**Chapter 5: Writing Classes**

**Multiple Choice Questions**:

1) A \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ diagram helps us visualize the contents of and relationships among the classes of a program.

a) class and object

b) UML

c) object-oriented

d) public

e) private

Answer: b

Explanation: A UML diagram helps us visualize the contents and relationships among the classes of a program. The other choices do not refer to any type of diagram.

2) *Regression testing* refers to

a) re-testing a program after fixing a problem to ensure that the fix worked and that it did not introduce another problem.

b) executing the statements in the program in reverse order.

c) executing the program on many different types of computers and comparing the results.

d) running a program with many different sets of inputs.

e) None of these describes regression testing

Answer: a

Explanation: Once an error is found and fixed, regression testing is the process of re-testing the program to both determine that the original problem is fixed and to detect if any new errors were introduced by the fix. Choice d) refers to the general notion of designing sets of inputs to exercise as many branches of a program as possible.

3) When applied to instance variables, the \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ visibility modifier enforces encapsulation.

a) static

b) final

c) public

d) private

e) none of the above

Answer: d

Explanation: The private visibility modifier guards against inappropriate data access, and therefore promotes encapsulation. Choices a) and b) are not visibility modifiers, and choice c) is a visibility modifier that allows public access to an object's data, which violates the principle of encapsulation.

4) Which of the following types of methods do not have any return type (not even a void return type)?

a) methods declared as static

b) methods declared as public

c) methods declared as private

d) constructors

e) all of the above have return types

Answer: d

Explanation: Constructors are the only methods that do not have any return type. They do not even have a void return type. All of the other methods must specify a return type or be declared as void.

5) Which of the following method headers is most likely a header for a mutator method?

a) public int getAge()

b) public double computeSalary()

c) public Person()

d) public void setAge(int newAge)

e) none of these are headers for a mutator method

Answer: d

Explanation: Mutators are methods that change the value of an instance variable, and are often referred to as "setters." Therefore, choice d) is the correct answer. Choice a) is an example of a header for a accessor method, often referred to as a "getter." Choice c) is a constructor, and choice b) is a class method.

6) A \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ variable is shared among all instances of a class.

a) static

b) final

c) public

d) private

e) none of the above

Answer: a

Explanation: A static variable is shared among all instances of a class. A final variable is a constant, a public variable is one that is accessible from outside the object and a private variable is one that cannot be accessed outside of the object.

7) A(n) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ is an application that displays the inner workings of an executing program.

a) stethoscope

b) telescope

c) debugger

d) signal analyzer

e) oscilloscope

Answer: c

Explanation: Debugging is the act of locating and correcting run-time and logic errors in your programs. A debugger is a software application that allows us to observe the inner workings of a program as it executes. Choice a) is an analytic tool used in medicine. Choice b) is a tool used in astronomy. Choices d) and e) are used in analyzing hardware components.

8) The \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ reference always refers to the currently executing object.

a) null

b) static

c) final

d) actual

e) this

Answer: e

Explanation: The this reference always refers to the currently executing object. A null reference is a reference that is not pointing to any object. The other three choices are not special references in Java.

9) A method that has multiple definitions is an \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ method.

a) overloaded

b) overridden

c) overlooked

d) overclocked

e) none of the above

Answer: a

Explanation: A method that has multiple definitions is an overloaded method. The versions of an overloaded method are distinguished by the number, type and order of their parameters. Overridden methods are methods that have been redefined later in an inheritance hierarchy. They will be studied in more detail later. Choice c and d are not types of methods in Java.

10) A(n) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ is a step-by-step process for solving a problem.

a) UML diagram

b) aggregate object

c) class

d) algorithm

e) none of the above

Answer: d

Explanation: An algorithm is a step-by-step solution for solving a problem. A UML diagram is a way of visually representing how classes and objects interact. An aggregate object is an object that is composed, in part, of other objects. A class can be thought of as a blueprint for a set of objects.

11) All methods (with the exception of constructors) must specify a return type. What is the return type for a method that does not return any values?

a) int

b) public

c) double

d) void

e) none of the above

Answer: d

Explanation: Methods that do not need to return any data should have void specified as the return type. A method cannot have public specified as its return type, so choice b is incorrect. Choices a) and c) specify a return type, and therefore they must return data of that type.

12) Methods that can be called directly through the class name and do not need to have an object instantiated are called \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

a) final

b) public

c) static

d) private

e) none of the above

Answer: c

Explanation: Methods that can be called directly through the class name must be declared as static. Choices b) and d) are visibility modifiers for a method. Methods declared as final cannot be overridden.

13) A(n) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ object is one that is made up, at least in part, of other objects.

a) static

b) aggregate

c) encapsulated

d) private

e) public

Answer: b

Explanation: An aggregate object is one that is made up of other objects. Choice a), d) and e) do not refer to types of objects. Encapsulated objects may be made up of primitive types or object types.

14) If a service is so complex that it cannot be reasonably be implemented using one method, it is often helpful to decompose it to make use of \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ support methods.

a) static

b) aggregate

c) private

d) public

e) final

Answer: c

Explanation: Private support methods are useful when a service is too complex to be defined in a single method. Therefore choice c) is correct.

15) The versions of an overloaded method are distinguished by \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

a) the number, type and order of their parameters

b) their identifiers

c) their classes

d) the number and type of their parameters

e) the number of their parameters

Answer: a

Explanation: Overloaded methods are two methods in the same class that have the same identifier, but a different number, type or order of parameters. Therefore, choice a) is correct and the rest are incorrect.**True/False Questions**:

1) A variable can always be referenced anywhere in a program.

Answer: False

Explanation: Scope rules, based on a variable's declaration, determine where it can be referenced.

2) An object can be thought of as a blueprint for a set of classes.

Answer: False

Explanation: A class can be thought of as a blueprint for a set of objects; not the other way around.

3) A return statement is not required at the end of every method.

Answer: True

Explanation: Constructors and methods with a return type of void do not require return statements.

4) *Black-box testing* uses the internal structure and implementation of the code to be tested in designing the tests.

Answer: False

Explanation: Black-box testing treats the code being tested as a black box that cannot be opened or examined. Tests are developed based on possible inputs and expected outputs. Black-box testing does not use knowledge of the internal design or the structure of the code.

5) A *break point* is a statement in a program that causes the program to crash.

Answer: False

Explanation: A breakpoint is a statement in a program that is marked for use by a debugger. When a program is running under the control of a debugger, execution will pause when a breakpoint is encountered. When paused, the contents of variables can be examined. Program execution can then be resumed in one of several ways: run until the program completes, run until the next breakpoint, run a single statement, etc.

6) A main method can only access static or local variables.

Answer: True

Explanation: A main method cannot access non-static and non-local variables because it is a static method. In particular, it cannot access any variables declared at the class level.

7) In a class that has variables called height and width, methods called getHeight() and getWidth() are examples of accessor methods.

Answer: True

Explanation: Accessor methods return the value of instance variables. They are often named using the word 'get' followed by the name of the instance variable.

8) Every class has a constructor, whether defined by the programmer or not.

Answer: True

Explanation: Every class automatically has a default constructor that doesn't take any parameters. The default constructor is used when objects are created if no other constructor is defined.

9) Variables that are declared as static are shared among all instances of a class.

Answer: True

Explanation: Static variables are sometimes called class variables because they are shared among all instances of a class.

10) Aggregation is sometimes described as a *has-a* relationship.

Answer: True

Explanation: Aggregate objects are objects that contain other objects as instance variables. Therefore the relationship between an object that is an aggregate of other objects is often described as a *has-a* relationship.**Short Answer Questions**:

1) What is the difference between an object and a class?

Answer: A class can be thought of as the blueprint for an object. In other words, the object is the embodiment of a class.

2) Explain the difference between actual parameters and formal parameters.

Answer: Formal parameters are used in the definition of a method, while actual parameters are the values that are used in the method call.

3) What is encapsulation? How can it be enforced in Java?

Answer: Encapsulation is the principle that objects should be self-governing. In other words, an object's internal data should be protected from outside access. Encapsulation can be enforced in Java by making instance variables private.

4) What is *instance data*?

Answer: Instance data refers to the data that will be stored in each object of a class. New memory space is allocated for instance data every time that an object is created. Each object of a class has its own unique memory allocation, so that its instance data is unique from that of any other objects of the same class.

5) Write a method called randomInRange that takes in two numbers representing a range. Print an error message and return zero if the second parameter is less than the first. Otherwise, the method should return a randomly generated integer in that range (inclusive). You may assume that the class has a static Random object called generator already declared and instantiated.

Answer:

public int randomInRange(int a, int b) {

if(b < a) {

System.out.println("Error, invalid range!");

return 0;

}

return generator.nextInt(b – a + 1) + a;

}

6) Write a method called isPalindrome that accepts a String as a parameter and returns true if the String is a palindrome, and false otherwise. You may assume that the entered String consists entirely of lowercase letters (meaning it contains no numbers, spaces, punctuation, etc). Hint: write code that creates a new string that is the original string reversed, and then check to see if the two strings are equal.

Answer:

public boolean isPalindrome(String s) {

String reverseS = new String("");

for(int i = s.length()-1; i >= 0; i--)

reverseS += s.charAt(i);

return reverseS.equals(s);

}

7) What is the difference between a *service method* and a *support method* in a class definition?

Answer: A service method is used to access or modify the data values of an object. They generally have public visibility. Support methods are designed to be used by service methods to support the tasks that the service methods perform. Support methods generally have private visibility.

8) Can a static method access instance data in a class?

Answer: No. Static methods are defined at the class level. Instance data is allocated at the object level. Static methods do not have access to object level data.

9) Student is a class that has the following instance variables: name (a String), yearInSchool (an int), and gpa (a double). Write a constructor for the Student class that takes parameters for these instance variables as initial values of the variables. Use the this reference in the constructor.

Answer:

public Student(String name, int yearInSchool, double gpa)

{

this.name = name;

this.yearInSchool = yearInSchool;

this.gpa = gpa;

}

10) Java uses *pass by value* for passing parameters on a method call. What does this mean if the parameter is a primitive data type? What does this mean if the parameter is a reference to an object?

Answer: *Pass by value* means that the value associated with the actual parameter is copied into the memory location of the formal parameter when the method is called. The formal and actual parameters occupy distinct locations in memory. If the parameter is a primitive data type, any modifications made to the formal parameter during the execution of the method will most likely not be reflected in the actual parameter. If the parameter is a reference to an object, then it holds the location of an object. While the formal and actual parameters will be distinct, they will both hold the address of the object and will be aliases. If the method modifies the object, then those modifications will be apparent when referencing the object using the actual parameter after the method terminates.

11) Give an example of how a static variable can be used in a class.

Answer: Suppose that a program needs to keep track of how many objects of a particular class have been created. Declaring and managing a counter at the program level could be a complex task. It might be simpler to let the class itself maintain the counter. A static int variable could be declared at the class level to be used as a counter. All of the constructors of the class could increment the static variable. A static method would also be needed to return the value of the counter.

12) Method overloading requires that the methods have different signatures. What parts of a method are used in the signature?

Answer: The method name, the number of parameters, the types of the parameters, and the orders of the parameters are all part of the method signature.

13) Explain why method overloading is useful.

Answer: Method overloading is useful because it allows multiple methods to have the same name as long as they have different parameter types. This allows for more flexibility in choosing identifiers for methods, since the alternative would require the programmer to have different method identifiers for every method that took in a different type of parameter.

14) How are unit testing, integration testing, and system testing related? How are they different?

Answer: They are related in that they are all forms of software testing. They differ in respect to the size (the amount of code) that they test. Unit testing tests the individual modules or components as units. These units may be individual classes or methods. Integration testing combines several related units and tests them as a collective entity. System testing tests the overall software system as a whole, which is comprised of several sets of software that have passed integration testing.

15) How can a print() or println() statement be used in debugging?

Answer: Print statements can be used to show the contents of variables at specific points in the code execution. This can be useful if a program produces incorrect results and the programmer or tester is trying to detect where an incorrect value first appeared in the code.