SELF TEST ANSWERS

1. Which is true? (Choose all that apply.)

A. "X extends Y" is correct if and only if X is a class and Y is an interface

B. "X extends Y" is correct if and only if X is an interface and Y is a class

C. "X extends Y" is correct if X and Y are either both classes or both interfaces

D. "X extends Y" is correct for all combinations of X and Y being classes and/or interfaces

Answer:

☑ **C** is correct.

☒ **A** is incorrect because classes implement interfaces, they don't extend them. **B** is incorrect because interfaces only "inherit from" other interfaces. **D** is incorrect based on the preceding rules. (Objective 1.2)

2. Which method names follow the JavaBeans standard? (Choose all that apply.)

A. `addSize`

B. `getCust`

C. `deleteRep`

D. `isColorado`

E. `putDimensions`

Answer:

☑ **B** and **D** use the valid prefixes '`get`' and '`is`'.

☒ **A** is incorrect because '`add`' can be used only with Listener methods. **C** and **E** are incorrect because '`delete`' and '`put`' are not standard JavaBeans name prefixes. (Objective 1.4)

3. Given:

```
class Voop {
  public static void main(String[] args) {
    doStuff(1);
    doStuff(1,2);
  }
  // insert code here
}
```

Which, inserted independently at line 6, will compile? (Choose all that apply.)

A. `static void doStuff(int... doArgs) { }`

B. `static void doStuff(int[] doArgs) { }`

C. `static void doStuff(int doArgs...) { }`

D. `static void doStuff(int... doArgs, int y) { }`

E. `static void doStuff(int x, int... doArgs) { }`

Answer:

☑ **A** and **E** use valid var-args syntax.

☒ **B** and **C** are invalid var-arg syntax, and **D** is invalid because the var-arg must be the last of a method's arguments. (Objective 1.4)

4. Given:

```
 1. enum Animals {
 2.   DOG("woof"), CAT("meow"), FISH("burble");
 3.   String sound;
 4.   Animals(String s) { sound = s; }
 5. }
 6. class TestEnum {
 7.   static Animals a;
 8.   public static void main(String [] args) {
 9.     System.out.println(a.DOG.sound + " " + a.FISH.sound);
10.   }
11. }
```

What is the result?

A. `woof burble`

B. Multiple compilation errors

C. Compilation fails due to an error on line 2

D. Compilation fails due to an error on line 3

E. Compilation fails due to an error on line 4

F. Compilation fails due to an error on line 9

Answer:

☑ **A** is correct; enums can have constructors and variables.

☒ **B, C, D, E,** and **F** are incorrect; these lines all use correct syntax. (Objective 1.3)

5. Given two files:

```
 1. package pkgA;
 2. public class Foo {
 3.   int a = 5;
 4.   protected int b = 6;
 5.   public int c = 7;
 6. }

 3. package pkgB;
 4. import pkgA.*;
 5. public class Baz {
 6.   public static void main(String[] args) {
 7.     Foo f = new Foo();
 8.     System.out.print(" " + f.a);
 9.     System.out.print(" " + f.b);
10.     System.out.print(" " + f.c);
11.   }
12. }
```

What is the result? (Choose all that apply.)

A. 5 6 7

B. 5 followed by an exception

C. Compilation fails with an error on line 7

D. Compilation fails with an error on line 8

E. Compilation fails with an error on line 9

F. Compilation fails with an error on line 10

Answer:

☑ **D** and **E** are correct. Variable a has default access, so it cannot be accessed from outside the package. Variable b has protected access in pkgA.

☒ **A, B, C,** and **F** are incorrect based on the above information. (Objective 1.1)

6. Given:

```
1. public class Electronic implements Device
         { public void doIt() { } }
2.
3. abstract class Phone1 extends Electronic { }
4.
5. abstract class Phone2 extends Electronic
     { public void doIt(int x) { } }
6.
```

```
 7. class Phone3 extends Electronic implements Device
        { public void doStuff() { } }
 8.
 9. interface Device { public void doIt(); }
```

What is the result? (Choose all that apply.)

A. Compilation succeeds

B. Compilation fails with an error on line 1

C. Compilation fails with an error on line 3

D. Compilation fails with an error on line 5

E. Compilation fails with an error on line 7

F. Compilation fails with an error on line 9

Answer:

☑ **A** is correct; all of these are legal declarations.

☒ **B, C, D, E,** and **F** are incorrect based on the above information. (Objective 1.2)

7. Given:

```
 4. class Announce {
 5.   public static void main(String[] args) {
 6.     for(int __x = 0; __x < 3; __x++) ;
 7.     int #lb = 7;
 8.     long [] x [5];
 9.     Boolean []ba[];
10.     enum Traffic { RED, YELLOW, GREEN };
11.   }
12. }
```

What is the result? (Choose all that apply.)

A. Compilation succeeds

B. Compilation fails with an error on line 6

C. Compilation fails with an error on line 7

D. Compilation fails with an error on line 8

E. Compilation fails with an error on line 9

F. Compilation fails with an error on line 10

Answer:

☑ **C, D,** and **F** are correct. Variable names cannot begin with a #, an array declaration can't include a size without an instantiation, and enums can't be declared within a method.

☒ **A, B,** and **E** are incorrect based on the above information. (Objective 1.3)

8. Given:

```
 3. public class TestDays {
 4.   public enum Days { MON, TUE, WED };
 5.   public static void main(String[] args) {
 6.     for(Days d : Days.values() )
 7.       ;
 8.     Days [] d2 = Days.values();
 9.     System.out.println(d2[2]);
10.   }
11. }
```

What is the result? (Choose all that apply.)

A. `TUE`

B. `WED`

C. The output is unpredictable

D. Compilation fails due to an error on line 4

E. Compilation fails due to an error on line 6

F. Compilation fails due to an error on line 8

G. Compilation fails due to an error on line 9

Answer:

☑ **B** is correct. Every `enum` comes with a `static values()` method that returns an array of the `enum`'s values, in the order in which they are declared in the `enum`.

☒ **A, C, D, E, F,** and **G** are incorrect based on the above information. (Objective 1.3)

9. Given:

```
 4. public class Frodo extends Hobbit {
 5.   public static void main(String[] args) {
 6.     Short myGold = 7;
 7.     System.out.println(countGold(myGold, 6));
 8.   }
 9. }
10. class Hobbit {
11.   int countGold(int x, int y) { return x + y; }
12. }
```

What is the result?

A. 13

B. Compilation fails due to multiple errors

C. Compilation fails due to an error on line 6

D. Compilation fails due to an error on line 7

E. Compilation fails due to an error on line 11

Answer:

☑ **D** is correct. The `Short myGold` is autoboxed correctly, but the `countGold()` method cannot be invoked from a static context.

☒ **A, B, C,** and **E** are incorrect based on the above information. (Objective 1.4)