Midterm Study Guide

This is simply a guide of topics that I consider important for the midterm. I don't promise to ask you about them all, or about any of these in particular; but I may very well ask you about any of these, as well as anything we discussed in class, in the discussion section, or that is in the programs and handouts.

1. Fundamentals
   a. How to compile a C program and save the results as something other than *a.out*
   b. How to log in to the CSIF
   c. Basic Linux commands: *ls*(1), *ps*(1), *rm*(1), *gcc*(1), *apropos*(1), *man*(1), *vi*(1)/*vim*(1)
2. Structure of a C program
   a. `main`, **`return`**, **`exit`**
   b. Idea of an exit status code
3. Basics of C
   a. Variable names
   b. Keywords
   c. Data types (**`int`**, **`float`**, **`double`**, **`char`**)
   d. Data type modifiers (**`signed`**, **`unsigned`**, **`long`**, **`short`**
   e. How true and false are represented
4. Expressions
   a. Arithmetic operators; precedence, associativity
   b. Increment (**`++`**) and decrement (**`--`**)
   c. Logical operators
   d. Relational operators and Boolean values
   e. Type coercion (char to int, int to float, float to double)
5. Input and output
   a. File pointers, especially stdin, stdout, stderr
   b. Opening and closing files
   c. getchar(), getc(), scanf(), fgets()
   d. putchar(), putc(), printf(), fputs()
6. Statements
   a. Assignments, including **`+=`** and other assignment operators
   b. **`for`** loop
   c. **`while`**, **`do ...while`** loops
   d. **`if`**, **`if ... else`**, **`if ... elif ... else`**, nested **`if`**s
   e. **`switch`** statement
   f. **`continue`**, **`break`**
7. Functions
   a. Declaring and defining functions
   b. Returning a value; **`return`** statement
   c. Parameters and arguments
   d. Scope (local vs. global, etc.)
   e. Static variables in function
8. Functions
   a. Defining them
   b. Parameter lists and how they work
   c. Returning a value; **`return`** statement
   d. Parameters and arguments
   e. Scope (local vs. global, etc.)
   f. Static variables in function

9. Pointers
    a. What it is
    b. Declaring and using pointer variables
    c. Dereferencing (`*`) and taking the address of (`&`)
    d. Use in parameter lists
    e. Array of pointers
10. Arrays
    a. Declaring an array
    b. Referencing an element of the array
    c. Relationship between arrays and pointers
    d. Arrays as function parameters and arguments
11. Strings
    a. What is a string
    b. Array of strings
    c. String functions: `strcpy`, `strcat`, `strcmp`, `n` versions, `strlen`, `strtok`
    d. String to number conversion, number to string conversion: the `sscanf`, `sprintf`, `snprintf`
12. Robust programming
    a. Checking input for validity
    b. Buffer overflows