



Eighth Edition

Be Prepared
for the
AP
Computer Science
Exam in Java

Chapter 6: Annotated Solutions
to Past Free-Response Questions

2014

Maria Litvin

Phillips Academy, Andover, Massachusetts

Gary Litvin

Skylight Publishing, Andover, Massachusetts

Skylight Publishing
Andover, Massachusetts

**Copyright © 2014-2022 by
Maria Litvin, Gary Litvin, and Skylight Publishing**

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the authors and Skylight Publishing.

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

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

- For students: apstudent.collegeboard.org
- For teachers: apcentral.collegeboard.org/courses

Scoring guidelines are usually posted over the summer.

The www.skylit.com/beprepared/x2014all.zip file contains complete Java classes that include solutions and test programs for runnable projects.

Question 1

Part (a)

```
public static String scrambleWord(String word)
{
    String scrambled = "";
    int i = 0;

    while (i < word.length())
    {
        String letter1 = word.substring(i, i+1);
        String letter2 = "";

        if (i < word.length() - 1)
            letter2 = word.substring(i+1, i+2);

        if (letter1.equals("A") && !letter2.equals("A") &&
            !letter2.equals(""))
        {
            scrambled += letter2 + letter1;
            i += 2;
        }
        else
        {
            scrambled += letter1;
            i += 1;
        }
    }
    return scrambled;
}
```

Part (b)

```
public static void scrambleOrRemove(List<String> wordList)
{
    int i = 0;
    while (i < wordList.size())
    {
        String word = wordList.get(i);
        String scrambled = scrambleWord(word);

        if (!scrambled.equals(word))
        {
            wordList.set(i, scrambled);
            i++;
        }
        else
            wordList.remove(i);
    }
}
```

Notes:

1. Alternative solution:

```
public static void scrambleOrRemove(List<String> wordList)
{
    for (int i = wordList.size() - 1; i >= 0; i--)
    {
        String word = wordList.get(i);
        String scrambled = scrambleWord(word);

        if (!scrambled.equals(word))
            wordList.set(i, scrambled);
        else
            wordList.remove(i);
    }
}
```

Question 2

```
public class Director extends Rock
{
    public Director()
    {
        super(Color.RED);
    }

    public void act()
    {
        if (getColor().equals(Color.GREEN))
        {
            ArrayList<Actor> neighbors =
                getGrid().getNeighbors(getLocation());
            for (Actor a : neighbors)
                a.setDirection(a.getDirection() + Location.RIGHT); 1
            setColor(Color.RED);
        }
        else
            setColor(Color.GREEN);
    }
}
```

Notes:

1. Or, simply:

```
a.setDirection(a.getDirection() + 90);
```

Question 3

Part (a)

```
public SeatingChart(List<Student> studentList, int rows, int cols)
{
    seats = new Student[rows][cols]; 1

    int r = 0, c = 0;

    for (Student s : studentList)
    {
        seats[r][c] = s;
        r++;
        if (r == rows)
        {
            r = 0;
            c++;
        }
    } 2
}
```

Notes:

1. When an array of objects is declared, all elements are set to null by default. This fact is not in the AP subset, but knowing it saves a couple of lines of code.
2. Alternative solution:

```
public SeatingChart(List<Student> studentList, int rows,
                    int cols)
{
    seats = new Student[rows][cols];

    for (int i = 0; i < studentList.size(); i++)
        seats[i % rows][i / rows] = studentList.get(i);
}
```

Part (b)

```
public int removeAbsentStudents(int allowedAbsences)
{
    int removedCount = 0;
    for (int r = 0; r < seats.length; r++)
    {
        for (int c = 0; c < seats[0].length; c++)
        {
            if (seats[r][c] != null &&
                seats[r][c].getAbsenceCount() > allowedAbsences) 1
            {
                seats[r][c] = null;
                removedCount++;
            }
        }
    }
    return removedCount;
}
```

Notes:

1. It is important to check first that the value `seats[r][c]` is not null. Failure to do so may result in `NullPointerException` when `getAbsenceCount` is called.

Question 4

```
public class Trio implements MenuItem
{
    private Sandwich sandwich;
    private Salad salad;
    private Drink drink;

    public Trio(Sandwich aSandwich, Salad aSalad, Drink aDrink)
    {
        sandwich = aSandwich;
        salad = aSalad;
        drink = aDrink;
    }

    public String getName()
    {
        return sandwich.getName() + "/" + salad.getName() +
            "/" + drink.getName() + " Trio";
    }

    public double getPrice()
    {
        double price1 = sandwich.getPrice();
        double price2 = salad.getPrice();
        double price3 = drink.getPrice();
        double smallest = Math.min(Math.min(price1, price2), price3);
        return price1 + price2 + price3 - smallest;
    }
}
```

Notes:

1. `Math.min` is not in the AP subset but is handy here. Alternatively you can write, of course:

```
double smallest = price1;

if (price2 < smallest)
    smallest = price2;

if (price3 < smallest)
    smallest = price3;
```