

1. (10 points) You are having a party and your guests are arriving and leaving at different times, (you will be at the party the entire time.) You want to find two other people to play a 3 person game with. You would like to know how long you can play the game for so you are looking for two people who will be together at the party for the longest time.

Given the start and end times of each of  $n$  guests:  $[(s_1, f_1), \dots, (s_n, f_n)]$  design an  $O(n \log n)$  algorithm that returns the maximum time that a pair of people will be together at the party.

(implementation level or high level description and time analysis. No correctness proof necessary.)

2. (10 points) A list of integers  $(a_1, \dots, a_n)$  is called unimodal if it increases up to a point and then decreases, i.e. there exists an index  $i$  such that  $a_1 < a_2 < \dots < a_i$  and  $a_i > a_{i+1} > \dots, a_n$ . The index  $i$  is called the “peak”. Design a  $O(\log n)$  algorithm that returns the “peak” of a unimodal list. (note that an increasing sequence is a unimodal list with peak at  $n$  and a decreasing sequence is a unimodal list with peak at 1.)

(high-level or implementation level, runtime analysis and proof of correctness.)

3. (10 points)

- (a) Let  $T(n)$  be the runtime of the Median of Medians algorithm. Then in class we saw that for some constant  $c$ ,  $T(n) = T(n/5) + T(7n/10) + cn$ . Show that  $T(n) = O(n)$ . Show that there exists a constant  $C$  such that  $T(n) \leq Cn$  for all  $n$  and use induction to prove it.
- (b) Suppose an algorithm has runtime given by  $R(n)$  such that  $R(n) = 2R(\sqrt{n}) + O(1)$ . Show that the runtime of  $R(n)$  is  $O(\log n)$ .

4. (10 points) Given an undirected graph  $G$  with  $n$  vertices and  $m$  edges, a Hamiltonian path starting at  $s$  and ending at  $t$  is a path in the graph that goes through each vertex exactly once.

Suppose the maximum degree of the graph is 3.

- (a) (9 points)

Design a backtracking algorithm that determines if  $G$  has a hamiltonian path that starts from  $s$  and ends at  $t$  that runs in  $O(2^n)$  time.

(Algorithm pseudocode or implementation level, proof of correctness, runtime analysis)

- (b) (1 point) Explain that your algorithm runs in  $O(2^{(3/4)n})$  time or explain how to modify your algorithm to achieve a runtime of  $O(2^{(3/4)n})$ .