ASSIGNMENT 1

Question 1:

a)Yes-This is because the instance of method will return true because when dynamic binding will take place in java it will analyses the current class and then proceed to check the parent classes. One of the superclasses of fruit is Apple and the parent class of Apple is Fruit. So, since it has the Fruit class as a superclass it inherits features and can we call it an instance

b) No-GoldenDelicious has no relation to the Orange because GoldenDelicious does not inherit anything from Orange and vice versa and they are not the same class, so fruit is not an instance of Orange.

c) Yes-fruit is an instance of Apple because fruit is of the object GoldenDelicious which is the subclass of the Apple class so it fruit comes from Apple due to this inheritance relation it will be an instance.

d) Yes-The answer is yes because fruit is an object of type GoldenDelicious so it would be an instance of GoldenDelicious because it is in the GoldenDelicious is its own class type, so the object does come from the class itself.

e) No-fruit has no derivation or relation to McIntosh because they only subclasses of Apples, but they do not affect each other thru inheritance so fruit is not an instance of McIntosh.

f) Yes-orange is an instance of Orange because the Orange is the class type of orange, so it comes from its own class, so it is an instance because it is an instance of if the class at question is a superclass or it’s own class type. So orange is of class type Orange and so its true

g) Yes-Fruit is the superclass of orange, so orange inherits from Fruit, so orange is an instance of orange. Therefore, it is an instance because of the inheritance relationship.

h) No-orange is not an instance of Apple because orange is neither derived nor related through an inheritance relationship. It can also seen that orange is not of the Apple class type so it can be said that orange is not an instance of Apple because it is of Orange class type.

i) Yes-fruit can invoke the makeAppleCider() method because of the inheritance relation. The class type GoldDelicious inherits the features, methods, and attributes of the Apple class due to inheritance so it can be invoked because Apple is the super class to GoldenDelicious.

j) No-This is due to the fact that orange is in the Orange class which does not inherit the makeAppleCider() because it has no relation to the Apple class other than the fact that they have the same super class which will make no difference so no the method will not be invoked.

k) No-fruit cannot invoke the makeOrangeJuice(); method because fruit is of class type GoldenDelicious which does not inherit the method . Orange is neither the superclass nor the class itself so it will not inherit the method so it cannot be invoked

l)Yes-orange can invoke the makeOrangeJuice(); method because it is in its own class. The orange object reference is of class type Orange which does contain and can invoke the makeOrangeJuice(); method so it can be invoked.

m) No-According to the 1st rule of polymorphism the reference variable type has to be that of the super class and the object has to the sub type but in this class, Orange has no relevant relationship in this case to happen so it will be an error so it is not legal and it is not the supertype to Apple.

n) No-According to the 1st rule of polymorphism the reference variable must be of the type superclass and the declaration/object must be that of the type subclass type .Therefore the code mentioned is not legal due to the fact that the reference is that of McIntosh which is the subclass to Apple which is the superclass. A correct form would be Apple p=new McIntosh(); which is legal.

o) Yes-This is legal because it follows the 1st rule of polymorphism because Apple is the superclass type and McIntosh is the subclass type. Therefore, this is legal.

Question 2:

a) The following for-each loop will loop through all the fruit objects in the array of objects and employ the saySomething();.The output will be :

I am a Fruit

I am an Apple

I am an Orange

I am a GoldenDelicious

I am a McIntosh

b) ((apple)f2).saySomething(“Greetings”);

c)No-The following code is not valid for the following reason. As seen in the code displayed in the assignment (Fruit[] apples=new Apple[3];) ,this code creates an array of references which are meant to refer to apple objects once instantiated. When we write apples[0]=new Orange(); we are playing an object of Orange type in that empty memory location which is meant for an Apple type object .Java will allow this to run but as the code runs it will produce a runtime error because the code would be trying to put an incorrect object in the Apple reference. So this would not be legal because it for that instance Fruit f2 would be an Orange. In conclusion, we can say that the following statement is not valid. A valid statement may only be where the object Apple or one of the subclasses to Apple.