CLASSICAL AND QUANTUM COMPUTING
EXERCISE XVIII

The following table describes jobs (or tasks) according to their starting time and time of completion.

| Job | Start | End |
|---|---|---|
| 1 | 6 | 10 |
| 2 | 1 | 5 |
| 3 | 1 | 6 |
| 4 | 9 | 12 |
| 5 | 5 | 7 |
| 6 | 6 | 14 |
| 7 | 3 | 7 |
| 8 | 10 | 14 |
| 9 | 13 | 16 |

Obviously there may be some conflicts, for example job 1 and job 6 would need to be done simultaneously. The scheduling problem requires that we find the subset of maximum size of jobs which do not conflict. For example doing jobs 3, 1, 9 in that order is a non-optimal solution.

A greedy algorithm is an algorithm that optimizes the choice at each iteration without regard to previous choices. In the case of our scheduling problem a greedy algorithm chooses the job which finishes in the least amount of time in order to fit as many jobs as possible. Obviously this cannot in general lead to an optimal solution.

Write a C++ or Java program which implements a greedy algorithm to solve the scheduling problem. In order for the algorithm to be efficient it is necessary to sort the jobs according to their times first.