True False

1) An instance variable can be directly referenced in a static method within the same class without creating an object

Answer: F

2) You declare int height; within a public method and declare int height; as an instance variable within the same class. This will not cause a compile error

Answer: T

3) Java assigns a default value to a variable declared within a method if the variable is not initialized

Answer: F

4) Object obj = new Object(); Student student = new Student(); the statement student = obj; will not cause a compile error

Answer: F

5) If the modifier in a method signature is omitted, the default modifier is “public”.

Answer: F

6) The purpose of a testbed main() within a Java class is to run system testing.

Answer: F

7) String str = new String(“Java”); the statement Object obj = str; creates a new instance of Object.

Answer: F

8) A subclass inherits all methods, instance variables, and constructors from the superclass.

Answer: F

9) The following statement will not cause a compile error, assuming all classes are properly defined.

public class MyBike extends MountainBike, RoadBike implements Bicycle {…}

Answer: F

10) Software inspection is a software testing technique that requires the software developers to run the programs with the test cases.

Answer: F

11) Assume Employee class is an abstract class. The following statement outside of Employee class will not cause a compile error: Employee[] emps = new Employee[Capacity];

Answer: T

12) The code below will compile.

class Square extends GeometricObject {

double length;

Square(double length) { GeometricObject(length); }

}

Answer: F

13) Pizza is defined as a Java Interface; the statement Pizza [] pizzas = new Pizza[Capacity]; will not cause a compile error.

Answer: T

14) A Java interface class may contain constructors

Answer: F

15) RuntimeException and Error are known as checked exceptions.

Answer: F

Answer Questions 16-19 based on the classes defined below. Assume TA ta; TB tb; are declared.

public class TA {

protected int a1;

public TA(int i) {a1 = i;}

public void doIt() {…}

public String toString() {…}

}

public class TB extends TA {

private int b1;

public TB(int I, int j) {super(i); b1=j;}

public String toString() {…}

}

16) An instance of TB includes 2 instance variables

Answer: T

17) The following statement will not cause a compile error: tb = new TA(3);

Answer: F

18) The following statements will not cause a compile error: tb = new TB(3,4); tb.doIt();

Answer: T

19) tb = new TB(2,1); ta = tb; System.out.print(ta.toString()); will call the toString() in TA.

Answer: F

20) The following code compiles without errors.

Public class Test extends A {

public static void main(String args[]) {

Test t = new Test();

t.print();

}

}

Class A {

String s;

A(String s) {

This.s = s;

}

Public void print() {

System.out.println(s);

}

}

Answer: F

Multiple choice/fill in the blank

1. You may directly assign a primitive data type value to a wrapper object. This is called:
   1. Auto unboxing
   2. Autoboxing
   3. Explicit casting
   4. Implicit casting
2. In a Class Diagram, the relationship between a Java Interface class and the class that implements it is a
   1. Composition
   2. Aggregation
   3. Generalization
   4. Realization
3. An aggregation relationship is usually represented as \_\_\_\_ in \_\_\_\_
   1. An instance variable/ the aggregating class
   2. An instance variable/ the aggregated class
   3. A method/ the aggregating class
   4. A method/ the aggregated class
4. The default values for instance variables with Boolean type, numeric type, and Object type are \_\_\_\_, respectively.
   1. true, 1, null
   2. false, 0 , null
   3. true, 0 , null
   4. true, 1, null
   5. false, 1, null
5. With the boundary analysis in Equivalence Class Partitioning, which set of values is considered as boundary values for a parameter n with integer type, where 3 <=n <=36?
   1. 3,12,24,36
   2. 2,3,36,37
   3. 0,3,35,36
   4. 0,-1,1,3,36
6. Which of the following statements is true?
   1. Every class has a default constructor
   2. At least one constructor must always be defined explicitly
   3. A default constructor is provided automatically if no constructors are explicitly declared.
   4. None of the above
7. Suppose A is an abstract class, B is a concrete subclass of A, and both A and B have a default constructor. Which of the following is correct?
   1. A a = new A();
   2. A a = new B();
   3. B b = new A();
   4. None of the above
8. What is the output of the following code?  
   public class Test {  
    public static void main(String args[]) {  
    String s1 = “Welcome to Java!”;  
    String s2 = “Welcome to Java!”;  
    if (s1 == s2)  
    System.out.println(“s1 and s2 reference to the same String object”);  
    else  
    System.out.println(“s1 and s2 reference to different String objects”);  
    }  
   }
   1. s1 and s2 reference to the same String object
   2. s1 and s2 reference to different String objects
9. Given the declaration Circle[] circles = new Circle[10], which of the following statements is most accurate?
   1. circles contains an array of 10 int values.
   2. circles contains an array of 10 objects of the Circle type.
   3. circles contains a reference to an array and each element in the array can hold a reference to a Circle object.
   4. circles contain a reference to an array and each element in the array can hold a Circle object.
10. Which of the following statements is not true?
    1. A public class can be accessed by a class from a different package
    2. A private method cannot be accessed by a class in a different package.
    3. A protected instance variable cannot be accessed by a subclass in a different package.
    4. A method with no modifier can be accessed by a class in the same package.
11. Which of the following statements regarding abstract methods is false?
    1. An abstract class can be used as a data type
    2. An abstract class can be extended.
    3. A subclass of a non-abstract superclass can be abstract
    4. A subclass can override a concrete method in an abstract superclass.
    5. None of the above
12. Which of the following statements is not true about an immutable object?
    1. The data of an immutable object cannot be modified.
    2. All instance variables of an immutable object must be private.
    3. All instance variables of an immutable object must be of primitive types.
    4. A reference type instance variable of an immutable object must also be immutable.
    5. An immutable object contains no mutator methods.
13. In a class Diagram, the notation for the visibility modifier “protected” is
    1. +
    2. –
    3. #
    4. ~
14. Every student has a profile. A student is uniquely identified by the student’s profile. What is the best relationship type you use in a Class Diagram between the Student class and the Profile class?
    1. Composition
    2. Aggregation
    3. Generalization
    4. Realization
15. The advantage of using the Java Collections framework is
    1. Reduce programming efforts
    2. Software reuse
    3. Improve program quality
    4. All of the above
16. Analyze the following code.  
    public class Test {  
     public static void main(String args[]) { new B(); }  
    }  
    class A {  
     int I = 7;  
     public A() { setI(20); System.out.println(“i from A is “ + i); }  
     public void setI(int i) { this.i = 3 \* i; }  
    }  
    class B extends A {  
     public B() { //System.out.println(“i from B is “ + i); }  
     @Override  
     public void setI(int i) { this.i = 3 \* i; }  
    }
    1. The constructor of class A is not called.
    2. The constructor of class A is called and it displays “i from A is 7”
    3. The constructor of class A is called and it displays “i from A is 40”
    4. The constructor of class A is called and it displays “i from A is 60”
17. What is the output of the following code?  
    public class Test {  
     public static void main(String args[]) {  
     new Person().printPerson();  
     new Student().printPerson();  
     }  
    }  
      
    class Student extends Person {  
     public String getInfo() { return “Student”; }  
    }  
      
    class Person {  
     public String getInfo() { return “Person”; }  
     public void printPerson() {  
     System.out.println(getInfo());  
     }  
    }
    1. Person Person
    2. Person Student
    3. Student Student
    4. Student Person
18. Given the following code, which of the expressions evaluated to be false?  
    class C1 {}  
    class C2 extends C1 {}  
    class C3 extends C2 {}  
    class C4 extends C1{}  
    C1 c1 = new C1();  
    C2 c2 = new C2();  
    C3 c3 = new C3();  
    C4 c4 = new C4();
    1. c1 instance of C1
    2. c2 instance of C1
    3. c3 instance of C1
    4. c4 instance of C2
19. What is the output of running class C?  
    class A {  
     public A() { System.out.print(“The default constructor of A is invoked”); }  
    }  
      
    class B extends A {  
     public B() { System.out.print(“The default constructor of B is invoked”); }  
    }  
      
    public class C {  
     public static void main(String[] args) {  
     B b = new B();  
     }  
    }
    1. Nothing displayed
    2. “The default constructor of B is invoked”
    3. “The default constructor of A is invokedThe default constructor of B is invoked”
    4. “The default constructor of B is invokedThe default constructor of A is invoked”
    5. “The default constructor of A is invoked”
20. Answer questions 20-23 based on the following code.  
    try {  
     statementA;  
     statementB;  
     statementC;  
    }  
    catch (ArithmeticException exp) {…}  
    catch (NumberFormatException exp) {…}  
    finally{…}  
    statementD;  
      
    If statementB causes an ArithmeticException, will statement be executed?
    1. Yes
    2. No
21. If statementB causes an IndexOutOfBoundsException, will statementD be executed?
    1. Yes
    2. No
22. If statementB causes an IndexOutOfBoundsException, will finally{…} be executed?
    1. Yes
    2. No
23. statementB causes an NumberFormatException, will statementD be executed?
    1. Yes
    2. No

Multiple Choice Answers

Bdabb cbacc ecad dbdcb baa

Short Answer Questions

1. List 3 reasons why writing “clean code” is necessary when developing software products
2. Briefly discuss the benefits of using an Object-Oriented approach to develop software products and how this is done in Java.
3. Given the Java method signature and requirements below.  
   public int memberFee(Frequency frequency, Boolean isRenewal) { }  
     
   -The method shall return the next member fee due for a member based on the paying frequency and the renewal status  
   -Frequency is an enum class where monthly, quarterly, and annually are defined for the frequency of paying the member fees. That is, members pay one month, thee month, or twelve month fees based on the indicated frequency.  
   -There is a one-time fee, $20, for initial enrollment of membership; member will not be charged of the one time fee if this is a renewal.  
   -Depending on the payment frequency, the method shall apply different monthly rates  
   -The method shall apply $30/month for member fee paid monthly, $25/month for quarterly, and $20/month for annually.
   1. Consider the Java method as a black box, list the equivalence classes for selecting the test cases. DO NOT write Java code, or you will get 0 points for this question!
   2. Do we need additional test cases for the boundary analysis to test the method? Why?

Short Answer Question Answers

1. Answers:
   1. Enhance the reusability of code, so it will be easier to change the code by other developers
   2. Clear and clean API will minimize the dependencies between classes, meaning less impact when making software changes.
   3. Make software testing and debugging easier if the logic is clear.
2. Answers:
   1. Enhance the maintainability, so it’s easier to adapt change. This includes:
      1. Information hiding to minimize the impact on client programs in case of software change (by encapsulation)
      2. Extendibility to add new features without modifying the existing code (by inheritance)
      3. Code reuse: to generalize data and operations in superclasses (by inheritance)
      4. Software change without modifying the existing code (by inheritance and overriding/ polymorphism/dynamic binding)
3. Answers:
   * 1. 6 equivalence classes:  
        Monthly, Renewal, Expected Output: 30  
        Monthly, NotRenewal, Expected Output: 20 + 30  
        Quarterly, Renewal, Expected Output: 25 \* 3  
        Quarterly, NotRenewal, Expected Output: 20 + 25 \* 3  
        Annually, Renewal, Expected Output: 20 \* 12  
        Annually, NotRenewal, Expected Output: 20 + 20 \* 12
     2. No, because all possible inputs are covered, and the user cannot create additional inputs outside of the ones given.