CSE143 Practice Final

1. Examine the following class & interface definitions:

```
interface X { … }
interface Y { … }
public class A implements X{ … }
public class B extends A implements Y { ... }
public class C extends B { … }
public class D extends A { … }
public class E extends Exception { … }
```

Part A
Which of the following statements are legal statements (e.g. if the statement were in main(), it would compile)? You may assume that all classes have a default constructor.

Hint: it may help to draw a picture of the inheritance/interface hierarchy to help you visualize the relationships between various structures.

a. A myA = new B(); Y N

b. A myA = new C(); Y N

c. C myC = new A(); Y N

d. D myD = new C(); Y N

e. X myX = new X(); Y N

f. X myX = new A(); Y N

g. X myX = new C(); Y N

h. Y myY = new A(); Y N

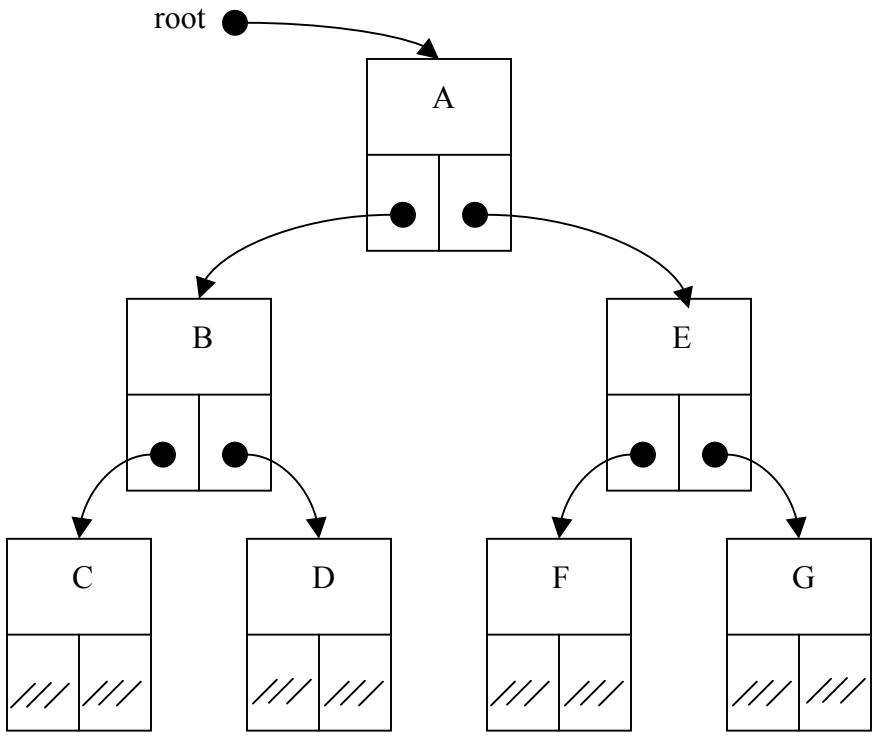
i. Y myY = new D(); Y N

j. Exception myE = new E(); Y N

Part B
Suppose that we now change A into an abstract class. Circle the letters corresponding to the statements above that *were legal before*, but are illegal now that A is abstract:

a b c d e f g h i j

2. A binary tree is a very common data structure that we haven't yet seen in class. Trees are a hierarchical organization of elements, and are used to store items just like lists or stacks. There is one element at the top of the tree, called the *root*. This element stores a single item, and also has references to two *child* elements, a left child and a right child. These child elements are themselves binary trees. Thus a binary tree looks like this (the nodes at the bottom level :



Part A
Which of the two lists from class does the binary tree remind you of (Recursive or Linked)? Why?

Part B
Write the class definition for a BinaryTree. The only method you need to write is the constructor—so you'll have to think about what a new, empty binary tree should look like.

```
public class BinaryTree {




   public BinaryTree() {




   }
}
```

Part C
A binary tree is useless if we can't see what's inside! Write a method to print out the value of each item in the tree. On the tree given above, your print() method should produce:

A B C D E F G

```
public void print() {









}
```

Part D ("extra credit")
Write another printing method, print2() for the BinaryTree. The difference between print() and print2() is that print2() should produce the following output on the tree given in the example:

G F E D C B A

```
public void print2() {









}
```

3. This question is a series of short-answers about the following problem:

> You're opening up a new restaurant called Grab-A-Byte. First, you know you need to find a set of suppliers to provide your restaurant with raw ingredients every week; however, since you're new at this, you're not yet sure who your suppliers will be. Second, you know you need to find yourself a set of chefs: you need a head chef who knows how to cook and develop original recipes, and you need four other chefs who only need to know how to cook. Third, you need to register your name with the Seattle Owners of Unionized Restaurants (SOUR). This is easy, since there's a single registry book that's maintained collectively by all restaurants in the Seattle area—you just have to add your name to this shared book.

Being the stellar programmer that you are, you decide to model your new restaurant in Java. But first, you must figure out the following:

Part A. What Java tool/concept is the most appropriate for defining the relationship between your restaurant and your suppliers? Why?

Part B. What Java tool/concept is the most appropriate for defining the relationship between the two kinds of chefs? Why?

Part C. What Java tool/concept is the most appropriate for modeling the SOUR registry? Why?

Part D. You decide to create a new class Chef to represent chefs within your restaurant model. Name one advantage of making Chef an inner class of your Restaurant class:

Name one advantage of making Chef an independent, public class:

Part E. Your friend Tubby has been successfully running his restaurant, Cup-Of-Java, for many years. He hears you're opening a restaurant and gives you a tip about chefs. He says that the longer they've worked for you, the more loyal they get. As a result, the only chefs that ever quit are the ones most recently hired. He warns that it's still possible to lose all your chefs, if they all quit before you're able to hire a new one; but, a chef will never quit unless everyone hired after him has already quit.

What data structure should you use to store your set of Chefs? Why?

4.

Part A. Name two major responsibilities that each of the following components must fulfill:

Model:

1.

2.

View:

1.

2.

Controller:

1.

2.

Part B. Why is it useful to have the model separated from the view?