**Concepts from exam 1 such as:**

* program components, variables, data types and how to choose them, various arithmetic operations, ways to control formatting of output

**Chapter 4**

* **relational operators:** >, <, >=, <=, ==, !=, what can and cannot be compared, how to handle comparison for floating point types
* **bool type:** true, false, boolalpha IO manipulator, the difference between truth values and integers
* **if statement:** conditional execution, else, else if, nested if statements, always use braces
* **logical operators:** && (logical and), || (logical or), ! (unary logical not), operands are Boolean, return value is Boolean
* **truth tables:** used to explain logical operators
* **relops and characters:** use ASCII values for comparison
* **relops and strings:** character by character, can only compare C++ strings, not C-strings
* **conditional operator:** var = x < 0 ? -x : y
* **flag variables:** Boolean variables that store a condition value
* **scope:** of a variable, the region of the program in which the variable exists and can be referred to

**Chapter 5**

* **increment/decrement operators:** ++, --, prefix vs. postfix form, be able to evaluate expressions that use these
* **while loop:** pretest loop, body executes zero or more times, used when you don’t know in advance when the condition will become false
* **do while loop:** posttest loop, will always execute at least once, compare and contrast with regular while loop
* **input validation:** common use of do while loop
* **while loop control with flag**: while (!done) for example
* **for loop:** for (initialization; test; update), understanding the number of loop iterations, used for counting and when the number of iterations is known in advance
* **break and continue:** why their use is so bad
* **counters and accumulators:**in loops, incrementing vs. accumulating, both very useful
* **text file IO:** what text files are, streams for reading and writing, opening streams, using streams, closing streams
* **stream extraction vs. getline:** the first for numeric input, the second for string input

**Chapter 6**

* **functions:** modularization, reuse, return type, parameter list, special void return type
* **function prototypes:** required in real code, and required by clang-llvm compiler
* **parameters:** formal parameters vs. actual parameters (aka parameters vs. arguments), formal: type and name, actual: value (could be variable, expression, literal, etc. . .)
* **naming functions:** functions *do* something so names should contain verbs
* **return statements:** where they can appear, make sure functions always return, never return from the middle of a loop
* **global variables:** never acceptable, global *constants* are acceptable if used in multiple functions
* **local variables:**defined inside functions, scope is function body, parameters are used as pre-initialized local variables
* **pass-by-value parameters:**argument is *copied* into formal parameter when function is called, formal parameter can be used as a variable, but it is a copy of the original
* **pass-by-reference parameters:**reference variable is a reference to another variable, declared using ampersand (&), an *alias* for another variable, changes actually change real variable, arguments must be variables
* **function design:** function should do only one thing