Users of a website are asked to provide a review of the website at the end of each visit. Each review, represented by an object of the `Review` class, consists of an integer indicating the user's rating of the website and an optional String comment field. The comment field in a `Review` object ends with a period ("."), exclamation point ("!"), or letter, or is a `String` of length 0 if the user did not enter a comment.

```java
public class Review
{
    private int rating;
    private String comment;
    //   Precondition: r >= 0
    //   c is not null.
    public Review(int r, String c)
    {
        rating = r;
        comment = c;
    }
public int getRating()
{
    return rating;
}
public String getComment()
{
    return comment;
}
//  There may be instance variables, constructors, and methods that are not shown.
}
```

The `ReviewAnalysis` class contains methods used to analyze the reviews provided by users. You will write two methods of the `ReviewAnalysis` class.

```
public class ReviewAnalysis
{
  /** All user reviews to be included in this analysis */
    private Review[] allReviews;
  /** Initializes allReviews to contain all the Review objects to be analyzed */
    public ReviewAnalysis()
    {
      /* implementation not shown */
    }
    public double getAverageRating()
    { /* to be implemented in part (a) */ }
    public ArrayList<String> collectComments()
    { /* to be implemented in part (b) */ }
}
```

a. Write the `ReviewAnalysis` method `getAverageRating`, which returns the average rating (arithmetic mean) of all elements of `allReviews`. For example, `getAverageRating` would return `3.4` if `allReviews` contained the following `Review` objects.

0	1	2	3	4
4 "Good! Thx"	3 "OK site"	5 "Great!"	2 "Poor! Bad."	3 ""

Complete method `getAverageRating`.

```
/** Returns a double representing the average rating of all the Review objects
to be
*       analyzed, as described in part (a)
*       Precondition: allReviews contains at least one Review.
*       No element of allReviews is null.
*/
public double getAverageRating()
```
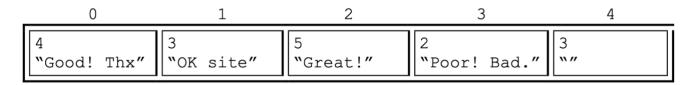
b. Write the `ReviewAnalysis` method `collectComments`, which collects and formats only comments that contain an exclamation point. The method returns an `ArrayList` of `String` objects containing copies of user comments from `allReviews` that contain an exclamation point, formatted as follows. An empty `ArrayList` is returned if no comment in `allReviews` contains an exclamation point.

The `String` inserted into the `ArrayList` to be returned begins with the index of the `Review` in `allReviews`.

- The index is immediately followed by a hyphen ("-").
- The hyphen is followed by a copy of the original comment.
- The `String` must end with either a period or an exclamation point. If the original comment from `allReviews` does not end in either a period or an exclamation point, a period is added.

The following example of `allReviews` is repeated from part (a).

0	1	2	3	4
4 "Good! Thx"	3 "OK site"	5 "Great!"	2 "Poor! Bad."	3 ""

The following `ArrayList` would be returned by a call to `collectComments` with the given contents of `allReviews`. The reviews at index 1 and index 4 in `allReviews` are not included in the `ArrayList` to return since neither review contains an exclamation point.

"0-Good! Thx."	"2-Great!"	"3-Poor! Bad."

Complete method `collectComments`.

```
/**  Returns an ArrayList of String objects containing formatted versions of selected
*    user comments, as described in part (b)
*        Precondition: allReviews contains at least one Review.
*        No element of allReviews is null.
*        Postcondition: allReviews is unchanged.
*/
public ArrayList<String> collectComments()
{
```