CSCI 2320
Fall 2018
Exam 1

Last Name: _____

First Name: _____

Write your answers on the answer sheets provided. NOTE: Some questions on this exam assume the existence of the following class:

```
public class Student
{
   public String name;
   public double gpa;
   public String toString { return name+" "+gpa; }
}
```

1. (2 pts each) Briefly define each of these terms:

   (a) `static` (when applied to a method in a class)

   (b) constructor

   (c) pure object model

2. (6 pts) The method below is intended to display all `Student` objects that have a GPA attribute at least as high as the provided parameter. The method returns the number of Student objects displayed. Rewrite the method and add JavaDoc-conforming comments so that the method adheres to the Java Programming Conventions discussed in class. In your rewrite, you should choose better identifier names and follow Java naming conventions. NOTE: `Student[] data` and `int n` are assumed to be attributes of the class so do not change their names.

   ```
   public int Numbers(double run)
   {
      int i,x=0;
      for (i=0; i<n; i++)
         if (data[i].gpa >= run) {
            System.out.println(data[i]);
            x++;
         }
      return x;
   }
   ```

3. NOTE: This question is focused on non-OOP use of classes. Suppose you have been asked by the Major League Baseball commissioner to create a program to track statistics of various teams. In doing so you have created the following class to represent a baseball player:

```
public class Player
{
   String name;
   int atBats;
   int baseHits;
   int rbis;    // runs batted in
}
```

(a) (4 pts) Write a section of code to create and initialize a `Player` object to represent a player named "Fred" who has has 120 at bats, 30 base hits, and 25 RBIs.

(b) (6 pts) Write a method call `mostBaseHits` that accepts an array of `Player` objects along with the number of elements in the array as parameters. The method should print the name of the player who has the most base hits.

4. In this problem we will continue working with the `Player` class but will modify it to be more in line with a typical object-oriented approach.

(a) (2 pts) Rewrite the class so that its attributes are private.

(b) (4 pts) Write a constructor for the class which will set the name of the player to the value of the parameter and will zero out everything else.

(c) (6 pts) Write a method called `battingAvg()` that will calculate and return as a double the player's batting average (defined to be the number of base hits divided by the number of times at bat). If the player hasn't had any at bats then the method should return 0.0. WARNING: an `int` divided by an `int` is always an `int` (unless you typecast).

(d) (6 pts) Write a getter method and a setter method for the attribute named `rbis`.

(e) (4 pts) Write a `toString` method that returns the player's name followed by their runs batted in followed by a slash followed by their batting average.

(f) (4 pts) Suppose, in your workspace, you had just modified the `Player` class as described above. Write the sequence of git statements you would use to record your work and post it to your bitbucket.org homework account.

(g) (4 pts) Draw a UML diagram that depicts the new design of the `Player` class.

(h) Suppose the `Player` class has been correctly implemented as described in this problem and your are ready to test it in `main()`:

i. (4 pts) Create an array of 20 `Player` objects whose names are "Player 1", "Player 2", etc.

ii. (2 pts) Suppose the array has been completely filled with objects. Write code to display the last entry in the array.

iii. (6 pts) Write code to save all data in the completely full array to a file named players.txt.