Inheritance Test 1

1. Which of the following is **not** a major aspect of object-oriented programming (OOP)?
2. inheritance
3. refactoring
4. encapsulation
5. polymorphism

Questions 2 – 3 refer to the following class definitions.

public class Reptile  
{  
 // implementation not shown  
}

public class Lizard extends Reptile  
{  
 // implementation not shown  
}

public class Gecko extends Lizard  
{  
 // implementation not shown  
}

1. Which of the following does not correctly instantiate a Lizard object?
2. Lizard liz = new Reptile();
3. Reptile liz = new Lizard();
4. Lizard liz = new Lizard();
5. All are correct
6. Which of the following does not correctly instantiate a Gecko object?
7. Reptile gecko = new Gecko();
8. Lizard gecko = new Gecko();
9. Gecko gecko = new Gecko();
10. All are correct

Questions 4-6 refer to the following class definitions.

public class Food {  
 private int cost;  
  
 public Food(int c) {  
 cost = c;  
 }

public int getCost() {  
 return cost;  
 }  
 public String toString() {  
 return "Cost = " + cost;  
 }  
}

public class Pizza extends Food {  
 private int size;  
  
 public Pizza(int c, int s) {  
 <\*1>  
 size = s;  
 }

public int getSize() {  
 return size;  
 }

public String toString() {  
 return <\*2> + ", Size = " + size;  
 }  
}

1. Which of the following correctly replaces <\*1> so that the instance variable **cost** is initialized properly?
2. cost = c
3. super.cost = c
4. super()
5. super(c)
6. What is output by the following code segment?

Food food = new Pizza(12, 8);  
System.out.println(food.getCost() + " " + food.getSize());

1. 0 0
2. 12 8
3. 8 12
4. 12 0
5. Look at the following code segment.

Pizza myPizza = new Pizza(10, 12);  
System.out.println(myPizza);

Which of the following correctly replaces <\*2> so that the code will display myPizza in the following format:  
 Cost = 10, Size = 12

1. "Cost = " + cost
2. "Cost = " + super.cost
3. super()
4. super.toString()

Questions 7-8 refer to the following class definitions.

public class A  
{

public void doSomething() {

System.out.print("A");

}

}

public class B extends A

{

public void doSomething() {  
 System.out.print("B");  
 }

public void doMore() {

doSomething();  
 System.out.print("BBB");

}

}

public class C  
{  
 public static void main(String[] args) {  
 A obj1 = new A();

A obj2 = new B();  
  
 obj1.doSomething (); // Line 1  
 obj2.doSomething (); // Line 2   
 }

}

1. What does Line 1 display?
2. A
3. B
4. AB
5. BA
6. What does Line 2 display?
7. A
8. B
9. AB
10. BA
11. Suppose the following code was added to the main method.

B obj3 = new B();

obj3.doMore(); // Line 3

What does Line 3 display?

1. BBB
2. ABBB
3. BBBB
4. Nothing is printed due to a syntax error

10. Suppose the following code was added to the main method.

A obj4 = new B();  
 obj4.doMore(); // Line 4

What does Line 4 display?

1. BBB
2. ABBB
3. BBBB
4. Nothing is printed due to a syntax error