Assignment 1: Line Drawing in Fortran and COBOL

Due: 19:00, Sun 14 Feb 2016 Full marks: 100

Introduction

In this assignment, you will implement a program to raster segments of straight lines to output. The program reads a sequence of data points in 2-D space, connects the points one by one with straight lines, and then output the lines. You will write the program *once using Fortran* and *once using COBOL*. You also need to submit a simple report (less than one page) to tell us your experience in using the two languages. The aim of this assignment is to let you have a taste of some old fashioned languages and have an understanding of what problems programmers encountered in the past.

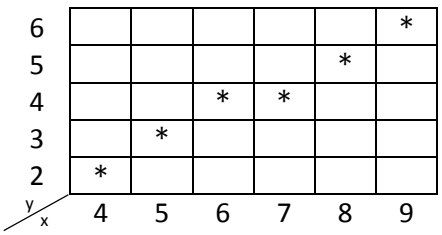
Digital Differential Analyzer (DDA)

The straight line connecting two distinct points $p_i = (x_i, y_j)$ and $p_j = (x_j, y_j)$ can be represented using the equation $y - y_i = m(x - x_i)$, where $m = \frac{y_j - y_i}{x_j - x_i}$ is the *slope* of the line. Suppose p_i and p_j contain *integer* coordinates only, in order to raster the straight line between p_i and p_j to a matrix (like a picture of pixels, the console, a text file, etc.), we need to approximate the straight line by interpolating the points between p_i and p_j with *integer* coordinates also. The *digital differential analyzer* (DDA) is an algorithm for such purpose. It considers two cases depending on the slope m.

Case 1: $|m| \leq 1$
Assume without loss of generality that $x_i < x_j$. We sample the integer x-coordinates in the interval $[x_i, x_j]$ in steps of 1, that is, $x_i, x_i + 1, x_i + 2, x_i + 3, \dots, x_j$. Then, the y-coordinates are interpolated as: $(x_i, y_i), \big(x_i + 1, rd(y_i + m)\big), \big(x_i + 2, rd(y_i + 2m)\big), \big(x_i + 3, rd(y_i + 3m)\big), \dots, (x_j, y_j)$, where $rd(k)$ means the round-off of k to the nearest integer (四捨五入; 5 or above is rounded up, 4 or below is truncated).

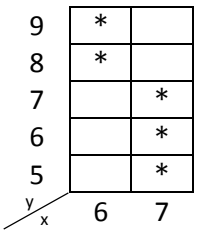
Case 2: $|m| > 1$
Assume without loss of generality that $y_i < y_j$. We sample the integer y-coordinates in the interval $[y_i, y_j]$ in steps of 1, that is, $y_i, y_i + 1, y_i + 2, y_i + 3, \dots, y_j$. Then, the x-coordinates are interpolated as: $(x_i, y_i), \big(rd(x_i + \frac{1}{m}), y_i + 1\big), \big(rd\big(x_i + \frac{2}{m}\big), y_i + 2\big), \big(rd\big(x_i + \frac{3}{m}\big), y_i + 3\big), \dots, (x_j, y_j)$.

Example 1. Two points $(4,2)$ and $(9,6)$. Slope $m = \frac{6-2}{9-4} = 0.8$. (Case 1)
The interpolated points would be $(4,2), \big(5, rd(2.8)\big), \big(6, rd(3.6)\big), \big(7, rd(4.4)\big), \big(8, rd(5.2)\big), (9,6)$. That is, $(4,2), (5,3), (6,4), (7,4), (8,5), (9,6)$. The following shows a plot of the points.



Copyright © 2016 CSE, CUHK

Example 2. Two points $(7,5)$ and $(6,9)$. Slope $m = \frac{9-5}{6-7} = -4$. (Case 2)
The interpolated points would be $(7,5), (rd(6.75), 6), (rd(6.5), 7), (rd(6.25), 8), (6,9)$. That is, $(7,5), (7,6), (7,7), (6,8), (6,9)$. The following shows a plot of the points.



Program Specification

You are required to write two programs, one in Fortran (file name `dda.for`) and the other in COBOL (file name `dda.cob`). The program flow and other requirements are specified below.

Program Flow

- The programs read n points $p_1, p_2, p_3, \ldots, p_n$ from an input file which must be named `input.txt`. You can assume that the input file always has the following format:

```
<n>
<x_1>␣<y_1>
<x_2>␣<y_2>
…
<x_n>␣<y_n>
```

  The first line contains an integer specifying the number of data points. Then in each of the subsequent `<n>` lines, there are two integers `<x_i>` and `<y_i>`, separated by a space character ␣, specifying the x-coordinate and y-coordinate of data point p_i. Besides, all integers in the input file must have a fixed width of 2. The following shows an example input file specifying five data points $(3,2)$, $(40,22)$, $(32,5)$, $(28,12)$, and $(0,15)$:

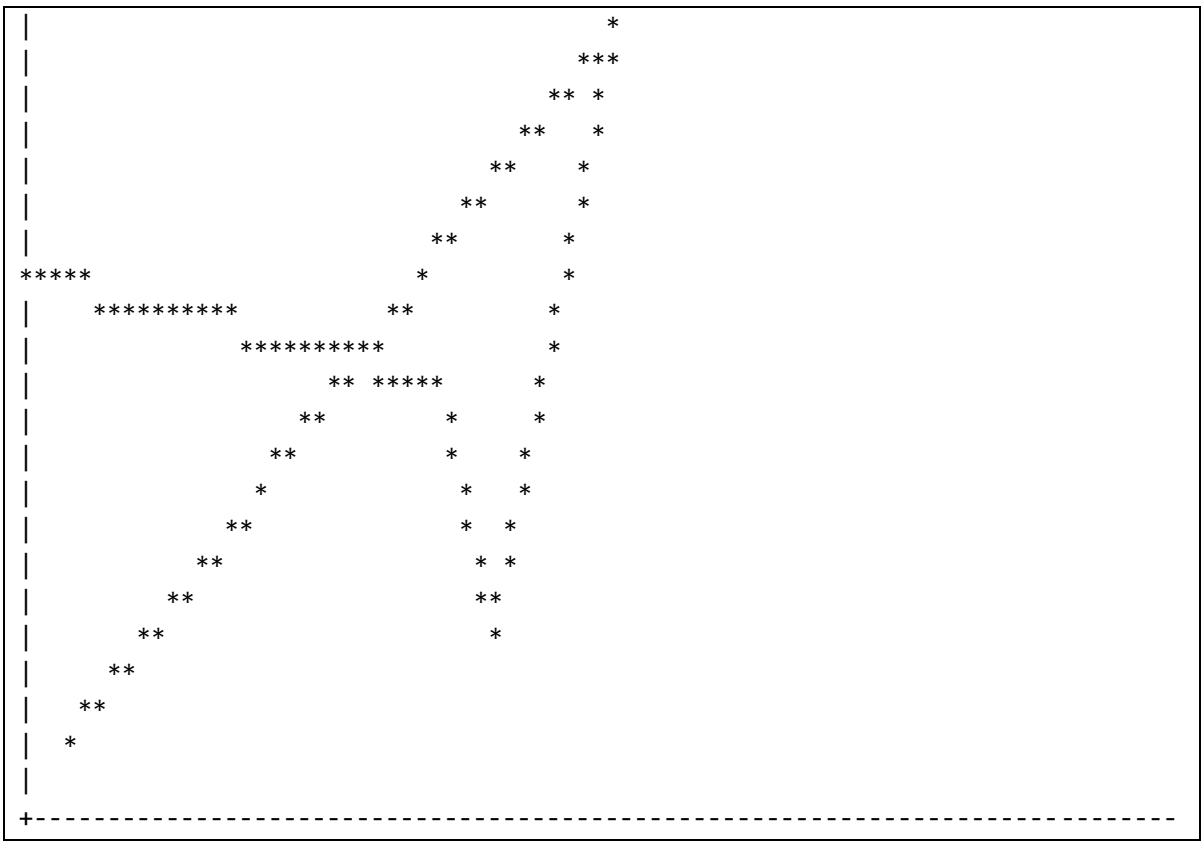
```
 5
 3  2
40 22
32  5
28 12
 0 15
```

- You can assume that *<n> must be at least 2*, and all points must have *x-coordinates in the range* $[0,78]$ and *y-coordinates in the range* $[0,22]$. With the standard 2-D coordinate system, i.e., x-axis being the horizontal axis and y-axis being the vertical axis, the origin $(0,0)$ is in the lower left corner.
- The programs then use the DDA algorithm to plot $n-1$ lines $p_1 \leftrightarrow p_2$, $p_2 \leftrightarrow p_3$, $p_3 \leftrightarrow p_4$, …, $p_{n-1} \leftrightarrow p_n$. Your *Fortran program should plot the lines to the standard output* (console), while your *COBOL program should plot the lines to an output file* named `output.txt`.
- In the output, you should print out 23 lines, each line of which contains 79 characters. This corresponds to the 2-D plane spanning the space between $(0,0)$ and $(78,22)$. The origin $(0,0)$ is in the lower left corner of your output and is printed as a '+'. The positive x-axis is printed as '-'.

Copyright © 2016 CSE, CUHK

The positive y-axis is printed as '|'. A point which is part of any of the $n-1$ lines is printed as '*'. A point which is neither a part of any of the $n-1$ lines nor the axes is printed as a space '␣'. Note that if a '*' has to be plotted at some point on the axes, then '+', '-', or '|' would not be printed at that point. (That is, '*' has higher priority than '+', '-', and '|'.)

- The following shows an example output using the five data points $(3,2)$, $(40,22)$, $(32,5)$, $(28,12)$, and $(0,15)$ in the example input file above:



Other requirements

- *The only control flow construct that you can use in your programs is `IF-GOTO`. All other selection and repetition constructs are **not** allowed* (such as `DO`, `WHILE`, `UNTIL`, `ELSE`, `ELSEIF` in Fortran and similar constructs in COBOL). The purpose of this restriction is to let you understand the difficulties that programmers encountered in the past.
- In case of any file opening error, you have to print an error message to standard output (console) and terminate the program.
- You cannot write your programs in other languages and then initiate system calls or external library calls in Fortran and COBOL.
- Your program should *include suitable comments as documentation*. Failure to do so will suffer from mark deduction.
- You should *decompose your programs into subroutines*. Do not clog the main programs.

Report

Your written report (`report.pdf`) should answer the following questions within one A4 page.

Copyright © 2016 CSE, CUHK

1. Compare the conveniences and difficulties in implementing the program in Fortran and COBOL. You can divide your comparison into specific tasks such as "file I/O", "DDA implementation", etc.
2. Compare Fortran and COBOL with a modern programming language (e.g., C/C++/Java/…) in different aspects (e.g., data types, parameter passing, paradigm, paradigm, etc.). You are free to pick your favorite modern programming language.

Submission and Marking

- Your program file names should be `dda.for` and `dda.cob`. Name your report as `report.pdf`. Submit all three files in Blackboard (https://elearn.cuhk.edu.hk/). Besides, your *report* has to be submitted to VeriGuide (http://www.veriguide.org/) as well.
- You can submit your assignment multiple times. Only the latest submission counts.
- Your programs should be *free of compilation errors*.
- *Plagiarism is strictly monitored and heavily punished if proven.* Lending your work to others is subjected to the same penalty as the copier. You have to insert the following statement as comments in your Fortran and COBOL programs.

```
CSCI3180 Principles of Programming Languages
--- Declaration ---
I declare that the assignment here submitted is original except for source material explicitly acknowledged. I also acknowledge that I am aware of University policy and regulations on honesty in academic work, and of the disciplinary guidelines and procedures applicable to breaches of such policy and regulations, as contained in the website http://www.cuhk.edu.hk/policy/academichonesty/
Assignment 1
Name:
Student ID:
Email Addr:
```

Copyright © 2016 CSE, CUHK