1) We would like to find an approximate median of \( n \) distinct integers in sublinear time. To do so, we down-sample \( m \ll n \) numbers, and output the median \( c \) of these \( m \) numbers. Let the sorted list of the given \( n \) numbers be \( \{x_1, x_2, \ldots, x_n\} \) and so the true median is \( x_{n/2} \). The approximate median is said to be a \( \pm k \)-approximation if \( c \in \{x_{n/2-k}, \ldots, x_{n/2+k}\} \). Suppose we want to design an algorithm that succeeds to find a \( \pm cn \)-approximation with probability at least \( 1 - \delta \). How large should \( m \) and what would be the running time of the algorithm? Prove your claim. You can assume one can find the median of \( m \) numbers in time \( O(m) \).

\textbf{Hint:} Try to get a bound that is only logarithmically dependent on \( 1/\delta \).

2) Recall a set of vectors \( v_1, \ldots, v_n \in \mathbb{R}^d \) are linearly independent if for any set of coefficients \( c_1, \ldots, c_n \in \mathbb{R} \), \( c_1 v_1 + \cdots + c_n v_n \neq 0 \).

The \( \text{rank}(A) \) of a matrix \( A \) is the maximum number of linearly independent columns of \( A \); it is also equal to the maximum number of linearly independent rows of \( A \). We say an \( n \times n \) matrix \( A \) is \textit{nonsingular} if \( \text{rank}(A) = n \). It can be shown that for any matrix \( A \), \( \text{det}(A) \neq 0 \) if and only if \( A \) is nonsingular. Let \( G = (X, Y, E) \) be a given bipartite graph with \( |X| = |Y| = n \). Using the above terminology, we can rewrite the algorithm that tests whether \( G \) has a perfect matching as follows: For each edge \( x_i, y_j \) of \( G \), choose \( A_{i,j} \) uniformly and independently from the set \( \{0, 1, \ldots, n^k\} \), and let the rest of entries of \( A \) be 0. Return yes if \( \text{rank}(A) = n \) and no otherwise.

a) Let \( A \) be the following matrix: For each nonadjacent pair \( x_i, y_j \), let \( A_{i,j} = 0 \); choose the rest of the entries of \( A \) arbitrarily. Use properties of the rank and determinant to show that if \( \text{rank}(A) = k \), then \( G \) has a matching of size at least \( k \).

b) Design a randomized algorithm to compute the size of the maximum matching of \( G \). Can you upper bound probability of failure of your algorithm? In your algorithm assume that you can compute the rank of a matrix in polynomial time.

\textbf{Hint:} Use Schwartz-Zippel lemma to argue that with high probability \( \text{rank}(A) \) is at least the size of the maximum matching of \( G \) then use part (a) to finish the proof.

3) Given a graph \( G \) with \( n \) vertices, let \( A \) be the adjacency matrix of \( G \) and let \( \lambda_1 \) be the largest eigenvalue of \( A \).

a) Use definition of eigenvalues \( (Av = \lambda v) \) to show that \( \lambda_1 \leq \max_v d(v) \), i.e., \( \lambda_1 \) is at most the largest degree of vertices of \( G \).

b) Use Rayleigh quotient to show that \( \lambda_1 \geq \frac{\sum_v d(v)}{n} \).

3-1