CS159 Review Activity

Name(s): ____

1. (6 points) Draw a UML diagram corresponding to the accompanying `Locatable` interface and the `Building`, `Car`, and `ApartmentBuilding` classes. For full credit, you must use correct UML "syntax", including arrow types, etc. You may use underlining in place of italics where appropriate. YOU DO NOT NEED TO INCLUDE INSTANCE VARIABLES OR METHODS IN YOUR DIAGRAM.

2. (15 points) Given the `Locatable` interface and the `Building`, `Car`, and `ApartmentBuilding` classes and the following declarations, which of the following will compile (C) or will not compile (N).

```
Object obj;
Locatable place;
Building bld;
Car auto;
```

(a) _____ `obj = new Locatable(38.443, -78.812);`
(b) _____ `auto = new Car("Chevy", 36.257, -81.321);`
(c) _____ `place = new ApartmentBuilding("91 E. Grace St.", 38.441, -78.871, 26);`
(d) _____ `bld = new ApartmentBuilding("91 E. Grace St.", 38.441, -78.871, 26);`
(e) _____ `bld = new Car("Chevy", 36.257, -81.321);`
(f) _____ `bld = new Building("701 Carrier Drive", 38.434,-78.863);`

Assuming that `place` contains an appropriate object, which of the following statements will compile(C) and which will not(N).

(g) _____ `place.longitude = 36.54;`
(h) _____ `double lat = place.getLatitude();`
(i) _____ `System.out.println(place.equals("some string"));`
(j) _____ `place.setLatitude(83.4);`

Assume that we are rewriting the `toString` method of the `ApartmentBuilding` class.

```
public String toString()
{
    String result = "";

    // Statements go here.
}
```

Which of the following statements would compile(C) and which will not(N) when included in this method?

(k) _____ `result += this.address;`
(l) _____ `result += getLongitude();`
(m) _____ `result += latitude;`
(n) _____ `result += getNumUnits();`
(o) _____ `result += Building.getAddress();`

3. (8 points) What output will be produced by the following program, `Driver.java`, given the code on the reference pages?

```
public class Driver
{
    public static void main(String[] args)
    {
        Building bld;
        ApartmentBuilding apt;
        bld = new ApartmentBuilding("18 Ohio", 37.441,
                                    -78.873, 26);
        apt = (ApartmentBuilding) bld;

        System.out.println(bld.toString());
        System.out.println(apt.toString());
        printInfo(bld);
        printInfo(apt);
    }

    public static void printInfo(Building place)
    {
        System.out.println("BUILDING: " + place.toString());
    }

    public static void printInfo(ApartmentBuilding place)
    {
        System.out.println("APARTMENT: " + place.toString());
    }

}
```

4. Write a utility method named `arcticOnly` that takes an `ArrayList` of `Locateable` objects named `global`, and returns a new ArrayList of `Locateable` objects that only includes the the `Locateable`s from `global` that have a latitude greater than 66.56.

```
public interface Locatable
{
    public double getLatitude();
    public double getLongitude();
}
```

```
public class Car implements Locatable
{
    private String model;
    private double latitude;
    private double longitude;

    public Car(String model, double latitude, double longitude)
    {
        this.model = model;
        setLocation(latitude, longitude);
    }

    public String getModel()
    {
        return model;
    }

    public double getLatitude()
    {
        return latitude;
    }

    public double getLongitude()
    {
        return longitude;
    }

    public void setLocation(double latitude, double longitude)
    {
        this.latitude = latitude;
        this.longitude = longitude;
    }

}
```

```
public abstract class Building implements Locatable
{
    protected String address;
    private final double latitude;
    private final double longitude;

    public Building(String address, double latitude, double longitude)
    {
        this.address = address;
        this.latitude = latitude;
        this.longitude = longitude;
    }

    public double getLatitude()
    {
        return latitude;
    }

    public double getLongitude()
    {
        return longitude;
    }

    public String getAddress()
    {
        return address;
    }

    public String toString()
    {
        return address;
    }
}
```

```
public class ApartmentBuilding extends Building
{
    private int numUnits;

    public ApartmentBuilding(String address, double latitude,
                              double longitude, int numUnits)
    {
        super(address, latitude, longitude);
        this.numUnits = numUnits;
    }

    public int getNumUnits()
    {
        return numUnits;
    }

    public String toString()
    {
        return address + " (" + numUnits + " units)";

    }
}
```