Question 1

(a) Part

```
public int getScore() {
    int totalScore = 0;

    if (levelOne.goalReached()) {
        totalScore = totalScore + levelOne.getPoints();

        if (levelTwo.goalReached()) {
            totalScore = totalScore + levelTwo.getPoints();

            if (levelThree.goalReached()) {
                totalScore = totalScore + levelThree.getPoints();
            }
        }
    }

    if (isBonus()) {
        totalScore = totalScore * 3;
    }

    return totalScore;
}
```

(b) Part

```
public int playManyTimes(int num) {
    int highest = 0;

    for (int i = 0; i < num; i++) {
        play();
        int score = getScore();
        if (score > highest) {
            highest = score;
        }
    }

    return highest;
}
```

Question 2

```
public class Textbook extends Book {
    private int edition;

    public Textbook(String n, int p, int e) {
        super(n, p);
        edition = e;
    }

    public boolean canSubstituteFor(Textbook other) {
        if (other.getTitle().equals(getTitle()) && edition >= other.getEdition()) {
            return true;
        } else {
            return false;
        }
```

```
    }

    public int getEdition() {
        return edition;
    }

    public String getBookInfo() {
        return super.getBookInfo() + "-" + edition;
    }
}
```

Question 3

(a) Part

```
public double getAverageRating() {
    int totalRating;

    for(int index = 0; index < allReviews.length; index++) {
        totalRating = totalRating + allReviews[index].getRating();
    }

    return (double) totalRating / allReviews.length;
}
```

(b) Part

```
public ArrayList<String> collectComments() {
    ArrayList<String> collectedComments = new ArrayList<String>();
    for(int index = 0; index < allReviews.length; index++) {
        if (allReviews[index].getComment().indexOf("!") != -1) {
            String newStr = new String(index + "-" + allReviews[index].getComment());
            String lastPart = newStr.substring(newStr.length - 1, newStr.length);

            if (!lastPart.equals("!") && !lastPart.equals(".")) {
                newStr = newStr + "."
            }

            collectedComments.add(newStr);
        }
    }

    return collectedComments;
}
```

Question 4

(a) Part

```
public void repopulate() {
    for (int row : grid) {
        for (int col : grid[row]) {
            int randomValue = (int) (Math.random() * MAX) + 1;
            while (randomValue % 10 != 0 || randomValue % 100 == 0) {
                randomValue = (int) (Math.random() * MAX) + 1;
            }

            grid[row][col] = randomValue;
```

```
        }
    }
}
```

(b) Part

```
public int countIncreasingCols() {
    int count = 0;

    for (int col = 0; col < grid[0].length; col++) {
        boolean ordered = true;

        for (int row = 1; row < grid.length; row++) {
            if (grid[row][col] < grid[row-1][col]) {
                ordered = false
            }
        }

        if (ordered == true) {
            count++;
        }
    }

    return count;
}
```