CS3230 Programming Assignment 2

Background

Space is very limited in NUS and assigning lectures to lecture halls efficiently is a pertinent problem. Due to the busy schedules of the lecturers, most of them can only give the lectures at a fixed time slot in the week.

Since there is a limited number of lecture halls, the lectures should be assigned such that the halls are well utilised. Hence, given all the lecture timings, you want to know what is the minimum number of lecture halls needed so that all the lectures can take place at their time slot.

However, there may not be enough lecture halls in the school to fit all the lectures, so you also want to know for a given fixed number of lecture halls, what is the minimum number of lectures that need to be cancelled so that all the lectures can take place at their time slot.

Details

There are a total of N lectures that need to be conducted labelled 0 to $N-1$ and each lecture must be given at a fixed time slot but there are only L lecture halls. For convenience, the time slot for lecture i starts at $start_i$ time units and ends at end_i time units where $start_i < end_i$. Another lecture can start right after a lecture ends in the same lecture hall which means if one lecture ends at 5 time units and another lecture starts at 5 time units, both can use the same lecture hall.

Given these limitations, you need to calculate 2 values.

1. If you don't want to cancel any lectures, what is the minimum number of lecture halls that need to be booked such that all the lectures can be scheduled in a way where no 2 lectures are in the same lecture hall at the same time.

2. If there are only L lecture halls, what is the minimum number of lectures that need to be cancelled so that all the lectures can take place at their time slot using only L lecture halls.

Implementation

You need to implement 2 functions to calculate the 2 values described above. The functions will take in 4 parameters:

- N - The number of lectures
- L - The number of lecture halls
- start - A list of N integers integers where start[i] is $start_i$ as described above
- end - A list of N integers where end[i] is end_i as described above

You are allowed to submit in either Java or C++11. Templates for both languages have been provided in the IVLE Workbin in the same folder as this task statement. The templates have implemented the I/O routines for you so you are strongly encouraged to use them. In addition, you are allowed to add other functions into your program to modularise your code but you must submit only **ONE** source file.

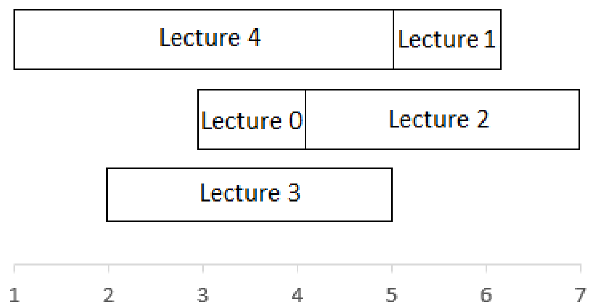
For testing purposes, the program will read in the parameters listed above from standard input in the exact order listed above for each test case with each parameter on a separate line. There can be multiple test cases within one run and the first integer gives the number of test cases. The program will output one line for each test case, which contains one integer that is the answer for that test case.

For example, if there are 5 lectures and 2 lecture halls where start = [3, 5, 4, 2, 1] and end = [4, 6, 7, 5, 5] then the input it expects will look like this:

```
1
5
2
3 5 4 2 1
4 6 7 5 5
```

Example

For the example given above, if no lectures can be cancelled then the minimum number of lecture halls required is 3. The first lecture hall would host lectures 4 and 1 in that order while the second lecture hall would host lectures 0 and 2 in that order and the third lecture hall would host lecture 3. The scheduling will look like the following diagram:



However, since area is only 2 lecture halls, the minimum number of lectures that need to be cancelled is 1. Specifically, cancelling lecture 3 will allow all the remaining lectures to fit into 2 lecture halls.

Submission

The assignment is split into multiple parts with a separate score for each part. You are encouraged to work on all the parts in order since it will help you design your algorithm step by step.

You will submit your solutions on `http://algorithmics.comp.nus.edu.sg/~mooshak/` using the same username and password as PA1. You are NOT allowed to share accounts and you must submit your own code. Your codes will be checked for plagiarism and any instances of plagiarism found will be subject to disciplinary action.

To submit your solutions, select the contest "CS3230 PA2" and login. You will see parts A, B, C and D which represent the parts described below. Submit your solution and it will be run against a set of 10 test cases. Your code must solve all test cases correctly and within 1 second for parts A, B and C and within **2 seconds for part D** in order to get the credit for that part. You will see the result of your submission immediately after submitting and you can submit as many times as you want before the deadline without any penalty for wrong answers.

In addition, sample test cases for each part has been provided for you on the IVLE Workbin to do your own testing. However, do note that these test cases are different from the ones on the online judge so solving those test cases correctly does not ensure that you will solve those on the judge correctly.

Scoring

Part A (2%)

In the first part, you are required to simply design any algorithm that solves the problem to show that you understand the problem. The test cases for this part will satisfy the following restrictions:

- $1 \leq L \leq N \leq 5$
- $1 \leq start_i < end_i \leq 10$ for all i

Part B (2%)

In the second part, you are required to design a Greedy algorithm to calculate the first value and it should run in time complexity $O(N \times max(end_i))$. In this part, $L = N$ for all test cases so there will always be enough lecture halls and thus no lectures have to be cancelled. The test cases for this part will satisfy the following restrictions:

- $1 \leq L = N \leq 500$
- $1 \leq start_i < end_i \leq 1000$ for all i

Part C (2%)

In the third part, you are required to design a Greedy algorithm to calculate the second value and it should run in time complexity $O(NL)$. You should add this algorithm on top of your solution for Part B to get the full credit. The test cases for this part will satisfy the following restrictions:

- $1 \leq L \leq N \leq 500$
- $1 \leq start_i < end_i \leq 1000$ for all i

Part D (2%)

In the fourth part, you are required to optimise your algorithms from Parts B and C so that it runs with time complexity $O(N \log N)$. (Hint: You may need to make use of other *non-linear* data structures). The test cases for this part will satisfy the following restrictions:

- $1 \leq L \leq N \leq 30000$
- $1 \leq start_i < end_i \leq 1000000000$ for all i

Deadline

The deadline for submission for this assignment is 12th Nov 2018 2359h so please start working on it **early**. The portal will automatically close at this time and no further submissions will be accepted.