



Ninth Edition

Be Prepared
for the
AP
Computer Science
Exam in Java

Chapter 5: Annotated Solutions
to Past Free-Response Questions

2023

Maria Litvin

Phillips Academy, Andover, Massachusetts

Gary Litvin

Skylight Publishing, Andover, Massachusetts

Skylight Publishing
Andover, Massachusetts

**Copyright © 2026 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.

ISBN 978-0-9972528-3-5

Skylight Publishing
14 Lincoln Street
Andover, MA 01810

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

www.skylit.com/beprepared/x2023all.zip contains complete Java code, including solutions and test programs for runnable projects.

The free-response questions for this exam are posted on AP Central:

apcentral.collegeboard.org

Scoring guidelines for teachers are usually posted over the summer.

Question 1

Part (a)

```
public int findFreeBlock(int period, int duration)
{
    int startMinute = -1;

    for (int minute = 0; minute < 60; minute++)
    {
        if (isMinuteFree(period, minute))
        {
            if (startMinute == -1)
                startMinute = minute;

            if (minute - startMinute + 1 >= duration)
                return startMinute;
        }
        else
        {
            startMinute = -1;
        }
    }

    return -1;
} 1,2
```

Notes:

1. A variation (courtesy John Schlamann, Hinsdale Township HS):

```
public int findFreeBlock(int period, int duration)
{
    int count = 0;

    for (int minute = 0; minute < 60; minute++)
    {
        if (isMinuteFree(period, minute))
        {
            count++;
            if (count >= duration)
                return minute - count + 1;
        }
        else
        {
            count = 0;
        }
    }

    return -1;
}
```

2. Alternative solution with a nested loop (less efficient) — `isMinuteFree` is called multiple times for the same minute:

```
public int findFreeBlock(int period, int duration)
{
    for (int startMinute = 0; startMinute <= 60 - duration;
        startMinute++)
    {
        int found = startMinute;

        for (int minute = startMinute;
            minute < startMinute + duration && found >= 0; minute++)
            // && found is optional
        {
            if (!isMinuteFree(period, minute))
                found = -1;
        }

        if (found >= 0)
            return found;
    }
    return -1;
}
```

Part (b)

```
public boolean makeAppointment(int startPeriod, int endPeriod,
                               int duration)
{
    for (int period = startPeriod; period <= endPeriod; period++)
    {
        int startMinute = findFreeBlock(period, duration);
        if (startMinute >= 0)
        {
            reserveBlock(period, startMinute, duration);
            return true;
        }
    }
    return false;
}
```

Question 2

```
public class Sign
{
    private String text;
    private int width;

    public Sign(String tx, int w)
    {
        text = tx;
        width = w;
    }

    public int numberOfLines()
    {
        int len = text.length();

        if (len % width == 0)
            return len / width;
        else
            return len / width + 1;
    }

    public String getLines()
    {
        if (text.length() == 0)
            return null;

        String outputText = "";1,2

        int lineNum = 0;
        while (lineNum < numberOfLines() - 1)
        {
            int i = lineNum * width;
            outputText += text.substring(i, i + width) + ";";
            lineNum++;
        }
        outputText += text.substring(lineNum * width);3
        return outputText;
    }
}
```

Notes:

1. The question states that repeated calls to `getLines` return the same result, so the original text must remain unchanged.
2. No need to create an array of substrings — they can be gathered in the resulting string.
3. Don't forget the last line.

Question 3

Part (a)

```
public void cleanData(double lower, double upper)
{
    int i = 0;

    while (i < temperatures.size())
    {
        double temp = temperatures.get(i); 1
        if (temp >= lower && temp <= upper)
            i++;
        else
            temperatures.remove(i);
    }
} 2
```

Notes:

1. Double is automatically converted to double due to unboxing.
2. Or:

```
public void cleanData(double lower, double upper)
{
    for (int i = temperatures.size() - 1; i >= 0; i--)
    {
        double temp = temperatures.get(i);

        if (temp < lower || temp > upper)
            temperatures.remove(i);
    }
}
```

Part (b)

```
public int longestHeatWave(double threshold)
{
    int maxLength = 1; 1
    int currentLength = 0;

    for (int i = 0; i < temperatures.size(); i++)
    {
        if (temperatures.get(i) > threshold)
        {
            currentLength++;
            if (currentLength > maxLength)
                maxLength = currentLength;
        }
        else
        {
            currentLength = 0;
        }
    }
    return maxLength;
}
```

Notes:

1. A “heat wave” must be at least two days.

Question 4

Part (a)

```
public boolean moveCandyToFirstRow(int col)
{
    if (box[0][col] != null)
        return true;

    for (int row = 1; row < box.length; row++)
    {
        if (box[row][col] != null)
        {
            box[0][col] = box[row][col];
            box[row][col] = null;
            return true;
        }
    }
    return false;
}
```

Part (b)

```
public Candy removeNextByFlavor(String flavor)
{
    for (int row = box.length - 1; row >= 0; row--)
    {
        for (int col = 0; col < box[row].length; col++)1
        {
            Candy candy = box[row][col];2
            if (candy != null && candy.getFlavor().equals(flavor))3
            {
                box[row][col] = null;
                return candy;
            }
        }
    }
    return null;
}
```

Notes:

1. Or: `box[0].length` because `box` is a rectangular 2D array.
2. Save `box[row][col]` before setting it to `null`.
3. Don't forget to check for `null` first: you can't call `null`'s methods.