

JAVA Programming Language Homework I — OO concept

Student ID: Name:

1. Which of the following techniques can be used to prevent the instantiation of a class by any code outside of the class?

- A. Declare all constructors with a void return type.
- B. Declare all constructors using the private access modifier.
- C. Do not declare any constructors inside a class definition.
- D. Do not include a return statement in the constructor.
- E. None of the above.

Answer:

2. Which of the following statements are true?

- A. A constructor can invoke the constructor of the direct superclass using the superclass constructor invocation statement “super”.
- B. By using constructor invocation statement “this”, a constructor can invoke another constructor of the same class.
- C. The constructor invocation statement, “this”, can legally appear anywhere in the constructor body.
- D. By using the constructor invocation statement “this”, a constructor can invoke itself.
- E. None of the above.

Answer:

3. Given the following Java code:

```
1.      public class Hello {
2.          String title="";
3.          int value;
4.          public Hello( ) {
5.              title = title + " World";
6.              System.out.print(title);
7.          }
```

```
8.         public Hello(int value) {
9.             this.value = value;
10.            title = "Hello";
11.            this();
12.        }
13.        public static void main(String[] args){
14.            Hello b = new Hello (5);
15.        }
16.    }
```

What is the result?

- A. Hello
- B. Hello World
- C. Compilation fails
- D. Hello World 5
- E. Hello Hello

Answer:

4. Given the following Java code:

```
1.     class Num {
2.         public static String b() { return "One"; }
3.         public static String b( int i ) { return "Two"; }
4.         public static String b( int i, int j ) throws Exception { return "Three"; }
5.         public static void main( String[] args ) {
6.             System.out.println( b(2) );
7.         }
8.     }
```

What is the result?

- A. One
- B. Two
- C. Compilation fails
- D. Three
- E. None of the above

Answer:

5. Given the following Java code:

Exhibit:

```
1.    public class SimpleCalc {  
2.        public int value;  
3.        public void calculate() { value += 7; }  
4.    }
```

And:

```
1.    Public class MultiCalc extends SimpleCalc {  
2.        public void calculate() { value -= 3; }  
3.        public void calculate( int multiplier) {  
4.            calculate( );  
5.            super.calculate( );  
6.            value *= multiplier;  
7.        }  
8.        public static void main(String[] args) {  
9.            MultiCalc calculator = new MultiCalc( );  
10.           calculator.calculate(2);  
11.           System.out.println(“ Value is: “ + calculator.value);  
12.       }  
13.   }
```

What is the result?

- A. Value is: 8
- B. Compilation fails.
- C. Value is: 12
- D. Value is: -12
- E. The code runs with no output.

Answer:

6. Given the following Java code:

```

1. public class Base {
2.     public static final String FOO = "foo";
3.     public static void main(String[] args) {
4.         Base b = new Base();
5.         Sub s = new Sub();
6.         System.out.println(Base.FOO);
7.         System.out.println(Sub.FOO);
8.         System.out.println(b.FOO);
9.         System.out.println(s.FOO);
10.        System.out.println(((Base)s).FOO);
11.    }
12. }
13. class Sub extends Base {public static final String FOO="bar";}

```

What is the result?

- A. fofofofofofofo
- B. foobarfoobarbar
- C. foobarfofofofo
- D. foobarfoobarfoo
- E. fofofofoobarbar

Answer:

7. Given the following Java code:

```

1. public class TestPoly {
2.     public static void main(String[] args) {
3.         Parent p = new Child();
4.     }
5. }
6.
7. class Parent {
8.     public Parent() {
9.         super();
10.        System.out.println("instantiate a parent");
11.    }
12. }

```

```
13.  
14.   class Child extends Parent {  
15.       public Child() {  
16.           System.out.println("instantiate a child");  
17.       }  
18.   }
```

What is the result?

- A. instantiate a child
- B. instantiate a parent
- C. instantiate a child
instantiate a parent
- D. instantiate a parent
instantiate a child
- E. Compilation fails

Answer:

8. Given the following Java code:

```
1.   public class TestPoly {  
2.       public static void main(String[] args) {  
3.           Parent p = new Child();  
4.       }  
5.   }  
6.  
7.   class Parent {  
8.       public Parent() {  
9.           super();  
10.          System.out.println("instantiate a parent");  
11.      }  
12.  }  
13.  
14.  class Child extends Parent {  
15.      public Child() {  
16.          System.out.println("instantiate a child");  
17.          super();
```

```
18.    }
19.    }
```

What is the result?

- A. instantiate a child
- B. instantiate a parent
- C. instantiate a child
instantiate a parent
- D. instantiate a parent
instantiate a child
- E. Compilation fails

Answer:

9. Given the following Java code:

```
1.    class C {
2.        public static void main(String[] args) {
3.            A tmp = new B ();
4.            tmp.m1 ();
5.            tmp.m2 ();
6.            ((B)tmp).m1 ();
7.            ((B)tmp).m2 ();
8.        }
9.    }
10.   class A {public void m1() { System.out.println ("A");}}
11.   class B extends A {
12.       public void m1() { System.out.println ("B1");}
13.       public void m2() { System.out.println ("B2");}
14.       public void m3() { System.out.println ("B3");}
15.       public void m4() { System.out.println ("B4");}
16.   }
```

What is the result?

- A. AB2B1B2
- B. B1B2B1B2
- C. Compiler Error

D. Runtime Error

E. None of the above

Answer:

10. Given the following Java code:

```
1.     public class Bootchy {
2.         int botch;
3.         String snootch;
4.         public Bootchy() {
5.             this("snootchy");
6.             System.out.print("first ");
7.         }
8.         public Bootchy(String snootch) {
9.             this(420, "snootchy");
10.            System.out.print("second ");
11.        }
12.        public Bootchy(int bootch, String snootch) {
13.            this.bootch=botch;
14.            this.snootch = snootch;
15.            System.out.print("third ");
16.        }
17.        public static void main(String[] args){
18.            Bootchy b = new Bootchy();
19.            System.out.print(b.snootch +" "+ b.bootch);
20.        }
21.    }
```

What is the result?

(A) snootchy 420 third second first

(B) snootchy 420 first second third

(C) first second third snootchy 420

(D) third second first snootchy 420

(E) third first second snootchy 420

Answer: