Comprehension Check

1. A(n) _____ allows you to treat a related group of statements as a single unit.

   a. statement

   b. variable

   c. function

   d. event

2. Functions must contain parameters. True or False?

3. Explain how to use a return statement to return a value to a statement that calls a function.

   A return statement is a statement that returns a value to the statement that called the function. To use a return statement, you use the return keyword with the variable or value you want to send to the calling statement.

4. A variable that is declared outside a function in a code declaration block is called a(n) _____ variable.

   a. local

   b. class

   c. program

   d. global

5. When a program contains a global variable and a local variable with the same name, the local variable takes precedence when its function is called. True or False?

6. How can you declare a global variable? (Choose all that apply.)

   a. By declaring the variable outside of a function

   b. By declaring the variable anywhere in your script section with the `global` keyword

   c. By declaring the variable inside of a function without the `var` keyword

   d. By declaring the variable in a function named `global`

7. Explain the concept of data types.

   Variables can contain many different kinds of values—for example, the time of day, a dollar amount, or a person's name. A data type is the specific category of information that a

variable contains. The concept of data types is often difficult for beginning programmers to grasp because in real life you don't often distinguish among different types of information. If someone asks you for your name, your age, or the current time, you don't usually stop to consider that your name is a text string and that your age and the current time are numbers. However, a variable's specific data type is very important in programming because the data type helps determine how much memory the computer allocates for the data stored in the variable. The data type also governs the kinds of operations that can be performed on a variable.

8. JavaScript is a strongly typed programming language. True or False?

9. Explain the purpose of the `null` data type.

The `null` value is a data type as well as a value that can be assigned to a variable. Assigning the value "null" to a variable indicates the variable does not contain a usable value. A variable with a value of "null" has a value assigned to it—null is really the value "no value." You assign the "null" value to a variable when you want to ensure that the variable does not contain any data. In contrast, an `undefined` variable is a variable that has never had a value assigned to it, has not been declared, or does not exist. The value `undefined` indicates that the variable has never been assigned a value—not even the `null` value. One use for an `undefined` variable is to determine whether a value is being used by another part of your script.

10. Which of the following values are integers? (Choose all that apply.)

a. 1

b. 1.1

c. 4e12

d. -10

11. Which of the following values are floating-point numbers?? (Choose all that apply.)

a. 3.0e5

b. .78

c. 1,385,456,200

d. -976,345

12. Which of following values can be assigned to a Boolean variable? (Choose all that apply.)

a. 0

b. 1

c. true

d. false

13. If you attempt to use a Boolean variable of false in a mathematical operation, JavaScript converts the variable to an integer value of 0. True or False?

14. The concatenation operator (+) is used for _____. (Choose all that apply.)

a. adding numbers

b. combining text strings

c. combining variables

d. incrementing numeric variables

15. Which of the following is the correct syntax for including double quotation and single marks within a string that is already surrounded by double quotation marks?

a. `"Shaquille \"Shaq\" O\'Neal is a basketball player."`

b. `"Shaquille "Shaq" O\'Neal is a basketball player."`

c. `"Shaquille /"Shaq/" O/'Neal is a basketball player."`

d. `"Shaquille ""Shaq"" O''Neal is a basketball player."`

16. Explain the difference between unary and binary operators.

A binary operator requires an operand before and after the operator. The equal sign in the statement `myNumber = 100;` is an example of a binary operator. A unary operator requires a single operand either before or after the operator. For example, the increment operator (`++`), an arithmetic operator, is used for increasing an operand by a value of one. The statement `myNumber++;` changes the value of the `myNumber` variable to "101".

17. Explain how to use the conditional operator.

The conditional operator executes one of two expressions, based on the results of a conditional expression. The syntax for the conditional operator is *`conditional expression ? expression1: expression2;`*. If the conditional expression evaluates to true, then `expression1` executes. If the conditional expression evaluates to false, then `expression2` executes.

18. Which of the following characters separates expressions in the conditional expression used with a conditional operator?

    a. ?

    b. :

    c. ;

    d. &&

19. The Or (`||`) operator returns true if _____. (Choose all that apply.)

    a. the left operand and right operand both return a value of true

    b. the left operand returns a value of true

    c. the left operand and right operand both return a value of false

    d. the right operand returns a value of true

20. Which of the following expressions returns a value of 56?

    a. 7 * (3 + 5)

    b. (7 * 3) + 5

    c. (7 * 3 + 5)

    d. 3 + 5 * 7

Short Quiz 1

1. What is the difference between arguments and parameters?

   The variables or values that you place in the parentheses of the function call statement are called arguments or actual parameters. Sending arguments to the parameters of a called function is called passing arguments. When you pass arguments to a function, the value of each argument is then assigned to the value of the corresponding parameter in the function definition. Parameters are placed within the parentheses that follow a function name. A parameter is a variable that is used within a function. Placing a parameter name within the parentheses of a function definition is the equivalent of declaring a new variable. However, you do not need to include the `var` keyword.

2. How do you execute a function?

   A function definition does not execute automatically. Creating a function definition only names the function, specifies its parameters, and organizes the statements it will execute. To execute a function, you must invoke, or call, it from elsewhere in your program. The code that calls a function is referred to as a function call and consists of the function name followed by parentheses, which in turn contain any variables or values to be assigned to the function parameters.

3. Why would you want to return a value from a function?

   In many instances, you may want your program to receive the results from a called function and then use those results in other code. For instance, consider a function that calculates the average of a series of numbers that are passed to it as arguments. Such a function would be useless if your program could not print or use the result elsewhere. As another example, suppose that you have created a function that simply prints the name of a student. Now suppose that you want to alter the program so that it uses the student name in another section of code. You can return a value from a function to a calling statement by assigning the calling statement to a variable.

4. What is variable scope?

When you use a variable in a JavaScript program, particularly a complex JavaScript program, you need to be aware of the variable's scope—that is, you need to think about where in your program a declared variable can be used. A variable's scope can be either global or local. A global variable is one that is declared outside a function and is available to all parts of your program. A local variable is declared inside a function and is only available within the function in which it is declared. Local variables cease to exist when the function ends. If you attempt to use a local variable outside the function in which it is declared, you will receive an error message.

Short Quiz 2

1. What is the difference between loosely typed and strongly typed programming languages?

   Many programming languages require that you declare the type of data that a variable contains. Programming languages that require you to declare the data types of variables are called strongly typed programming languages. Strong typing is also known as static typing, because data types do not change after they have been declared. Programming languages that do not require you to declare the data types of variables are called loosely typed programming languages. Loose typing is also known as dynamic typing because data types can change after they have been declared. JavaScript is a loosely typed programming language. In JavaScript, you are not required to declare the data type of variables in JavaScript and, in fact, are not allowed to do so. Instead, the JavaScript interpreter automatically determines what type of data is stored in a variable and assigns the variable's data type accordingly.

2. Explain exponential notation.

   Exponential notation, or scientific notation, is a shortened format for writing very large numbers or numbers with many decimal places. Numbers written in exponential notation are represented by a value between 1 and 10 multiplied by 10 raised to some power. The value of 10 is written with an uppercase or lowercase *E*. For example, the number 200,000,000,000 can be written in exponential notation as 2.0e11, which means "two times

ten to the eleventh power." Floating-point values in JavaScript range from approximately ±1.7976931348623157 x 10^{308} to ± 5 x 10^{-324}.

3. What are Boolean values and how do you use them?

A Boolean value is a logical value of true or false. You can also think of a Boolean value as being yes or no, or on or off. Boolean values are most often used for deciding which parts of a program should execute and for comparing data. In JavaScript programming, you can only use the words true and false to indicate Boolean values. In other programming languages, you can use the integer values of 1 and 0 to indicate Boolean values of true and false—1 indicates true and 0 indicates false. JavaScript converts the values true and false to the integers 1 and 0 when necessary. For example, when you attempt to use a Boolean variable of true in a mathematical operation, JavaScript converts the variable to an integer value of 1.

4. Explain how to use the concatenation and compound assignment operators with strings.

JavaScript has two operators that can be used with strings: + and +=. When used with strings, the plus sign is known as the concatenation operator. The **concatenation operator** (+) is used to combine two strings. For example, the following code combines a string variable and a literal string, and assigns the new value to another variable:

```
var destination = "Honolulu";

var location = "Hawaii";

destination = destination + " is in " + location;
```

The combined value of the `location` variable and the string literal that is assigned to the destination variable is "Honolulu is in Hawaii."

You can also use the compound assignment operator (+=) to combine two strings. The following code combines the two text strings, but without using the `location` variable:

```
var destination = "Honolulu";
```

```
destination += " is in Hawaii";
```

Note that the same symbol—a plus sign—serves as the concatenation operator and the addition operator. When used with numbers or variables containing numbers, expressions using the concatenation operator return the sum of the two numbers. As you learned earlier in this chapter, if you use the concatenation operator with a string value and a number value, the string value and the number value are combined into a new string value, as in the following example:

```
var textString = "The legal voting age is ";

var votingAge = 18;

newString = textString + votingAge;
```

5. What are escape characters and escape sequences?

An escape character tells the compiler or interpreter that the character that follows it has a special purpose. In JavaScript, the escape character is the backslash `\`. Placing a backslash in front of an apostrophe tells the JavaScript interpreter that the apostrophe is to be treated as a regular keyboard character, such as "a", "b", "1", or "2", and not as part of a single quotation mark pair that encloses a text string. You can also use the escape character in combination with other characters to insert a special character into a string. When you combine the escape character with other characters, the combination is called an escape sequence. The backslash followed by an apostrophe `\'` and the backslash followed by a double quotation mark `\"` are both examples of escape sequences.

Short Quiz 3

1. What is the difference between division (`/`) operator and the modulus (`%`) operator?

The division operator performs a standard mathematical division operation. For example, dividing 15 by 6 results in a value of 2.5. By contrast, the modulus operator returns the remainder that results from the division of two integers.

2. How do you use prefix and postfix operators?

The increment (++) and decrement (--) unary operators can be used as prefix or postfix operators. A prefix operator is placed before a variable. A postfix operator is placed after a variable. The statements `++count;` and `count++;` both increase the `count` variable by one. However, the two statements return different values. When you use the increment operator as a prefix operator, the value of the operand is returned *after* it is increased by a value of one. When you use the increment operator as a postfix operator, the value of the operand is returned *before* it is increased by a value of one. Similarly, when you use the decrement operator as a prefix operator, the value of the operand is returned *after* it is decreased by a value of one, and when you use the decrement operator as a postfix operator, the value of the operand is returned *before* it is decreased by a value of one. If you intend to assign the incremented or decremented value to another variable, then whether you use the prefix or postfix operator makes a difference.

3. Explain how to use the += compound addition assignment operator.

You can use the += compound addition assignment operator to combine two strings as well as to add numbers. In the case of strings, the string on the left side of the operator is combined with the string on the right side of the operator, and the new value is assigned to the left operator. Before combining operands, the JavaScript interpreter attempts to convert a nonnumeric operand, such as a string, to a number. If a nonnumeric operand cannot be converted to a number, you receive a value of "NaN". The value "NaN" stands for "Not a Number" and is returned when a mathematical operation does not result in a numerical value.

4. Explain how the JavaScript interpreter compares nonnumeric values.

When two nonnumeric values are used as operands, the JavaScript interpreter compares them in alphabetical order. The statement `arithmeticValue = "b" > "a";` returns true because the letter *b* is alphabetically greater than the letter *a*. When one operand is a number and the other is a string, the JavaScript interpreter attempts to convert the string value to a number. If the string value cannot be converted to a number, a value of false is

returned. For example, the statement `arithmeticValue = 10 == "ten";` returns a value of false because the JavaScript interpreter cannot convert the string "ten" to a number.

5. Explain how to use logical operators.

Logical operators are used for comparing two Boolean operands for equality. For example, a script for an automobile insurance company may need to determine whether a customer is male *and* under 21 in order to determine the correct insurance quote. As with comparison operators, a Boolean value of true or false is returned after two operands are compared. The Or (`||`) and the And (`&&`) operators are binary operators (requiring two operands), whereas the Not (`!`) operator is a unary operator (requiring a single operand). Logical operators are often used with comparison operators to evaluate expressions, allowing you to combine the results of several expressions into a single statement. For example, the And (`&&`) operator is used for determining whether two operands return an equivalent value. The operands themselves are often expressions. The Not (`!`) operator returns true if an operand evaluates to false and returns false if an operand evaluates to true

Short Quiz 4

1. What is associativity and how does it affect operator precedence?

Associativity is the order in which operators of equal precedence execute. As an example of how associativity is evaluated, consider the multiplication and division operators. These operators have an associativity of left to right. Thus the expression 30 / 5 * 2 results in a value of 12. Although the multiplication and division operators have equal precedence, the division operation executes first due to the left to right associativity of both operators.

2. Which operator has the highest level of associativity?

The . [] () and new operators have the highest level of precedence.

3. Which operators have the lowest level of associativity?

The comma (,) has the lowest level of precedence.