Lab 5 (29 Sep 2020)

Problem 1: Write a O(nlogn) program to find if an array A (of size n) has a *majority element* in A. A majority element is an element that occurs more than n/2 times in A. You **can not** order (that is, compare or sort) elements of A, you can only check if two elements are equal.

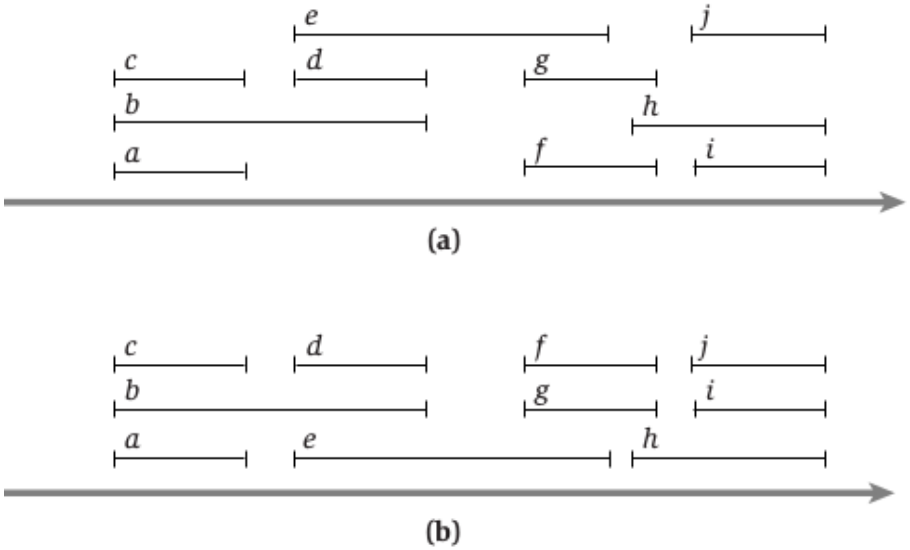
Problem 2: Implement the greedy algorithm for interval scheduling. Given a list of intervals you need to print a subset of non-overlapping intervals whose size is maximal. You can take the start/finish times of the intervals to be positive integers. A sample i/o is given below:

Sample Input**:**
Enter the number of intervals: 8
1 3
2 8
2 5
3 7
4 8
4 6
6 12
7 10

Output**:**
[1, 3] [4, 6] [7, 10]

Problem 3: Implement the greedy interval *partitioning* algorithm. Given a list of jobs (intervals) you need to find the minimum number of resources needed to schedule **all** the jobs such that each resource schedules only non-overlapping jobs. (The example/picture below is taken from KT.)

Input**:**
Enter the number of intervals: 10
1 3
1 6
1 3
4 6
4 10
8 12
8 12
11 15
13 15
13 15



Output**:**
Minimum number of resources: 3
Resource 1 jobs: [1,3] [4,10] [11,15]
Resource 2 jobs: [1,6] [8,12] [13,15]
Resource 3 jobs: [1,3] [4,6] [8,12] [13,15]