1. Given

class Foo

{

class Bar{ }

}

class Test

{

public static void main (String [] args)

{

Foo f = new Foo();

/\* Line 10: Missing statement ? \*/

}

}

which statement, inserted at line 10, creates an instance of Bar?

1. Foo.Bar b = new Foo.Bar();
2. Foo.Bar b = f.new Bar();
3. Bar b = new f.Bar();
4. Bar b = f.new Bar();

Answer: b

1. Which statement is true about a static nested class?
2. You must have a reference to an instance of the enclosing class in order to instantiate it.
3. It does not have access to nonstatic members of the enclosing class.
4. It's variables and methods must be static.
5. It must extend the enclosing class.

Answer: b

1. What modifiers can be used with inner classes?
2. abstract
3. public
4. static
5. final

Answer: a, b, d

1. Which statement(s) about inner classes is (are) TRUE?
2. It must be declared final
3. It can only be used in enclosing class
4. Enclosing class can also be inner class
5. It is member of enclosing class

Answer: c, d

1. Which statement is true about an inner class?
2. You must have a reference to an instance of the enclosing class in order to instantiate it.
3. It does not have access to non-static members of the enclosing class.
4. It's variables and methods must be static.
5. It must extend the enclosing class.

Answer: a

1. Given

class Foo

{

static class Bar{ }

}

class Test

{

public static void main (String [] args)

{

Foo f = new Foo();

/\* Line 10: Missing statement ? \*/

}

}

which statement, inserted at line 10, creates an instance of Bar?

1. Foo.Bar b = new Foo.Bar();
2. Foo.Bar b = f.new Bar();
3. Bar b = new f.Bar();
4. Bar b = f.new Bar();

Answer: a

1. Given

class OuterClass {

public int var = 0;

public class InnerClass {

public void output(){

/\* Line 5: missing statement \*/

}

}

}

Which statement(s) inserted at line 5 will output var?

1. System.out.println(OuterClass.var);
2. System.out.println(OuterClass.this.var);
3. System.out.println(var);
4. System.out.println(this.var);

Answer: b, c

1. Consider inner interface. Which access level can it have?
2. public
3. private
4. protected
5. package private

Answer: a, b, c, d

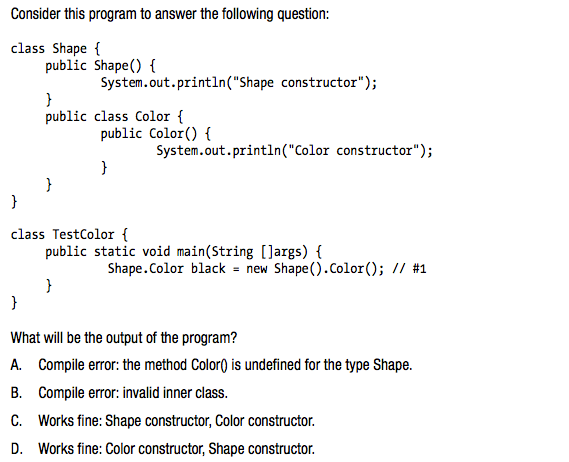
1. Which statements about inner interfaces are TRUE?
2. Its implementation must also be inner class of the same enclosing class
3. It can be declared static
4. It can contain other inner interfaces
5. It cannot contain inner classes.

Answer: b, c

1. Which of the statements are FALSE about nested interfaces?
2. They must be declared static
3. They cannot be declared in inner class
4. They can contain inner classes
5. They can contain nested interfaces

Answer: a, c,

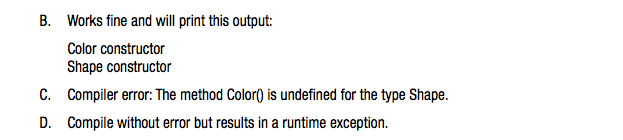
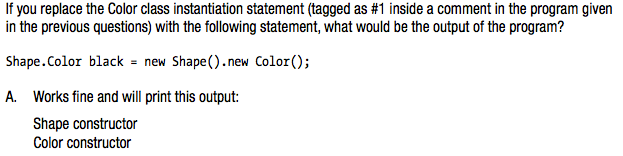
11)



Answer: a. Compile error: the method Color() is undefined for the type shape.

(You need to create an instance of outer class shape in order to create an inner class instance).

12)

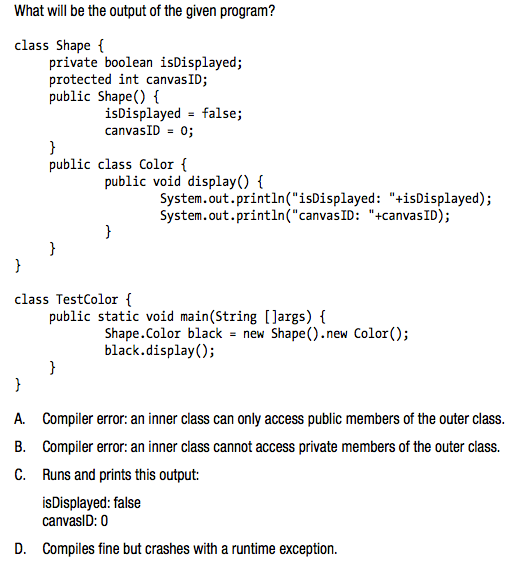


Answer: a. Works fine and will print this output:

Shape constructor

Color constructor

13)



**Answer**: C. runs and prints this output:

isdisplayed: false canvasid: 0

(an inner class can access all members of an outer class, including the private members of the outer class).

14)

Результат выполнения?

class Outer {

protected int x = 5;

private class Inner extends Outer {

public Inner() {

x = 3;

}

public void method() {

System.out.println(Outer.this.x);

System.out.println(x);

}

}

public static void main(String[] args) {

new Outer().new Inner().method();

}

}

1. 3 5
2. 3 3
3. 5 3
4. 5 5
5. Compilation error

**Answer:** C15)

Результат выполнения?

class Outer2 {

static public class Inner {

public class InInner {

public int y;

}

}

}

class Test {

public static void main(String[] args) {

System.out.println(Outer2.Inner.new InInner().y);

}

}

1. 0
2. Initialization error (Compilation error)
3. Cannot find symbol (Compilation error)

**Answer**: c

16)

В каких строках ошибки компиляции?

class Mult {

public Mult getMultiplier(int x) { //1

return new Mult() { //2

public void multiply() { //3

System.out.println(x \* 2); //4

}

};

}

}

class BlaBla {

public static void main(String[] args) {

new Mult().getMultiplier(3).multiply(); //5

}

}

1. 1, 2
2. 4
3. 4 5
4. 5
5. 2

**Answer**: c. 4 строка потому что переменная x должна быть финальной, 5 строка из-за того, что метод multiply вне зоны видимости (в классе Mult он не объявлен).

17)

Будет ли ошибка компиляции?

interface Interface {

class Clazz {

interface Interface2 {

int x = 4;

}

}

class MyClass implements Clazz.Interface2 {

}

}

class InterfaceTest {

public static void main(String[] args) {

System.out.println(new Interface.MyClass().x);

}

}

1. ага
2. не

Answer: b. Все нормально, будет выведено «4». Clazz – статичный класс, внутри которого интерфейс (который всегда статичен).

18\*)

Что произойдет?

public class A {

public static class X {

public static class Y {

public static String Z = "life is good";

}

public static C Y;

}

public static class C {

public static String Z = "life is pain";

}

public static void main(String[] args) {

System.out.println(X.Y.Z);

}

}

1. Compile error
2. Runtime error
3. Output: “life is good”
4. Output: “life is pain”

**Answer**: d. У полей приоритет.