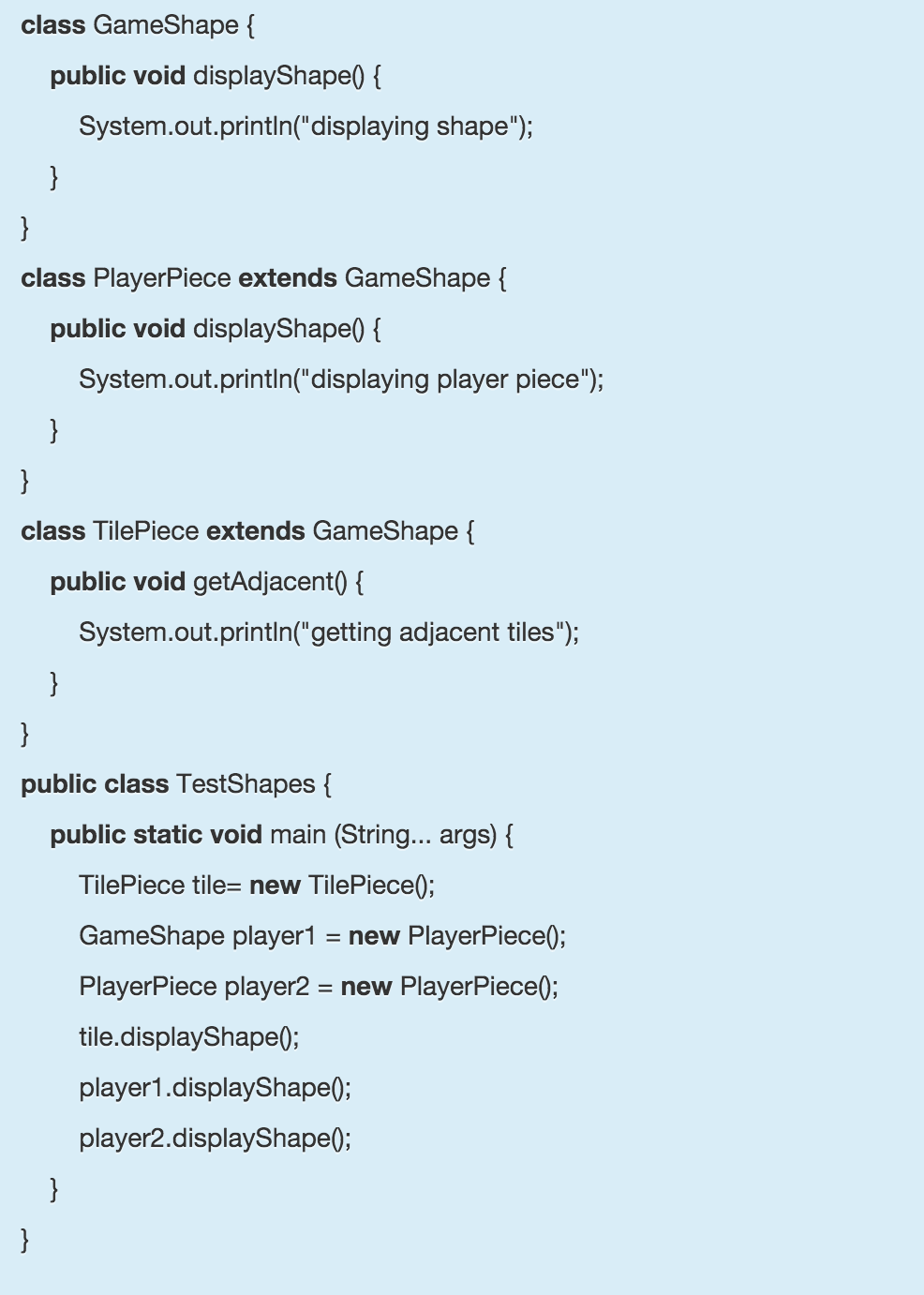
1. You need to write a method which prints the average of an array of integers. How could the method signature look like?
   1. public double displayAverage(int[] a)
   2. public int displayAverage(int[] a)
   3. public void displayAverage(int[] a)
2. Which one is true about final classes?
   1. final classes cannot be extended.
   2. final classes can be extended.
   3. final classes cannot be implemented.
3. Which of the following operators can be used in conjunction with a String object?
   1. +=
   2. \*
   3. +
   4. ++
   5. -
4. Which is true?
   1. "X implements Y" is correct if and only if X is an interface and Y is a class
   2. "X implements Y" is correct if and only if X is a class and Y is an interface
   3. "X implements Y" is correct if X and Y are either both classes or both interfaces.
   4. "X implements Y" is correct for all combinations of X and Y being classes and / or interfaces.
5. Given:

public class Frodo extends Hobbit {  
  
 public static void main(String[] args) {  
 int myGold = 5;  
 int result = new Frodo().countGold(myGold, 5);  
 System.out.println(result + “ : ” +myGold);  
 }  
}  
  
class Hobbit {  
  
 int countGold(int x, int y) {  
 return x + y;  
 }  
}

* 1. Compilation fails
  2. 10 : 10
  3. 10 : 5
  4. 5 : 10
  5. 0 : 5
  6. An exception is thrown

1. Given the following code:

What is the output of the program?

* 1. displaying shape

displaying shape

displaying player piece

* 1. displaying shape

displaying player piece

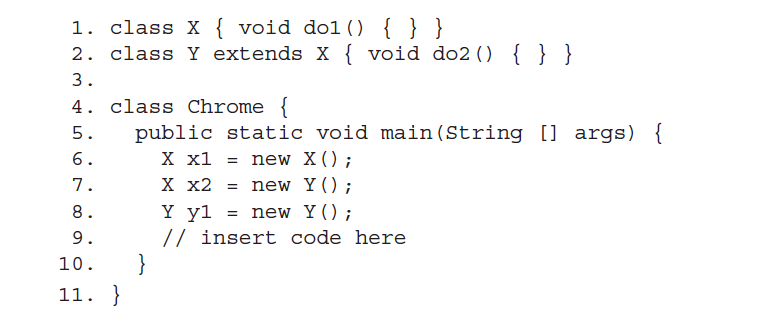
displaying player piece

* 1. displaying shape  
     displaying shape  
     displaying shape
  2. displaying player piece

displaying shape

displaying player piece

1. An abstract class can contain only abstract methods.
   1. true
   2. false
2. Given the following:

Which, inserted at line 9, will compile?

* 1. x2.do2();
  2. x2.do1();
  3. x1.do2();
  4. y1.do2();
  5. None of the above will compile

1. The first statement of every constructor must be a call to either this() (an overloaded constructor) or super().

Note: the call may be implicit (added by the compiler), or explicit (added by the developer).  
Decide if the above is true or false:

* 1. true
  2. false

1. class A{ }

class B { B(){ } }

class C{ C(String s){ } }  
  
Which of the following classes have a default constructor?

* 1. A
  2. A and B
  3. B
  4. C
  5. B and C

1. Given:

class OverloadingTest{

void m1(int x){

System.out.println("m1 int");

}

void m1(double x){

System.out.println("m1 double");

}

void m1(String x){

System.out.println("m1 String");

}

}

public class TestClass {

public static void main(String[] args) {

OverloadingTest ot = new OverloadingTest();

ot.m1(1.0);

}

}

What will be the output?

* 1. It will fail to compile
  2. m1 int
  3. m1 double
  4. m1 String

1. Which of the following are true regarding overloading of a method?
   1. An overloading method must have a different parameter list and same return type as that of the overloaded method.
   2. If there is another method with the same name but with a different number of arguments in a class then that method can be called as overloaded.
   3. If there is another method with the same name and same number and type of arguments but with a different return type in a class then that method can be called as overloaded.
   4. An overloaded method means a method with the same name and same number and type of arguments exists in the super class and sub class.
2. You are writing a class named Bandwidth for an internet service provider that keeps track of number of bytes consumed by a user. The following code illustrates the expected usage of this class:   
     
   class User {

Bandwidth bw = new Bandwidth();

public void consume(int bytesUsed) {

bw.addUsage(bytesUsed);

}

// ... other irrelevant code

}

class Bandwidth {

private int totalUsage;

private double totalBill;

private double costPerByte;

//add your code here

// ...other irrelevant code

}

Your goal is to implement a method addUsage (and other methods, if required) in Bandwidth class such that all the bandwidth used by a User is reflected by the totalUsage field and totalBill is always equal to totalUsage\*costPerByte. Further, that a User should not be able to tamper with the totalBill value and is also not able to reduce it.   
  
Which of the following implementation(s) accomplishes the above?

* 1. public void addUsage(int bytesUsed){

if(bytesUsed>0){

totalUsage = totalUsage + bytesUsed;

totalBill = totalBill + bytesUsed\*costPerByte;

}

}

* 1. protected void addUsage(int bytesUsed){  
      totalUsage += bytesUsed;   
      totalBill = totalBill + bytesUsed\*costPerByte;   
     }
  2. private void addUsage(int bytesUsed){  
      if(bytesUsed>0){   
      totalUsage = totalUsage + bytesUsed;   
      totalBill = totalUsage\*costPerByte;   
      }  
     }
  3. public void addUsage(int bytesUsed){

if(bytesUsed>0){

totalUsage = totalUsage + bytesUsed;

}

}

public void updateTotalBill(){   
 totalBill = totalUsage\*costPerByte;

}

1. Which of these statements are true?
   1. A static method can call other non-static methods in the same class by using the 'this' keyword.
   2. A class may contain both static and non-static variables and both static and non-static methods.
   3. Each object of a class has its own copy of each non-static member variable.
   4. Instance methods may access local variables of static methods.
   5. All methods in a class are implicitly passed a 'this' parameter when called.