show the symbol table that the compiler will generate for sub1.

b. Show the stack as it will appear while the machine is executing the region called “more code” in sub1.

c. Show the stack as it will appear just before the `jsr sub2` instruction in sub1.

d. What assembly language code will be generated by the
compiler for the line “return d;” in sub1.

e. Show the stack as it will appear while executing the region labeled “more code” in sub2.

f. Show the assembly language code that will be generated for the line “x = y + g;” in sub2.

g. Show the assembly language code that would be generated just after the jsr sub2 statement in sub1 to clean up after the subroutine call. Assume that the assembler will temporarily store the result of the sub2 call in register R0.
2. **The stack.** Suppose that you knew that a buffer overflow vulnerability existed in the program that prompts for the login password in Microsoft Windows Vista®. Describe in detail how you could use this vulnerability to gain root access to a machine running this OS.

3. **The data path.** Using the reference sheet on the last page of these review problems, list the components of the LC-3 CPU that would be involved in executing all stages, except fetch, of the LC-3 instruction: `LDI R3, VAR1`

4. **The data path.** What is the purpose of the 6-bit line that carries IR[5:0] to the ADDR2MUX in the LC-3 architecture? Give an example of an instruction that uses this line.

5. **Polled I/O.** Suppose we extend the LC-3 by adding a network card. To send a byte of data to the network, you need to place it into the low-order byte of the card’s Network Data Register (NDR), which is memory mapped at address xFFCC. The network card has a Network Status Register (NSR), mapped to address xFFCE. Bit #7 of the NSR is the *ready-to-send* bit (*bit 0 is the least significant bit; bit 15 is the most significant*). When bit #7 is 1, the card is ready to send data. When bit #7 of the NSR is 0, the device is not ready to send, and no data should be written into the NDR. Write a short LC-3 program that uses polling-I/O to send two characters, found in memory at addresses x3100 and x3101, to the network.

![Diagram showing network card connections with NDR at xFFCC and NSR at xFFCE. Bit #7 = RTS is highlighted.]](image)
6. **Interrupts & the stack.** Describe the function of the registers Saved.USP and Saved.SSP in processing interrupts on the LC-3.

7. Suppose that you are asked to write a device driver for a new (interrupt-driven) wireless network card for the LC-3 computer. What are the parts of the device driver? How will your driver interact with the operating system?

8. What is DMA, and how can it cause problems when used in conjunction with a cache memory?

9. **Review.** What is the difference between these two assembler directives?
   
   ```assembly
   VAR1 .FILL x0002
   VAR1 .BLKW 2
   ```

10. **Review.** Suppose I declare an array as follows:

    ```assembly
    ARR .BLKW 10
    ```

    Show three different ways I can access the 4th element of the array in assembly language.
Study Topics and a Few Practice Problems for the Final Exam