Midterm 1

- **The exam has four questions each worth 25 points, and will last 120 minutes.**
- Answer all questions. Read them carefully first. Not all parts of a problem are weighted equally.
- Begin each problem on a new page
- Be precise and concise.
- The problems may **not** necessarily follow the order of increasing difficulty.
- Good luck!

1 Is There A Cycle?

Design a linear time algorithm that given a directed graph G, outputs a cycle in the graph if there is one, or else a source vertex and a sink vertex. Just an algorithm and a clear justification of why it always gives a valid output are needed.

2 FFT simplified

(a) Write the 2-by-2 Fourier transform matrix. What root of unity did you use? Write it in the form $a + bi$.

(b) Denote by H_1 your 2-by-2 solution to the previous part. We recursively define the 2^n-by-2^n matrix H_n as:

$$H_n = \begin{bmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{bmatrix}$$

Explicitly write down the 4-by-4 matrix H_2.

(c) Let $N = 2^n$. Given an N-dimensional vector v, give an $O(N \log N)$-time algorithm to compute $H_n v$. Just the algorithm and runtime analysis is needed.

3 Booking Flights

We want to book a cheap flight route to travel from city s to city t. There are n cities with airports including s and t. Airlines offer m flights, where the ith flight goes non-stop from city u_i to v_i and costs c_i dollars (all c_i are positive integers). We wish to find the cheapest route from s to t, but if there are multiple cheapest routes, we wish to find among them one with the fewest flights (to minimize the number of airports we have to transit through). Give an algorithm that outputs such a flight itinerary, and a clear justification of correctness.

Note: If you modify Dijkstra's itself, the correctness of Dijkstra's probably does not imply the correctness of your algorithm. You will have to give a modified version of the proof of correctness as well for full credit. If, on the other hand, you use Dijkstra's as a black box in your algorithm, using Dijkstra's correctness as a black box in your proof will probably suffice.

4 Netflix Similarity

There are n movies on Netflix (identified by the numbers $1, \ldots, n$) and Alice and Bob have watched all of them! Alice lists all n movies according to her ranking starting with her favorite $(a_1, \ldots, a_n)$ and similarly, Bob lists all movies according to his ranking $(b_1, \ldots, b_n)$. One measure of difference between Alice and Bob's tastes is the number of inversions between their orderings, i.e. the number of pairs of movies u, v such that u is rated higher than v by Alice but lower by Bob. Design an $O(n \log n)$ algorithm to compute the number of inversions. Justify the correctness of your algorithm.

Example: if Alice's ordering is $(4, 3, 1, 2)$, with 4 her favorite movie and 2 her least favorite, and Bob's ordering is $(3, 4, 2, 1)$, then the number of inversions is 2, corresponding to the pairs $1, 2$ and $3, 4$.

Note: if you are unable to solve this problem, for half the points you may solve the problem where you are given just one sequence of n numbers $(x_1, \cdots, x_n)$, and you wish to compute the number of inversions in that list, i.e. pairs (i, j) such that $i < j$ but $x_i > x_j$ (this was covered in lecture!)

If you are solving the problem for full points, you may call the algorithm from lecture as a black box if needed.