# ****Chapter 1****

1. What is software?
2. Why is software important?
3. Where is software important?
4. What could go wrong if some software fails? List some examples.
5. Where does software play an important role? List some examples.
6. What are some jobs related to software development? List some.
7. What’s the difference between computer science and programming?
8. Where in the design, construction, and use of a ship is software used?
9. What is a server farm?
10. What kinds of queries do you ask online? List some.
11. What are some uses of software in science? List some.
12. What are some uses of software in medicine? List some.
13. What are some uses of software in entertainment? List some.
14. What general properties do we expect from good software?
15. What does a software developer look like?
16. What are the stages of software development?
17. Why can software development be difficult? List some reasons.
18. What are some uses of software that make your life easier?
19. What are some uses of software that make your life more difficult?

# ****Chapter 2****

1. What is the purpose of the “Hello, World!” program?  
   To print the sentence “Hello, World” out on the console.
2. Name the four parts of a function.  
   A return type, a name, a parameter list enclosed in parentheses and a function body enclosed in curly braces.
3. Name a function that must appear in every C++ program.  
   The main function.
4. In the “Hello, World!” program, what is the purpose of the line return 0;?  
   To return 0 if the function body has been successfully completed.
5. What is the purpose of the compiler?  
   To ‘compile’ the source code to machine code which the computer can run.
6. What is the purpose of the #include directive?  
   ‘It instructs the computer to make available (“to include”) facilities from the file after the #include directive.’
7. What does a .h suffix at the end of a file name signify in C++?  
   That the file is a header file.
8. What does the linker do for your program?  
   The linker links object code files (from libraries or local files) together into the executable program.
9. What is the difference between a source file and an object file?  
   Source files are the instruction files that contain the literal c++ code, object files are the compiled files which are platform dependent.
10. What is an IDE and what does it do for you?  
    IDE stands for interactive development environment, and it helps with programming by providing an environment in which to code, build and debug, possibly with syntax coloring etc.
11. If you understand everything in the textbook, why is it necessary to practice?  
    Because practice is not the same as the theory and programming needs practice.

# ****Chapter 3****

1. What is meant by the term *prompt*?  
   Waiting for user input.
2. Which operator do you use to read into a variable?  
   cin >>
3. If you want the user to input an integer value into your program for a variable named number, what are two lines of code you could write to ask the user to do it and to input the value into your program?  
    int number;  
    cout << “Please enter a number:\n”;  
    cin >> number;
4. What is \n called and what purpose does it serve?  
   \n is called a newline, it serves to move the cursor the next line.
5. What terminates input into a string?  
   A whitespace or newline
6. What terminates input into an integer?  
   A whitespace or newline
7. How would you write

cout << "Hello, ";  
cout << first\_name;  
cout << "!\n";

as a single line of code?  
cout << "Hello, " << first\_name << "!\n";

1. What is an object?  
   Some memory that holds a value of a given type.
2. What is a literal?  
   A representation of a value of a certain type (for instance integers, strings or Boolean values).
3. What kinds of literals are there?  
   Chars and strings
4. What is a variable?  
   A named object.
5. What are typical sizes for a char, an int, and a double?  
   char:1 byte , int: 4 bytes, double: 8 bytes.
6. What measures do we use for the size of small entities in memory, such as ints and strings?  
   Bytes (1 byte = 8 bits).
7. What is the difference between = and ==?  
   = is an assignment operator, == is a comparison operator
8. What is a definition?  
   A declaration that sets aside memory for an object.
9. What is an initialization and how does it differ from an assignment?  
   In an initialization you declare the variable; an assignment gives the variables value.
10. What is string concatenation and how do you make it work in C++?  
    Adding strings together to form a larger string, you use the ‘+’ operator for this.
11. Which of the following are legal names in C++? If a name is not legal, why not?  
    Legal ones : This\_litle, This\_little\_pig, This\_1\_is fine, \_this\_is\_ok, MiniMineMine, number  
    Not legal ones: 2\_For\_1\_special (starts with number), latest thing (contains space), the\_$12\_method (contains $, which is not a letter, digit or underscore), correct? (contains question mark)
12. Give five examples of legal names that you shouldn’t use because they are likely to cause confusion.  
    - single character variables unless in loop  
    - names with 1 or l where 1 and l look alike  
    - names starting with an underscore  
    - variable names xx1, xx2 etc
13. What are some good rules for choosing names?  
    Not too short but also not too long, clear what the variable holds, don’t start with underscore (might interfere with system variables)
14. What is type safety and why is it important?  
    Making sure that operations on variables are allowed, and will that the result of the operation is valid. This is important because you might encounter unexpected behavior/results otherwise.
15. Why can conversion from double to int be a bad thing?  
    Because a double can have larger values than an int.
16. Define a rule to help decide if a conversion from one type to another is safe or unsafe.  
    Check the value of the type to see if it fits in the range of the type to be converted to.