Capitulo 6

This activity contains 37 questions.

1. *Section 6.2 Program Components in C++*

6.2 Q1: All of the following are true of functions except:

- ◯ *The definition of a function usually is visible to other functions.*
- ◯ *The implementation of a function is hidden from the caller.*
- ◯ *A function call must specify the name and arguments of the function.*
- ◯ *They define specific tasks that can be used at many points in a program.*

2. *6.2 Q2: Functions can:*

- ◯ *Do any of the above.*
- ◯ *Be used as building blocks to create new programs.*
- ◯ *Be reused any number of times.*
- ◯ *Return a result to the caller function.*

3. *Section 6.3 Math Library Functions*

6.3 Q1: All math library functions:

- ◯ *Return data type int.*
- ◯ *Can only be called after creating a math object.*
- ◯ *Must be called by preceding the function name by cmath::.*
- ◯ *Are global functions.*

4. *6.3 Q2: Which of the following expressions returns the trigonometric sine of x?*

- ◯ *sine(x).*
- ◯ *sin(x).*
- ◯ *trig_sine(x).*
- ◯ *trig_sin(x).*

5. *6.3 Q3: Which of the following is not included in ?*

- [ ] *floor.*
- [ ] *log.*
- [ ] *ln.*
- [ ] *pow.*

6. *Section 6.4 Function Definitions with Multiple Parameters*

6.4 Q1: The function prototype double mySqrt(int x);

- ( ) *Declares a function called mySqrt which takes a double as an argument and returns an integer.*
- ( ) *Defines a function called mySqrt which takes an argument of type x and returns a double.*
- ( ) *Declares a function called mySqrt which takes an integer as an argument and returns a double.*
- ( ) *Defines a function called double which calculates square roots.*

7. *6.4 Q2: Using the following function definition, the parameter list is represented by:*

```
A B ( C )
{
   D
}
```

- ( ) *D.*
- ( ) *C.*
- ( ) *B.*
- ( ) *A.*

8. *Section 6.5 Function Prototypes and Argument Coercion*

6.5 Q1: A function prototype does not have to:

- ( ) *Terminate with a semicolon.*
- ( ) *Agree with the function definition.*
- ( ) *Match with all calls to the function.*
- ( ) *Include parameter names.*

9. *6.5 Q2: A function prototype can always be omitted when:*

- *A function is defined before it is first invoked.*
- *A function does not return a value.*
- *A function is invoked before it is first defined.*
- *A function takes no arguments.*

10. *6.5 Q3: Converting from type _____ to type _____ will result in the loss of data.*

- *bool, char.*
- *float, double.*
- *short, long.*
- *int, char.*

11. *Section 6.6 C++ Standard Library Header Files*

6.6 Q1: Each standard library has a corresponding:

- *Cd-rom.*
- *Variable type.*
- *Function.*
- *Header file.*

12. *6.6 Q2: Which of the following C++ Standard Library header files does not contain a C++ Standard Library container class?*

- *<stack>.*
- *<vector>.*
- *<string>.*
- *<list>.*

13. *Section 6.7 Case Study: Random Number Generation*

6.7 Q1: The rand function generates a data value of the type:

- *int.*
- *unsigned int.*
- *long int.*
- *short int.*

14. *6.7 Q2: A variable that can only have values in the range 0 to 65535 is a:*

- *Four-byte int.*
- *Two-byte int.*
- *Four-byte unsigned int.*
- *Two-byte unsigned int.*

15. *6.7 Q3: In the expression n = x + rand() % y;*

- *y is the scaling value.*
- *x is the scaling value.*
- *Both (a) and (b).*
- *y is the shifting value.*

16. *6.7 Q4: srand:*

- *Should be called before each call to rand.*
- *Should be used instead of rand to generate truly random numbers.*
- *Is unnecessary in C++.*
- *Can use the time function's return value as an optimal seed value.*

17. *Section 6.8 Case Study: Game of Chance and Introducing enum*

6.8 Q1: Enumeration constants:

- *Must have unique identifiers.*
- *Can be assigned other values once they have been defined.*
- *Must have unique integer values.*
- *Are declared using the keyword const.*

18. *Section 6.9 Storage Classes*

6.9 Q1: An identifier's storage class:

- *All of the above.*
- *Determines the period during which that identifier exists in memory.*

- *Determines where the identifier can be referenced in a program.*
- *Determines whether an identifier is known only in the current source file or in any source file with proper declarations.*

19. *6.9 Q2: Depending on the circumstances, the compiler may ignore the storage class specifier:*

- *static.*
- *extern.*
- *register.*
- *auto.*

20. *6.9 Q3: Which of the following is not true of static local variables?*

- *They can be of type int.*
- *They are initialized to zero if not explicitly initialized by the programmer.*
- *They are accessible outside of the function in which they are defined.*
- *They retain their values when the function in which they are defined terminates.*

21. *Section 6.10 Scope Rules*

6.10 Q1: Labels are the only identifiers with:

- *Function-prototype scope.*
- *File scope.*
- *Block scope.*
- *Function scope.*

22. *6.10 Q2: The only identifiers that can be reused elsewhere in a program without any ambiguity are:*

- *Those in the parameter list of a function prototype.*
- *Global variables.*
- *static local variables.*
- *Those in the parameter list of a function definition.*

23. *Section 6.11 Function Call Stack and Activation Records*

6.11 Q1: An activation record will be popped off the function call stack whenever:

- *A function returns control to its caller.*
- *A function declares a local variable.*
- *A function calls itself.*
- *A function calls another function.*

24. *6.11 Q2: Which of the following is not included in a function's activation record?*

- *Local variables it has declared.*
- *The name of the function.*
- *The return address of its caller function.*
- *Parameter values received from its caller.*

25. *Section 6.12 Functions with Empty Parameter Lists*

6.12 Q1: Which of the following is false about the following function prototype?

```
void functionA( void );
```

- *It could have been written void functionA( );.*
- *It does not return a value.*
- *It does not receive any arguments.*
- *It could have been written functionA(void);.*

26. *Section 6.13 Inline Functions*

6.13 Q1: The inline keyword:

- *Increases function-call overhead.*
- *Can reduce a function's execution time but increase program size.*
- *Can decrease program size but increase the function's execution time.*
- *Should be used with all frequently used functions.*

27. *Section 6.14 References and Reference Parameters*

6.14 Q1: When an argument is passed-by-value, changes in the

calling function _____ affect the original variable's value; when an argument is passed call-by-reference, changes _____ affect the original variable's value.

- *Do not, do.*
- *Do not, do not.*
- *Do, do not.*
- *Do, do.*

28. *6.14 Q2: A reference parameter:*

- *Cannot be modified.*
- *Is declared by following the parameter's type in the function prototype by an ampersand (&).*
- *Is an alias for its corresponding argument.*
- *Both (a) and (b).*

29. *6.14 Q3: Call-by-reference can achieve the security of call-by-value when:*

- *The const qualifier is used.*
- *A pointer to the argument is used.*
- *The value being passed is small.*
- *A large argument is passed in order to improve performance.*

30. *Section 6.15 Default Arguments*

6.15 Q1: In regards to default arguments, which of the following is false?

- *They must be the rightmost (trailing) arguments in a function's parameter list.*
- *Default values cannot be global variables or function calls.*
- *Default values can be constants.*
- *When an argument is omitted in a function call, the default value of that argument is automatically inserted by the compiler and passed in the function call.*

31. *6.15 Q2: If the function int volume(int x = 1, int y = 1, int z = 1); is called by the expression volume(3), how many default arguments*

are used?

- *One.*
- *Two.*
- *Three.*
- *None.*

32. *Section 6.16 Unary Scope Resolution Operator*

6.16 Q1: The unary scope resolution operator is used:

- *To access a global variable when it is out of scope.*
- *To access any variable in an outer block when a local variable of the same name is in scope.*
- *To access a global variable when a local variable of the same name is in scope.*
- *To access a local variable with the same name as a global variable.*

33. *Section 6.17 Function Overloading*

6.17 Q1: Which of the following does the C++ compiler not examine in order to select the proper overloaded function to call?

- *The return type of the function.*
- *b. The number of arguments in the function call.*
- *d. It examines all of the above.*
- *Types and order of the arguments in the function call.*

34. *Section 6.18 Function Templates*

6.18 Q1: If a function's program logic and operations are identical for each data type it could receive as argument(s) then a _____ should be used.

- *Overloaded function.*
- *Recursive function.*
- *Macro.*
- *Function template.*

35. *Section 6.19 Recursion*

6.19 Q1: A recursive function is a function that:

- *Takes 3 arguments.*
- *Is inside of another function.*
- *Returns a double.*
- *Calls itself, directly or indirectly.*

36. *Section 6.20 Example Using Recursion: Fibonacci Series*

6.20 Q1: Assuming the following pseudocode for the Fibonacci series, what is the value of the 5th Fibonacci number (fibonacci (5))?

```
fibonacci( 0 ) = 0
fibonacci( 1 ) = 1
fibonacci( n ) = fibonacci( n - 1 ) + fibonacci( n - 2 )
```

- *d. 7.*
- *5.*
- *1.*
- *3.*

37. *Section 6.21 Recursion vs. Iteration*

6.21 Q1: Recursion is to the base case as iteration is to what:

- *Failure of the loop continuation test.*
- *A selection structure.*
- *A repetition structure.*
- *The counter.*

Clear Answers / Start Over Submit Answers for Grading

Some questions in this exercise may have more than one correct answer. To answer such questions correctly, you must select all the correct answers. Also note that answer choices in this exercise appear in a different order each time the page is loaded.