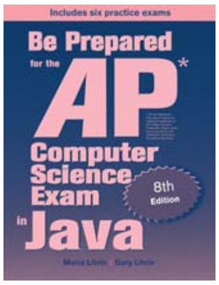

Eighth Edition

Be Prepared for the AP Computer Science Exam in Java

Chapter 6: Annotated Solutions to Past Free-Response Questions

2022

Maria Litvin
Phillips Academy, Andover, Massachusetts

Gary Litvin
Skylight Publishing, Andover, Massachusetts

Skylight Publishing
Andover, Massachusetts

Library of Congress Control Number: 2021950662

ISBN 978-0-9972528-7-3

Skylight Publishing
9 Bartlet Street, Suite 70
Andover, MA 01810

web: www.skylit.com
e-mail: sales@skylit.com
support@skylit.com

www.skylit.com/beprepared/x2022all.zip contains complete Java code, including solutions and test programs for runnable projects.

The free-response questions for this exam are posted on apstudent.collegeboard.org and, for teachers, on AP Central:

- For students: `apstudents.collegeboard.org`
- For teachers: `apcentral.collegeboard.org`

Scoring guidelines for teachers are usually posted over the summer.

Question 1

Part (a)

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

  if (levelOne.goalReached())
  {
    score += levelOne.getPoints();
    if (levelTwo.goalReached()) [1]
    {
      score += levelTwo.getPoints();
      if (levelThree.goalReached())
      {
        score += levelThree.getPoints();
      }
    }
  }

  if (isBonus())
    score *= 3;

  return score;
}
```

Notes:

1. Not

```
  if (levelOne.goalReached())
  {
    score += levelOne.getPoints();
  }
  if (levelTwo.goalReached())
  {
    score += levelTwo.getPoints();
  }
  ...
```

Part (b)

```
public int playManyTimes(int num)
{
  int bestScore = 0;
  for (int i = 0; i < num; i++)
  {
    play();
    int currentScore = getScore();
    if (currentScore > bestScore)
      bestScore = currentScore; 1
  }
  return bestScore;
}
```

Notes:

1. If you know the `Math.max` method, you can write simply

   ```
   bestScore = Math.max(bestScore, getScore());
   ```

   to update `bestScore`.

Question 2

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

  public Textbook(String bookTitle, double bookPrice, int ed)
  {
    super(bookTitle, bookPrice);
    edition = ed;
  }

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

  public int getEdition()  [1]
  {
    return edition;
  }

  public boolean canSubstituteFor(Textbook other)
  {
    return getTitle().equals(other.getTitle()) &&
                       getEdition() >= other.getEdition();  [2]
  }
}
```

Notes:

1. The `getEdition` method is not mentioned in the `Textbook` class description, but it is mentioned in the table of examples and it logically belongs in this class.

2. Also works

```
return getTitle().equals(other.getTitle()) &&
                   edition >= other.edition;
```

   It is a common misconception to think that `other.edition` is not accessible here because it is private in `Textbook`. "privacy" applies to the code of the class, not to individual objects of the class.

Question 3

Part (a)

```
public double getAverageRating()
{
  double totalScore = 0;

  for (Review r : allReviews)
    totalScore += r.getRating();

  return totalScore / allReviews.length;
} 1
```

Notes:

1. Or:

```
public double getAverageRating()
{
  int totalScore = 0;
  for (Review r : allReviews)
    totalScore += r.getRating();
  return (double)totalScore / allReviews.length;
}
```

Part (b)

```
public ArrayList<String> collectComments()
{
  ArrayList<String> comments = new ArrayList<String>();

  for (int i = 0; i < allReviews.length; i++)
  {
    String s = allReviews[i].getComment();

    if (s.indexOf("!") >= 0)
    {
      String last = s.substring(s.length() - 1);
      if (!last.equals(".") && !last.equals("!")) 1
        s += ".";

      comments.add(i + "-" + s);
    }
  }
  return comments;
}
```

Notes:

1. Or (not in the Java "AP subset" but OK):

```
      if (!s.endsWith(".") && !s.endsWith("!"))
        s += ".";
```

Question 4

Part (a)

```
public void repopulate()
{
  for (int r = 0; r < grid.length; r++)
  {
    for (int c = 0; c < grid[0].length; c++)
    {
      int x = 0;
      while (x % 100 == 0)
         x = 10*((int)((MAX/10)*Math.random()) + 1); 1, 2

      grid[r][c] = x;
    }
  }
} 3
```

Notes:

1. The value of `MAX` is "not shown." It is safe to assume that `MAX` >= 10; otherwise there would be no acceptable values to fill the grid.

2. This expression gives a random number from 1 to `MAX` that is divisible by 10. Not too many of such numbers will be divisible by 100, so we can just wait for one that is not. Also acceptable (but very inefficient):

```
int x = 0;
while (x % 10 != 0 || x % 100 == 0)
   x = (int)(MAX*Math.random()) + 1;
grid[r][c] = x;
```

3. An alternative approach is to first create a list of all eligible numbers, then randomly choose a number from that list to put into the grid:

```
ArrayList<Integer> randomNumbers = new ArrayList<Integer>();
for (int x = 10; x <= MAX; x += 10)
  if (x % 100 != 0)
    randomNumbers.add(x);

for (int r = 0; r < grid.length; r++)
{
  for (int c = 0; c < grid[0].length; c++)
  {
    int i = (int)(randomNumbers.size()*Math.random());
    grid[r][c] = randomNumbers.get(i);
  }
}
```

Part (b)

```
public int countIncreasingCols()
{
  int count = 0;
  for (int col = 0; col < grid[0].length; col++)
  {
    boolean increasing = true;
    for (int row = 1; row < grid.length && increasing; row++)
    {
      if (grid[row][col] < grid[row-1][col])
        increasing = false;
    }
    if (increasing) 1
      count++;
  }
  return count;
} 2
```

Notes:

1. There is no need for a separate check for a grid with one row: if `grid.length` is 1, the `for` loop is never entered, and `increasing` remains true, so the count is incremented for each "column."

2. Normally `countIncreasingCols` would use a helper method:

```
private boolean isIncreasing(int col)
{
  for (int row = 1; row < grid.length; row++)
    if (grid[row][col] < grid[row-1][col])
      return false;
  return true;
}

public int countIncreasingCols()
{
  int count = 0;
  for (int col = 0; col < grid[0].length; col++)
    if (isIncreasing(col))
      count++;
  return count;
}
```

Solutions with a helper method will get full credit.