Homework Assignment #6: Programming Exercise #1

Due Date & Time

2:10PM Monday, November 19, 2018. Late submission will be penalized by 20% for each working day overdue.

Problem Description

Implement an algorithm that computes the skyline of a list of buildings. The representations of a building and a skyline are as discussed in class except that parentheses are omitted and commas are replaced by spaces. The height and width of a building are always positive integers.

You may assume that there are at most 1000 buildings in each input. The first line of an input is an integer n, indicating the number of buildings. It is followed by n triples of integers, each triple in a separate line, indicating the coordinates of a building.

Sample input:

```
8
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
```

Correct output for the sample input:

```
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29
```

Notes

This assignment constitutes 4% of your grade. You may discuss the problem with others, but copying code is strictly forbidden.

Language Requirements

All the IM students are required to implement their algorithms in standard C/C++. For other students, implement your algorithms in one of the following languages.

- Standard C/C++
- Java 8
- Python 3

Do not use any library that is not included in the standard installation of compilers or interpreters. Never use a compiler-specific feature. Make sure that your code can be compiled by a standard compiler (or interpreted by a standard interpreter) without any specific argument, for example, "`gcc b067050xx.c`", "`g++ b067050xx.cpp`", "`g++ b067050xx.cc`", "`javac b067050xx.java`", or "`python3 b067050xx.py`".

Interface Requirements

Your application must accept a single argument, which is an input file. No other arguments will be provided. For example, "`./b067050xx case1`", "`java b067050xx case1`", or "`python3 b067050xx.py case1`". The application must output the skyline directly in a single line. Do not output any other verbose messages.

An example of an invalid output:

```
Skyline: 1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29
```

Another example of an invalid output:

```
Skyline:
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29
```

File Requirements

You are required to provide the following three files.

- A single source file, named by your student ID (for example, b067050xx.c or b067050xx.py), containing the implementation of the algorithm.
- A README file describing how to compile your source file.
- A MS Word file or a PDF file describing algorithmic techniques applied in the implementation.

Submission Guidelines

- Pack the three required files, in a .zip file, named with the pattern "`b067050xx-hw6.zip`".
- Send the .zip file to r06725007@ntu.edu.tw.

Grading

Your work will be graded according to its correctness, performance, and presentation. Before submission, you should have tested your program on several input cases. You should organize and document your program in such a way that other programmers, for example your classmates, can understand it. In the documentation of your program, you should describe how you have applied the algorithmic techniques, in particular design by induction, learned in class. For example, if you choose to use the merging of two skylines as a building block, try to elaborate on how induction has helped in the design of the merging procedure.

Below is a more specific grading policy:

Criteria	Score
incomplete, doesn't compile, or runtime errors	≤ 20
complete, compiles, but with incorrect results	≤ 60
correct but with an $> O(n \log n)$-time algorithm	≤ 80
correct and with an $O(n \log n)$-time algorithm	≤ 100