Questions

Below are a list of questions with location of the related concepts in the course material.

Location: Setting up your environment --- Lesson 1

Question 1: What IDE are we using?
Answer:
 Block ++

Question 2: How do you set up your programming environment?
Answer:
 Follow the instructions at
Page in study guide: 2

Location: Variable --- Lesson 3

Question 3: What is a variable for?
Answer:
 It allows to store values & reuse them
Question 4: How do you read a variables from the keyboard?
Answer: Using "cin>>"
Question 5: What are valid variable names:
Answer: They must respect these rules
(i) Any combination of numeric characters and letters of the alphabet can be used, except that a name cannot start with a numeric character.

(ii) The only other character that may be used in a name is the underscore character "_". Names may also
 start with "_".

(iii) Certain words like *int* and *return* can't be used. However, a variable can be declared with the name
 cout, *cin* or *endl*.
Page in study guide: Page 21-22

Location: Assignment ---Lesson 4

Question 6: What do you typically use for variable assignment?
Answer:
 "="
Qestion 7: What are the other assignment operators in C++?
Answer: +=, -=, /= , *=, ++ and –.
Page in study guide: 35

Location: Variable Diagram ---Lesson #5

Question 8: What use is a variable diagram?
Answer:

It help get an insight into the inner working of a program. Mainly, the steps the variables go through. They help track difficult bugs.
Page in study guide: 43

Location: Floating point number ---Lesson 6

Question 9: What are floating point number for?
Answer:
They are used to represent fraction: non-whole quantities
Question 10: What is *implicit type conversion* ?

Answer:
It's when a value of a certain type gets converted into a value of another type.
Question 11: What is ex*plicit type conversion* ?

Answer: You use the wanted type in parenthese before the value to be converted to get a value of the wanted type.
Page in study guide: 54

Location: String & character ---Lesson 7

Question 12: What are the C++ type for String & Character?

Answer:
They are respectively string & char
Question 13: What is the difference between string and character?
Answer: Character is a one letter value; while string is a multiple letter value. It's truue a string can also be a one letter value
Question 14: What are '\t' and '\n'?
Answer: The represent tabular space and newline characters
Page in study guide: 72

Location: 'If' statement ---Lesson #8

Question 15: When do you use an 'if' statement?
Answer:
The statement is used to conditionally execute some code.
Question 16: What are the relational operators used by conditionals?

Answer:
<, >, <=, >=, ==, !=
Page in study guide: 86

Location: While loop ---Lesson 9

Question 17: #What use is the while loop?
Answer:
It controls the repeatition of a piece of code based on a condition
Page in study guide: 95

Location: Debugging ---Lesson #10

Question 18: What are the various type of programming error?
Answer:
Syntax error, run-time error and logical error
Page in study guide: 105

Location: Boolean value ---Lesson 11

Question 19: What are the boolean operators corresponding to the logical AND, OR and NOT
Answer: They are respectively: &&, || and !
Question 20: Why is “if (10 < n < 20)” a logical error?
Answer: The program gets evaluated differently from what the condition suggest. See Page 119 for detailed walkthrough.
Question 21: Why is it better to use “y % x” rather than “y/x” in some instance?
Answer: When x = 0, the former won't throw an error; while the latter will throw an error.
Page in study guide: 117; 119

Location: Switch Statement ---Lesson 13

Question 22: When do you use the 'switch' statement?
Answer: A switch statement is appropriate when based on the value of a given variable, a certain piece of code needs to be run. There needs to be more than two potential values for an 'if' statement not to be a better alternative.
Question 23: What role does 'break' play in a 'switch' statement?
Answer: The 'break' keyword interrupt the switch statement, taking the code execution out of the current switch statement.
Page in study guide: 147

Location: Iteration ---Lesson 14

Question 24: What are the iterations structures available in C++?
Answer: “while”, “do...while” and “for”.
Question 25: What should you keep in mind when using a while loop?
Answer: When using a *while* loop, remember to always apply the three rules:

1. The variable(s) that appear in the loop condition (i.e. the loop control variable(s)) must be initialised when the while loop is first encountered.

2. Test the loop control variable(s) in the condition of the loop. The condition must specify the values of the control variable(s) for which the loop must continue repeating, and hence (implicitly) the values for which the loop must terminate.

3. Inside the body of the loop, the value of the loop control variable(s) should be changed to ensure that the loop condition becomes false at some stage.

Question 26: What is a a *counter-driven loop*?
Answer: A loop whose iteration is determined by a variable that keeps count of the number of times the loop is executed.
Question 27: What is an *accumulator*?
Answer: When a loop is used to accumulate the sum of a sequence of values, the variable where the

sum is stored is an accumulator.
Question 28: What is a *sentinel-driven loop*?
Answer: A while loop that executes until a sentinel value is input
Question 29: What is a block?
Answer: A *block* is another name for a statement sequence, i.e. a series of statements enclosed by braces *{* and *}*.
Page in study guide: 155; 156; 157

Location: For loop---Lesson 15

Question 30: When do you use a for loop?
Answer: A "for" loop is appropriate as an alternative to a counter driven loop.
Question 31: When is a while loop more appropriate than a for loop for a counter driven loop?
Answer: A while loop is more appropriate if you don't know beforehand how many iterations will be needed
Page in study guide: 172; 181

Location: Nesting Loop ---Lesson 16

Question 32: What is structure nesting? What programming structure can be nested?
Answer: Structure nesting is when a programming structure appears in another one. If, switch, while, "do ...while" and "for" can all be nested.
Question 33: What does the operator "%" do?
Answer: The % operator gives the remainder when one number is divided by another number.
Page in study guide:

Location: Local and Global variables ---Lesson 19

Question 34: What is the difference between local and global variables?
Answer: Local variables are variables defined within a function; while global variables are variables that are not declared in a function, but are accessible within it.
Question 35: What is the difference between parameters & local variables?
Answer: Parameters have their value passed to the function when the function is called; while local variables are variables defined within a function. Moreover, locally declared variable is inaccessible to other functions.
Question 36: When do you use global variables instead of local variables?
Answer: We should use local rather than global variables when variables belong to specific functions. This prevents functions from inadvertently changing the values of variables needed in other functions.
Question 37: When do you use parameters rather than global variables?
Answer: We should use parameters rather than global variables when more than one function needs access to the same variables.
Page in study guide: 224

Location: Void functions ---Lesson 20

Question 38: What is a void function?
Answer: It's a function that doesn't return anything.
Page in study guide: 235

Location: Reference parameters---Lesson 21

Question 39: When do you use reference parameter?
Answer: Reference parameters are used when we want the function to change values of global variables.
Question 40: How do you identify reference parameters?
Answer: A reference parameter has ampersand symbol “&” between the type and name of the parameter.
Question 41: What difference is there between the name of void function & the name of value-returning function?
Answer: The name of a void function should describe what it does, so the name is generally a verb or contains a verb (e.g. displayRow or calcAverage). The name of a value-returning function should describe the value that it returns, so the name is generally a noun (e.g. average).
Question 42: How do you get a function to return multiple values?
Answer: If a function needs to return more than one value, make it a void function and use two or more reference parameters. Don’t use reference parameters with a value returning function.
Page in study guide: 245

Location: One dimensional array ---Lesson 24

Question 43: When do you use an array?
Answer: We use an array if we have many data values of the same type and that represent the same sort of thing.
Question 44: Why can't array be passed by value?
Answer: Arrays cannot be passed by value because this would be inefficient. Pass-by-value would involve copying the entire array
Question 45: What is the risk of always passing an array parameter by reference?
Answer: The fact that all array parameters are passed by reference can be a problem however, because a function can inadvertently change the values in an array when it shouldn’t.
Page in study guide: 297, 299

Location: Two dimentional array ---Lesson 26

Question 46: What is a two dimentional array?
Answer: A two-dimensional array can be viewed as a one-dimensional array of which the elements are also one-dimensional arrays.
Page in study guide: 318

Location: String manipulation ---Lesson 27

Question 47: How do you access a character in a string?
Answer: We can refer to the individual characters of a string by using a subscript in square brackets after the name of the string object. For example, oneWord[4] refers to the fifth character in the string oneWord. This is because the subscript of a string starts at 0.
Question 48: What are *mutators* & *accessors*?
Answer: We call member functions that change an object mutators and member functions that do not, accessors.
Page in study guide: 332

Location: Structs ---Lesson 28

Question 49: When do you use a struct?
Answer: Use a struct if you have related data (probably not of the same type), especially if you would otherwise have to pass a lot of separate data values to a function.
Page in study guide: 343

Location: Arrays of structs ---Lesson 29

Question 50: When do you use array of structs?
Answer: When you find you are having to define parallel arrays (i.e. arrays with the same number of elements) think whether you can't use a struct to store the data of corresponding elements together.
Page in study guide: 352

Location: Classes ---Lesson 30

Question 51: What is a class?
Answer: A class combines data and functions to operate on that data.
Question 52: What is encapsulation?
Answer: Encapsulation not only refers to this combination of data and functions, but also to the fact that the data is hidden, i.e. inaccessible, to the rest of the program.
Question 53: What is a *constructor*?
Answer: A constructor is a special member function of a class that has the name of the class and has no return type. A constructor is called automatically (implicitly) when an object of the class is declared. In general, a constructor is used to initialise the member variables of an object.
Page in study guide: 356; 357; 368