4. A school district would like to get some statistics on its students' standardized test scores. Scores will be represented as objects of the following `ScoreInfo` class. Each `ScoreInfo` object contains a score value and the number of students who earned that score.

```
public class ScoreInfo
{
  private int score;
  private int numStudents;

  public ScoreInfo(int aScore)
  {
    score = aScore;
    numStudents = 1;
  }

  /** adds 1 to the number of students who earned this score
   */
  public void increment()
  {  numStudents++;  }

  /** @return this score
   */
  public int getScore()
  {  return score;  }

  /** @return the number of students who earned this score
   */
  public int getFrequency()
  {  return numStudents;  }
}
```

The following `Stats` class creates and maintains a database of student score information. The scores are stored in sorted order in the database.

```
public class Stats
{
  private ArrayList<ScoreInfo> scoreList;
    // listed in increasing score order; no two ScoreInfo objects contain the same score

  /** Records a score in the database, keeping the database in increasing score order. If no other
   *  ScoreInfo object represents score, a new ScoreInfo object representing score
   *  is added to the database; otherwise, the frequency in the ScoreInfo object representing
   *  score is incremented.
   *  @param score a score to be recorded in the list
   *  @return true if a new ScoreInfo object representing score was added to the list;
   *          false otherwise
   */
  public boolean record(int score)
  { /*  to be implemented in part (a)  */}

  /** Records all scores in stuScores in the database, keeping the database in increasing score order
   *  @param stuScores an array of student test scores
   */
  public void recordScores(int[] stuScores)
  {  /*  to be implemented in part (b)  */  }

  // There may be instance variables, constructors, and methods that are not shown.
}
```

(a) Write the `Stats` method `record` that takes a test score and records that score in the database. If the score already exists in the database, the frequency of that score is updated. If the score does not exist in the database, a new `ScoreInfo` object is created and inserted in the appropriate position so that the database is maintained in increasing score order. The method returns `true` if a new `ScoreInfo` object was added to the database; otherwise, it returns `false`.

Complete method `record` below.

```
/** Records a score in the database, keeping the database in increasing score order. If no other
 *  ScoreInfo object represents score, a new ScoreInfo object representing score
 *  is added to the database; otherwise, the frequency in the ScoreInfo object representing
 *  score is incremented.
 *  @param score a score to be recorded in the list
 *  @return true if a new ScoreInfo object representing score was added to the list;
 *          false otherwise
 */
public boolean record(int score)
```

(b) Write the `Stats` method `recordScores` that takes an array of test scores and records them in the database. The database contains at most one `ScoreInfo` object per unique score value. Each `ScoreInfo` object contains a score and an associated frequency. The database is maintained in increasing order based on the score.

In writing `recordScores`, assume that `record` works as specified, regardless of what you wrote in part (a).

Complete method `recordScores` below.

```
/** Records all scores in stuScores in the database, keeping the database in increasing score order
 *  @param stuScores an array of student test scores
 */
public void recordScores(int[] stuScores)
```

STOP

END OF EXAM