Design and Analysis of Algorithms **Pgm Ex #1**
CS3230 **14 March 2016**

CS3230: Assignment 1
Find the Center of Mass of Images

1 Objective

The purpose of this assignment is for you to practice the following skills which you are expected to master in this course:

- Algorithm design;
- Correctness analysis;
- Efficiency analysis;
- Implementation of algorithms.

2 Problem

Background

Some algorithms for optical character recognition compare a scanned image with templates of "perfect" characters. Part of the difficulty with such comparisons is deciding where to start the comparison. This is because the characters in the scanned image are subject to noise and distortion, resulting in changes in size, position, and orientation. A procedure that is sometimes used to deal with changes in position matches the center of mass of the scanned character and the templates against which it is compared. In this problem you are to determine the center of mass of the scanned image.

For our purposes, a scanned image will be a square array of real numbers, each of which represents the gray-scale value of a point in a scanned image. Each gray-scale value will be between 0 (representing a totally white region) and 1 (representing a totally black region). The center of mass is a particular element of an array. Suppose a center of mass is in the ith row and jth column. Then the **difference** between the sum of the elements of the two array portions above and below the ith row is **minimal**. Likewise, the difference of the sums of the elements in the two array portions to the left and to the right of the jth column is minimal.

Consider the array shown below (5×5), which might have resulted from scanning a small image. The center of mass for this array is uniquely in row 3, column 3. The difference of the sum of the elements in each array portion formed by ignoring the third row is 0.1 (the sums are 5.55 and 5.65). The difference of the sum of each array portion formed by ignoring the third column is 0.00 (the sums are both 5.60). Note that in general the center of mass will **not** be in the center of the matrix.

0.70	0.75	0.70	0.75	0.80
0.55	0.30	0.20	0.10	0.70
0.80	0.10	0.00	0.00	0.80
0.70	0.00	0.00	0.00	0.80
0.80	0.90	0.80	0.75	0.90

Your Task

Write a program that computes and outputs the center of mass for images of size 25×25. To receive full marks your program's efficiency should be of $O(n^2)$. You can write your program either in C/C++ or in Java. The task has been set up in the **CodeCrunch** programming environment, which we are using for the submission of your program. You can access CodeCrunch for either C/C++ or Java through the following two links:

```
https://codecrunch.comp.nus.edu.sg/task_view.php?tid=2558
```

```
https://codecrunch.comp.nus.edu.sg/task_view.php?tid=2559
```

Input

The input to your program is a square matrix with dimensions 25×25. The matrix represents a scanned image. *Your program needs to read the input from STDIN.* Several test cases have been set up on **CodeCrunch** and when you compile and execute your program there the system will tell you whether your program generates the correct results and thus passes the tests.

Note: We will test your program with some additional input cases.

Output

Your program needs to generate a plain text output written to STDOUT. The output must only contain 1 line with 3 values, i.e., "i j z" (use a space as separator between the values). Here i and j denote the row and column of the center of mass, respectively (two integer numbers), and z refers to the value at this position (*a real number, written with 2 decimal digits after the decimal point*). *Note: rows and columns are sequentially numbered starting from 1.*

Note: If there is more than one center of mass, the one with the *largest* row and column should be output.

Note that your program's output will be scored **automatically** on **CodeCrunch**. It is therefore very important that your output format is as described above. Otherwise your output may be mis-judged.

3 Deliverables

- A Java or C/C++ program with the name "Main.java" or "main.c" (or "main.cpp") submitted via **CodeCrunch**. Please make sure that your program reads its input from STDIN and writes its results to STDOUT.

- A short report (12pt font, at most 3 A4 pages – max. 2 pages are preferred), containing the following:

  - Your name and matric number;
  - A description of your algorithm (you can use pseudo-code or plain English);
  - A correctness proof for your algorithm;
  - A time complexity analysis for your algorithm;
  - (Optional) Any other information that you feel is important about your algorithm and implementation.

  Your report should be in PDF format and named as U080887X_Ex1.pdf, where U080887X is to be replaced by your matric number. Remember to write your name and matric number at the beginning of your report. Submit your report to IVLE (see below).

4 Submission

- Submit your program, Main.java, main.c or main.cpp via CodeCrunch before **23:59** on Thursday **17 March 2016**. (We will know your program from your login information.)
- Submit your U080887X_Ex1.pdf report to the IVLE Workbin folder called **Ex1** before **23:59** on Thursday **17 March 2016**.
- Late submission: submit to the folder **Ex1Late** before **23:59** on **18 March 2016**, but 30% marks will be deducted for such a late submission. The same applies for the submission of your program.
- Late submission: submit to the folder **Ex1Latest** before **23:59** on **19 March 2016**, but 60% marks will be deducted for such a late submission. The same applies for the submission of your program.
- No submissions will be accepted after **23:59** of **19 March 2016**.

5 Grading Policy

This exercise is worth 14% of your overall grade, and is graded on a 100 point basis, with 50 pts for your program and another 50 pts for your report.

Grading of Program

The program will be automatically graded on the **CodeCrunch** server. Therefore, please make sure that your program is compilable on CodeCrunch (with Java or C/C++). Before you submit your program, please test it on **CodeCrunch** server. **Note that we do not have time to debug your program! No marks will be given for this part if your program does not compile and run on the CodeCrunch.**

Note that we will let your program run for a maximum amount of time during scoring. Some programs never finish and hence our automatic testing procedure will terminate such programs after what we think is sufficient time to compute the output.

Five input files will be used for grading. Some of the files are already loaded for testing on the **CodeCrunch** server. The others will be randomly generated. Each correct output is worth 10 pts. You will gain all the 50 pts of this part only when your program can handle all five input files.

Grading of Report

The 50 pts of this part are distributed as follows:

- An $O(n^2)$ algorithm (20 pts); n is the dimension of the input matrix.
- Correctness proof (15 pts).
- Complexity analysis (15 pts).

The main grading criteria are correctness, clarity and rigor.

Hint

You may want to think about this problem in 1 dimension before you move to 2 dimensions.

Please note: Plagiarism will not be tolerated!