Problem for Online on Greedy Algorithm for Section B2
Interval Scheduling Problem

The Problem:

There are some job/task requests to run on a shared machine. Each job has a specific start and end time, called an interval. The machine can run at most one job at a time and hence if two job requests' intervals overlap (in time), you cannot accept both the requests. We say two job requests do not overlap if both the start time and end time of one job is greater than or equal to the end time of the other job. Now given a set of job requests, your task is to maximize the number of requests accepted.

The Greedy Solution:

This problem is solvable by greedy strategy. The idea is to accept the job that finishes first, i.e., the job with the smallest finishing time. This will allow the machine to become free at the earliest still satisfying a request. Hence, sort all the jobs in the order of non decreasing finishing time and start accepting non overlapping jobs from that order.

The Input:

The first line of the input contains an integer '*t*' representing the number of test cases. Then '*t*' test cases follow. Each test case has the following form:

- Line 1: A single integer, *N* representing the number of job requests.
- Lines 2: Space separated 2*N* integers representing <start time, end time> pairs for the *N* jobs;

The Output

For each test case, there will be three lines of output. The lines will be as per following format:

- Line 1: Maximum number of jobs that can be accepted.
- Lines 2: Space separated indices of the accepted jobs. Assume the starting index *1*. Print the indices in the order they are picked.
- Lines 3: Space separated start time and end/finish time of the accepted jobs in the order they are picked.

Sample Input:	**Sample Output**
2	2
4	3 4
1 4 2 7 1 3 6 9	1 3 6 9
7	4
5 8 2 6 1 4 8 9 7 11 3 6 4 5	3 7 1 4
	1 4 4 5 5 8 8 9

Problem for Offline on Greedy Algorithm for Section B2

Variation of interval scheduling problem

1. Consider the following variation on the Interval Scheduling Problem. The machine (that runs jobs) can operate 24 hours a day, every day. Hence if a job is accepted to run on the machine, it must run continuously, every day, for the period between its start and end times. Note that certain jobs can begin before midnight and end after midnight; this makes for a type of situation different from what we saw in the Interval Scheduling Problem. Now given a set of job requests, your task is to maximize the number of requests accepted. Give a polynomial time algorithm to solve the problem. You may assume for simplicity that no two jobs have the same start or end times.
For example, consider the following four jobs, specified by (start-time, end-time) pairs. (6 P.M., 6 A.M.), (9 P.M., 4 A.M.), (3 A.M., 2 P.M.), (1 P.M., 7 P.M.). The optimal solution would be to pick the two jobs (9 P.M., 4 A.M.) and (1 P.M., 7 P.M.), which can be scheduled without overlapping.

2. You are required to use HEAP data structure in case your algorithm need to extract jobs based on the max/min value of some parameter. The heap should be constructed in O(N) time.

3. Submit a **hand written** report with the following components:

   a) Problem description
   b) Data structure used
   c) Pseudo code of the algorithm
   d) Proof of correctness
   e) Analysis of running time