AP Computer Science A
Free Response
Mr. Neat

Consider a grade-averaging scheme in which the final average of a student’s scores is computed differently from the traditional average if the scores have “improved”. Scores have improved if each score is greater than or equal to the previous score. The final average of the scores is computed as follows.

A student has n scores indexed from 0 to n-1. If the scores have improved, only those scores with indexes greater than or equal to n/2 are averaged. If the scores have not improved, all the scores are averaged.

Consider the following incomplete CVRecord class declaration. Each CVRecord object stores a list of that student’s scores, stores that student’s name and contains methods to compute that student’s final average.

Complete the CVRecord class by filling in all of the missing code segments. In addition, write a client program that uses the CVRecord class and produces the following output.

```
Drake's scores are: 50 50 20 80 53 average is: 50.6
President Biden's scores are: 20 50 50 53 80 average is: 61.0
LeBron's scores are: 20 50 50 80 average is: 65.0
```

Note: In writing finalAverage, you must call the appropriate methods defined in the class. Assume that these methods work as specified. No credit will be given for code that re-implements these methods rather than calling these methods.

```
class CVRecord {

    private int[] scores;
    private String name;

    public CVRecord(int[] sc, String es)
    {
        scores = sc;
        name = es;
    }


    // findMax() finds and returns the highest score in the array scores

    public int findMax()
    {
        // implement this method
    }




    // hasImproved() returns true if each successive value in scores is greater
    // than or equal to the previous value;
    // otherwise, returns false

    private boolean hasImproved()
    {
        // implement this method
    }

    // average()  returns the average (arithmetic mean) of the values in scores
    // whose subscripts are between first and last, inclusive
    // precondition: 0 <= first <= last < scores.length

    private double average(int first, int last)
    {
        // implement this method
    }

    // if the values in scores have improved, finalAverage() returns the average
    // of the elements in scores with indexes greater than or equal to scores.length/2;
    // otherwise, returns the average of all of the values in scores

    public double finalAverage()
    {
        // implement this method
    }
    public String toString()
    {
        String temp = "";
        for(int i=0; i<scores.length;i++)
        {
            temp = temp+" "+scores[i];
        }
        return name+"'s scores are:" + temp;
    }
}
```