1.) Which of the following assignments are valid?

//the rule is subclass inherits from superclass therefore subclass "IS A" superclass based on inheritance

//also constructor methods run first from the top of hierarchy then down each constructor method in the //hierarchy. new RedBloodCell() constructs 1st. a Cell, 2nd. a BloodCell then 3rd. a RedBloodCell

Cell c = **new** Cell(); //OK cell object is a cell type

Cell c = **new** BloodCell(); //OK bloodcell object descended from cell and "IS A" cell type thru inheritance

Cell c = **new** RedBloodCell(); //OK redbloodcell object descended from cell and "IS A" cell thru inheritance

BloodCell c = new Cell(); //cell object does not inherit from red blood cell. type mismatch

BloodCell c = **new** BloodCell(); //OK bloodcell object is a bloodcell type

BloodCell c = **new** RedBloodCell(); //OK redbloodcell object descended from bloodcell and "IS A" bloodcell type based on inheritance

RedBloodCell c = new Cell(); //cell object does not inherit from redbloodcell. type mismatch

RedBloodCell c = new BloodCell(); //blood cell object does not inherit from redbloodcell. type mismatch

RedBloodCell c = **new** RedBloodCell(); //OK redbloodcell object is a redbloodcell type

2.) Explain the difference between the “this” keyword and the “super” keyword.

“this” refers to methods/data in the current class. “Super” refers to methods/data in the parent class.

3.) What I the difference between an “IS-A” and a “HAS-A” relationship? How do you create a “HAS-A” relationship in your code?

Has-A means an object of one class “has a” reference to an object of another and/or same class. It is also known as “composition”. There is no specific keyword to implement HAS-A relationship but mostly we use the “new” keyword.

4.) Imagine you are going to write a program to play card games…

The Card class would be better implemented composed of a value (ace, 2, …, king) and type (spade, diamond, heart, club).

5.) Modified true or false: if false explain.

a. FALSE A subclass’s method can explicitly call a superclass’s method using the super (not this) keyword.  
b. TRUE An abstract class can have abstract and defined methods but an interface cannot HAVE ANY CODE.  
c. FALSE ~~A~~n abstract class has constructors but an interface cannot have any constructors. Interfaces have only STATIC fields and declared methods.  
d. TRUE A class can only extend 0 or 1 other class. A class cannot extend multiple classes.  
e. FALSE A class can implement 0, 1 or many interface(s).

6.)   
 -------------------------------------  
 | Car |   
 | ============ |  
 | |  
 | toString() “vroom” |  
 | m1() “car 1” |  
 | m2() “car 2” |  
 | |  
 -------------------------------------  
 |  
 |  
 -------------------------------------  
 | Truck |   
 | ============ |  
 | |  
 | toString() inherit from parent |  
 | m1() “truck1” |  
 | m2() inherit from parent |  
 | |  
 -------------------------------------

Car myCar = **new** Car(); //nothing prints in myCar instantiation

Truck myTruck = **new** Truck();//nothing prints in myTruck instantiation

System.***out***.println(myCar); //prints “vroom” System.out.println(Object) method will call toString() of Car

myCar.m1(); //prints “car 1” per m1() method in Car

myCar.m2(); //prints “car 2” per m2() method in Car

System.***out***.println(myTruck); //prints “vroom" System.out.println(Object) inherits toString() of Car

myTruck.m1(); //prints “truck1” for m1() method in Truck

myTruck.m2(); //prints "car 2" m2() method inherited from Car

7.)  
 -------------------------------------  
 | SeaCreature |   
 | ============ |  
 | |  
 | toString() “ocean-dwelling” |  
 | method1() “creature 1” |  
 | method2() “creature 2” |  
 | |  
 -------------------------------------  
 |  
 |  
 ---------------------------------------------  
 | |  
 | |  
------------------------------------- -------------------------------------  
| Squid | | Mammal |   
| ============ | | ============ |  
| | | |  
| toString() “squid” | | toString() inherit from parent |  
| method1() inherit from parent | | method1() “warm-blooded” |  
| method2() “tentacles” | | method2() inherit from parent |  
| | | |  
------------------------------------- -------------------------------------  
 |  
 |  
 -------------------------------------  
 | Whale |   
 | ============ |  
 | |  
 | toString() “BIG!” |  
 | method1() “spout” |  
 | method2() inherit from parent |  
 | |  
 -------------------------------------  
  
  
The FOR loop calls toString(), method1, and method2()   
for each object in the SeaCreature array of elements.  
The elements in the array are;  
0. Squid  
1. Whale  
2. SeaCreature  
3. Mammal  
This is what is written to the console;  
----------------------------------------------------  
squid  
creature 1  
tentacles

BIG!

spout

creature 2

ocean-dwelling

creature 1

creature 2

ocean-dwelling

warm-blooded

creature 2