Name:_____ Class:_____ Date:_____

Chapter 02: Using Data

True / False

1. A variable can hold more than one value at a time.
   a. True
   b. False

ANSWER: False
POINTS: 1
REFERENCES: 54

2. The **`int`** data type is the most commonly used integer type.
   a. True
   b. False

ANSWER: True
POINTS: 1
REFERENCES: 64

3. Multiplication, division, and remainder always take place after addition or subtraction in an expression.
   a. True
   b. False

ANSWER: False
POINTS: 1
REFERENCES: 95

4. The term *parse* means to break into component parts.
   a. True
   b. False

ANSWER: True
POINTS: 1
REFERENCES: 90

5. You can create a confirm dialog box with five arguments.
   a. True
   b. False

ANSWER: True
POINTS: 1
REFERENCES: 92

6. Once a variable has been declared and initialized, new values may not be assigned to the variable.
   a. True
   b. False

ANSWER: False
POINTS: 1
REFERENCES: 55

Chapter 02: Using Data

7. The expression `boolean isTenLarger = (10 < 5)` will produce a value of `true`.
   a. True
   b. False
ANSWER: False
POINTS: 1
REFERENCES: 70

8. Even if a statement occupies multiple lines, the statement is not complete until the semicolon is reached.
   a. True
   b. False
ANSWER: True
POINTS: 1
REFERENCES: 56

9. You are limited to declaring a maximum of three variables in a single statement.
   a. True
   b. False
ANSWER: False
POINTS: 1
REFERENCES: 56

10. The **`byte`** and **`short`** data types occupy less memory and can hold only smaller values.
   a. True
   b. False
ANSWER: True
POINTS: 1
REFERENCES: 64

Multiple Choice

11. A data item is ____ when it cannot be changed while a program is running.
   a. variable b. constant
   c. primitive d. literal
ANSWER: b
POINTS: 1
REFERENCES: 54

12. A ____ is a named memory location that you can use to store a value.
   a. cast b. variable
   c. reference d. primitive
ANSWER: b
POINTS: 1
REFERENCES: 54

Chapter 02: Using Data

13. Primitive types serve as the building blocks for more complex data types, called ____ types.
    a. integer b. literal
    c. reference d. data
ANSWER: c
POINTS: 1
REFERENCES: 54

14. ____ refers to the order in which values are used with operators.
    a. Associativity b. Initialization
    c. Declaration d. Floating
ANSWER: a
POINTS: 1
REFERENCES: 55

15. In Java, you use variables of type ____ to store integers, or whole numbers.
    a. `num` b. `double`
    c. `var` d. `int`
ANSWER: d
POINTS: 1
REFERENCES: 54, 64

16. A(n) ____ variable can hold only one of two values: `true` or `false`.
    a. `integer` b. `boolean`
    c. `true` d. `comparison`
ANSWER: b
POINTS: 1
REFERENCES: 70

17. The term ____ refers to the mathematical accuracy of a value.
    a. float data b. real integers
    c. significant digits d. single-precision floating-point number
ANSWER: c
POINTS: 1
REFERENCES: 71

18. A ____ data type can hold 14 or 15 significant digits of accuracy.
    a. `double` b. `float`
    c. `char` d. `boolean`
ANSWER: a
POINTS: 1
REFERENCES: 71

19. You use the ____ data type to hold any single character.
    a. `single` b. `char`

c. `byte` d. `float`

ANSWER: b
POINTS: 1
REFERENCES: 72

20. In Java, ____ is a built-in class that provides you with the means for storing and manipulating character strings.
a. `Escape` b. `Type`
c. `String` d. `Character`

ANSWER: c
POINTS: 1
REFERENCES: 74

21. You can store any character, including nonprinting characters such as a backspace or a tab, in a(n) ____ variable.
a. `int` b. `char`
c. `boolean` d. `set`

ANSWER: b
POINTS: 1
REFERENCES: 75

22. The characters ____ move the cursor to the next line when used within a `println()` statement.
a. `/n` b. `\n`
c. `.+` d. `$`

ANSWER: b
POINTS: 1
REFERENCES: 75

23. In Java, when a numeric variable is concatenated to a `String` using the ____, the entire expression becomes a `String`.
a. plus sign b. equal sign
c. concatenate statement d. string statement

ANSWER: a
POINTS: 1
REFERENCES: 58

24. You use ____ operators to perform calculations with values in your programs.
a. calculation b. arithmetic
c. integer d. precedence

ANSWER: b
POINTS: 1
REFERENCES: 93

25. ____ occurs when both of the operands are integers.
a. ~~Data modeling~~ ~~b. Type cast~~
c. Integer division d. Unlike assignment

ANSWER: c
POINTS: 1
REFERENCES: 94

26. The percent sign is the ____ operator.
a. remainder b. remaining
c. percentage d. integer division
ANSWER: a
POINTS: 1
REFERENCES: 94

27. What is the value of `result` after the following statement is executed?
```
int result = 2 + 3 * 4;
```
a. 9 b. 10
c. 14 d. 20
ANSWER: c
POINTS: 1
REFERENCES: 95

28. The ____ is the type to which all operands in an expression are converted so that they are compatible with each other.
a. unifying type b. data type
c. numbered d. primitive
ANSWER: a
POINTS: 1
REFERENCES: 101

29. A(n) ____ dialog box asks a question and provides a text field in which the user can enter a response.
a. question b. `JOptPane`
c. confirm d. input
ANSWER: d
POINTS: 1
REFERENCES: 87

30. Each primitive type in Java has a corresponding class contained in the `java.lang` package. These classes are called ____ classes.
a. case b. primitive
c. type-wrapper d. show
ANSWER: c
POINTS: 1
REFERENCES: 89

31. A(n) ____ dialog box typically displays the options Yes, No, and Cancel.
a. confirm b. input
c. message d. answer

ANSWER: a
POINTS: 1
REFERENCES: 91

32. Which of the following is NOT a component of a variable declaration statement?
a. data type identifier b. symbolic constant
c. variable name d. ending semicolon
ANSWER: b
POINTS: 1
REFERENCES: 55

33. You may declare an unlimited number of variables in a statement as long as the variables are ____.
a. the same data type b. initialized to the same value
c. properly commented d. floating point numbers
ANSWER: a
POINTS: 1
REFERENCES: 56

34. When a numeric variable is concatenated to a `String`, the entire expression becomes a(n) _____.
a. `int` b. constant
c. method d. `String`
ANSWER: d
POINTS: 1
REFERENCES: 58

35. Which escape sequence will move the cursor to the beginning of the current line?
a. `\b` b. `\r`
c. `\\` d. `\n`
ANSWER: b
POINTS: 1
REFERENCES: 75

Completion

36. A(n) _____ is a simple data type.
ANSWER: primitive type
POINTS: 1
REFERENCES: 54

37. A(n) _____ operator compares two items and the result has a Boolean value.
ANSWER: relational
comparison
POINTS: 1
REFERENCES: 70

38. A(n) _____ number contains decimal positions.
ANSWER: floating-point
`float`
`double`
POINTS: 1
REFERENCES: 71

39. _____ forces a value of one data type to be used as a value of another type.
ANSWER: Type casting
type casting
POINTS: 1
REFERENCES: 102

40. When you write programs that accept _____, there is a risk that the user will enter the wrong type of data.
ANSWER: user input
POINTS: 1
REFERENCES: 83

Matching

Match each term with the correct statement below.
a. operand
b. cast operator
c. assignment
d. operator precedence
e. garbage value
f. primitive type
g. `float`
h. `boolean`
i. escape sequence
REFERENCES: 93
102
55
95
56
54
71
70
75

41. `true` or `false`
ANSWER: h
POINTS: 1

42. The operator that is represented by an equal sign (=)
ANSWER: c
POINTS: 1

43. A programming term for an unknown value
ANSWER: e
POINTS: 1

44. Java consistently specifies their size and format
ANSWER: f
POINTS: 1

45. A value that can be used on either side of an operator
ANSWER: a
POINTS: 1

46. Rules for the order in which parts of a mathematical expression are evaluated
ANSWER: d
POINTS: 1

47. A floating-point data type
ANSWER: g
POINTS: 1

48. Created by placing the desired result type in parentheses
ANSWER: b
POINTS: 1

49. Begins with a backslash followed by a character
ANSWER: i
POINTS: 1

Subjective Short Answer

50. A variable declaration is a statement that reserves a named memory location. It includes what four elements?
ANSWER: A data type that identifies the type of data that the variable will store
An identifier that is the variable's name
An optional assignment operator and assigned value, if you want a variable to contain an initial value
An ending semicolon
POINTS: 1
REFERENCES: 55

51. Define an integer and then list and describe the four integer data data types.
ANSWER: An integer is a whole number without decimal places. The types `byte`, `short`, `int`, and `long` are all variations of the integer type. The int data type is the most commonly used integer type. A variable of type int can hold any whole number value from –2,147,483,648 to +2,147,483,647. The `byte` and

`short` types occupy less memory and can hold only smaller values; the `long` type occupies more memory and can hold larger values.

POINTS: 1

REFERENCES: 64

52. Describe how to use the boolean data type. Show two examples of a boolean variable assignment; one that uses true or false and one that uses a relational operator.

ANSWER: Boolean logic is based on true or false comparisons. Whereas an int variable can hold millions of different values (at different times), a boolean variable can hold only one of two values—true or false. Besides assigning true and false, you also can assign a value to a Boolean variable based on the result of a comparison.

```
boolean isItPayday = false;
boolean isItPayday = (today="Friday");
```

POINTS: 1

REFERENCES: 70

53. What is the difference between the `float` data type and the `double` data type?

ANSWER: Java supports two floating-point data types: `float` and `double`. A `float` data type can hold floating-point values of up to six or seven significant digits of accuracy. A `double` data type requires more memory than a `float`, and can hold 14 or 15 significant digits of accuracy. The term *significant digits* refers to the mathematical accuracy of a value. For example, a `float` given the value 0.324616777 displays as 0.324617 because the value is accurate only to the sixth decimal position.

POINTS: 1

REFERENCES: 71

54. What is an escape sequence and why would a Java programmer use it to store a character?

ANSWER: You can store any character—including nonprinting characters such as a backspace or a tab—in a `char` variable. To store these characters, you can use an escape sequence, which always begins with a backslash followed by a character—the pair represents a single character.
You might want to use an escape sequence when you want to produce console output on multiple lines in the command window without using multiple println() methods.

POINTS: 1

REFERENCES: 75

55. Describe and give an example of operator precedence.

ANSWER: Operator precedence refers to the rules for the order in which parts of a mathematical expression are evaluated. The multiplication, division, and remainder operators have the same precedence. Their precedence is higher than that for the addition and subtraction operators. Addition and subtraction have the same precedence. In other words, multiplication, division, and remainder always take place from left to right prior to addition or subtraction in an expression. For example, the following statement assigns 14 to `result`: `int result = 2 + 3 * 4;`.

POINTS: 1

REFERENCES: 95

56. In Java, how is it possible to perform mathematical operations on operands with unlike types?

ANSWER: When you perform arithmetic operations with operands of unlike types, Java chooses a unifying type for

the result. The unifying type is the type to which all operands in an expression are converted so that they are compatible with each other. Java performs an implicit conversion; that is, it automatically converts nonconforming operands to the unifying type.

POINTS: 1

REFERENCES: 101

57. Explain how you can override a unifying type imposed by Java. Show an example.

ANSWER: You can purposely override the unifying type imposed by Java by performing a type cast. Type casting forces a value of one data type to be used as a value of another type. To perform a type cast, you use a cast operator, which is created by placing the desired result type in parentheses. Using a cast operator is an explicit conversion. The cast operator is followed by the variable or constant to be cast.
Example:

```
double bankBalance = 189.66;
float weeklyBudget = (float) (bankBalance / 4);
```

POINTS: 1

REFERENCES: 102

58. How can you create and use an input dialog box in Java?

ANSWER: You can create an input dialog box using the `showInputDialog()` method. Six overloaded versions of this method are available, but the simplest version uses a single argument that is the prompt you want to display within the dialog box. The `showInputDialog()` method returns a `String` that represents a user's response; this means that you can assign the `showInputDialog()` method to a `String` variable and the variable will hold the value that the user enters.

POINTS: 1

REFERENCES: 87

59. How would you ask the user to confirm an action using a dialog box?

ANSWER: A confirm dialog box that displays the options Yes, No, and Cancel can be created using the `showConfirmDialog()` method in the `JOptionPane` class. Four versions of the method are available; the simplest requires a parent component (which can be null) and the String prompt that is displayed in the box. The `showConfirmDialog()` method returns an integer containing one of three possible values: `JOptionPane.YES_OPTION, JOptionPane.NO_OPTION, or JOptionPane.CANCEL_OPTION`.

POINTS: 1

REFERENCES: 91

60. Describe how the use of named constants can provide advantages over the use of literal values.

ANSWER: Using named constants makes programs easier to read and understand.
When a constant is defined, you can change the constant at one location, which saves time and prevents you from missing other references.
Using named constants reduces typographical errors that may not be recognized by the compiler.
Constants can be easily identified when named conventionally (all caps).

POINTS: 1

REFERENCES: 57

61. Describe why it is important to assign an appropriate data type to variables in an application.

ANSWER: If you attempt to assign a value that is too large for the data type of the variable, the compiler issues an

error message, and the application does not execute. If you choose a data type that is larger than you need, you waste memory.

POINTS: 1

REFERENCES: 65

62. Describe how the `Scanner` class works with the `System.in` object in order to provide flexibility. Provide an example of using the Scanner class with System.in.

ANSWER: The `System.in` object is designed to read bytes only. Since it is common to accept data of other types, the `Scanner` object can connect to the `System.in` property. This creates a `Scanner` object that will be connected to the default input device.

```
Scanner inputDevice = new Scanner(System.in);
```

POINTS: 1

REFERENCES: 78

63. `100 = salesAmount;`

In terms of assignment operators, why is the above statement illegal?

ANSWER: This assignment operator has a right-to-left associativity. Associativity is the order in which values are used with operators. An identifier that can appear on the left side of an assignment operator sometimes is referred to as an lvalue, and an item that can appear only on the right side of an assignment operator is an rvalue. A variable can be used as an lvalue or an rvalue, but a literal constant can only be an rvalue. Since 100 is a numeric constant, it is an rvalue, which is an item that can appear only on the right side of the assignment operator.

POINTS: 1

REFERENCES: 55

64. Describe three ways in which a named constant differs from a variable.

ANSWER: In its declaration statement, the data type of a named constant is preceded by the keyword final.
A named constant can be assigned a value only once, and then it cannot be changed later in the program. Usually you initialize a named constant when you declare it; if you do not initialize the constant at declaration, it is known as a blank final, and you can assign a value later. Either way, you must assign a value to a constant before it is used.
Named constants conventionally are given identifiers using all uppercase letters, using underscores as needed to separate words.

POINTS: 1

REFERENCES: 56

65. Write the statement to declare an uninitialized integer value for `salesAmt`. Then write the statement to declare a constant named SALESAMT with a value of 20.99.

ANSWER:

```
int salesAmt;

final double SALESAMT = 20.99;
```

POINTS: 1

REFERENCES: 56,57

66. Write the statement that will declare and assign two integer variables, `salesAmt` and `costAmt`, in a single statement. Assign values of your choice to the variables.

ANSWER: `int salesAmt = 100, costAmt = 15;`

POINTS: 1
REFERENCES: 56

67.
```
import javax.swing.JOptionPane;
public class salesJune
{
    public static void main(String[] args)
    {
int storeSales = 250;
    }
}
```

In the above code, complete the statement that will display a message dialog box that will appear centered on the screen and will display the following text:
Congratulations! June sales were $250!

ANSWER: `JOptionPane.showMessageDialog(null, "Congratulations! June sales were $" + storeSales + "!";`

POINTS: 1
REFERENCES: 59-60

68.
```
final int COSTPERITEM = 10;
double sales2012 = amtSold * COSTPERITEM;
```

In the above statements, identify the named constant and describe how a programmer can recognize named constants.

ANSWER: The named constant identifier is `COSTPERITEM`.
Constant declaration statements use the `final` keyword.
Constants are conventionally given identifiers in all uppercase letters.

POINTS: 1
REFERENCES: 56

69. Write the statement that will declare a `char` data type named `testScore` that will hold a letter grade of your choice.

ANSWER: `char testScore = 'A';`

POINTS: 1
REFERENCES: 72

70.
```
public class YourGrade
{
    public static void main(String[] args)
    {
  int projectPoints = 89;
  System.out.print("Your grade for this class is ");
  System.out.print(projectPoints);
  System.out.println("%");
    }
}
```

Given the above code, what will be the output at the command prompt?

ANSWER: Output will be as follows:

```
Your grade for this class is 89%
```

A blank line will follow the output.

POINTS: 1

REFERENCES: 58-59

71. Describe the error message that will be produced when the following code is compiled.

```
public class SalesOct
{
    public static void main(String[] args)
    {
int salesAmt;
System.out.print("October sales are $");
System.out.println(salesAmt);
     }
}
```

ANSWER: The second `println` statement will generate an error message because the variable used in the statement is undeclared. It is legal to declare an uninitialized variable, but it cannot be used in a `println()` statement uninitialized. If you assign a numeric value to `int salesAmt`, the program will compile.

POINTS: 1

REFERENCES: 63

72.
```
public class EndValue
{
    public static void main(String[] args)
    {
int aByte = 940;
   System.out.print("The ending value is "+ aByte);
    }
}
```

When the above code is compiled, what error message will be generated and why?

ANSWER: The above code will result in the error message “possible loss of precision”. The assigned value of 940 to the `aByte` variable is larger than the maximum value allowed. A `byte` type can hold a value between -128 and 127. Thus, the accuracy of the number has been compromised.

POINTS: 1

REFERENCES: 67

73. Why is the following relational operator expression invalid? How could you rewrite the statement so that it is valid?

```
boolean isGradePassing = (grade => 70);
```

ANSWER: In this statement, the order of the operator symbols is reversed. It is illegal to use =<, =>, and =!.

The statement could be modified as follows:

```
boolean isGradePassing = (grade >= 70);
```

POINTS: 1

REFERENCES: 70

74.
```
char aCharacter = 2;
int aNumber = '2';
```

In the above statements, what values will be output after a `println()` statement is executed? Why are the output results different for the two statements?

ANSWER: `aCharacter` will output a blank.
`aNumber` will output a value of 50.

Unicode values are used to assign a unique numeric code. Every computer stores each character it uses as a number and each character is assigned a unique Unicode numeric value.

POINTS: 1

REFERENCES: 73

75. How could you alter the following statement to display "Welcome" on one line and "back" on another line? Show two possible solutions.

```
System.out.println("Welcome back");
```

ANSWER: There are two possible options:

```
System.out.println("Welcome\nback");
```

and

```
System.out.println("Welcome");
System.out.println("back");
```

POINTS: 1

REFERENCES: 76