Nested Classes

QUIZ

```
public class MyOuter {
public static class MyInner {
public static void foo() { }
}
}
```

which statement, if placed in a class *other* than MyOuter or MyInner, instantiates an instance of the nested class?

A. MyOuter.MyInner m = new MyOuter.MyInner();

B. MyOuter.MyInner mi = new MyInner();

C. MyOuter m = new MyOuter();
MyOuter.MyInner mi = m.new MyOuter.MyInner();

D. MyInner mi = new MyOuter.MyInner();

Which two are true about a static nested class?

A. You must have a reference to an instance of the enclosing class in order to instantiate it.

B. It does not have access to nonstatic members of the enclosing class.

C. Its variables and methods must be static.

D. It can be instantiated using new MyOuter.MyInner();.

E. It must extend the enclosing class.

```
Class A
{
    int x=12;
    static class B
    {
                void m1(){ //print x}
    }
}
```

Will the above code compile ?

. Given the following,

```
class Boo {
Boo(String s) { }
Boo() { }
}
class Bar extends Boo {
Bar() { }
Bar(String s) {super(s);}
void zoo() {
// insert code here
}
}
```

which two create an anonymous inner class from within class Bar? (Choose two.)

A. Boo f = new Boo(“24”) { };

B. Boo f = new Bar() { };

C. Boo f = new Boo() {String s; };

D. Bar f = new Boo(String s) { };

E. Boo f = new Boo.Bar(String s) { };

Given the following,

```
1.class Foo {
2. class Bar{ }
3.}
4.class Test {
5. public static void main (String [] args) {
6. Foo f = new Foo();
7. // Insert code here
8. }
9.}
```

which statement, inserted at line 7, creates an instance of Bar?

A. Foo.Bar b = new Foo.Bar();

B. Foo.Bar b = f.new Bar();

C. Bar b = new f.Bar();

D. Bar b = f.new Bar();

E. Foo.Bar b = new f.Bar();

Which two are true about a method-local inner class?

A. It must be marked final.

B. It can be marked abstract.

C. It can be marked public.

D. It can be marked static.

E. It can access private members of the enclosing class.

. Which is true about an anonymous inner class?

A. It can extend exactly one class and implement exactly one interface.

B. It can extend exactly one class and can implement multiple interfaces.

C. It can extend exactly one class or implement exactly one interface.

D. It can implement multiple interfaces regardless of whether it also extends a class.

E. It can implement multiple interfaces if it does not extend a class.

```
1.public class TestObj {
2. public static void main (String [] args) {
3. Object o = new Object() {
4. public boolean equals(Object obj) {
5. return true;
6. }
7. }
8. System.out.println(o.equals("Fred"));
9. }
10.}
```

what is the result?

A. An exception occurs at runtime.

B. true

C. fred

D. Compilation fails because of an error on line 3.

E. Compilation fails because of an error on line 4.

F. Compilation fails because of an error on line 8.

G. Compilation fails because of an error on a line other than 3, 4, or 8.

```
1. public class HorseTest {
2. public static void main (String [] args) {
3. class Horse {
4. public String name;
5. public Horse(String s) {
6. name = s;
7. }
8. }
9. Object obj = new Horse("Zippo");
10. Horse h = (Horse) obj;
11. System.out.println(h.name);
12. }
13. }
```

what is the result?

A. An exception occurs at runtime at line 10.
B. Zippo
C. Compilation fails because of an error on line 3.
D. Compilation fails because of an error on line 9.
E. Compilation fails because of an error on line 10.
F. Compilation fails because of an error on line 11.

11. Given the following,

```
1. public class HorseTest {
2. public static void main (String [] args) {
3. class Horse {
4. public String name;
5. public Horse(String s) {
6. name = s;
7. }
8. }
9. Object obj = new Horse("Zippo");
10. System.out.println(obj.name);
11. }
12. }
```

what is the result?

A. An exception occurs at runtime at line 10.

B. Zippo

C. Compilation fails because of an error on line 3.

D. Compilation fails because of an error on line 9.

E. Compilation fails because of an error on line 10.