1. You are given a flow network with n vertices and $m = \Theta(n)$ edges, and where the maximum s-t flow is $F = 2^{\sqrt{n}}$. You want to find a maximum flow in this network. Select the fastest algorithm among the following to do this task:
   - Edmonds-Karp I is the fastest with time $O(m^2\sqrt{n})$. Remember that it has running time $O(m^2 \log F)$.

2. You are given a flow network with n vertices and $m = \Omega(n)$ edges, and where the maximum s-t flow is $F = O(\sqrt{n})$. You want to find a maximum flow in this network. Select the fastest algorithm among the following to do this task:
   - Ford-Fulkerson is the fastest with time $O(n\sqrt{n})$. Remember that it has running time $O(mF)$.

3. Which of the following is/are polynomial-time algorithms to find a maximum s-t flow in a directed network.
   - EK1, EK2, and Dinic's because FF is dependent on max flow F, which can be represented using log(F) bits in the input. So FF is exponential in terms of the input. All other algorithmns are polynomial in terms of n, m, log(F).

4. You want to find a maximum matching in a graph G. Which of the following statements is true.

   (a) You can use a max-s-t-flow algorithm to find a maximum matching if G is bipartite.

   This is true because you can connect a source to one group and sink to the other to find a maximum matching using flow

   (b) You can use a max-s-t-flow algorithm to find a maximum matching even if G is non-bipartite.

   This is false because there is no way to connect a source to a group and sink to a group so that there are no edges within the groups, and all of the edges leave the group. This makes it impossible to find a set of edges with no overlapping vertices using only max flow.

   (c) Edmonds' blossom algorithm finds a maximum matching if G is bipartite.

   The blossom algorithm does not depend on whether or not a graph is bipartite, because it searches for blossoms, contracts the graph, and repeats. Nowhere in this process does the bipartiteness of the graph come into play

   (d) Edmonds' blossom algorithm finds a maximum matching even if G is non-bipartite.

(e) Finding maximum matchings even in non-bipartite graphs can be done in polynomial time.

Edmond's blossom algo is polynomial.

5. In the following 2-player zero-sum game, give the maxmin-optimal strategy of the row-player (shooter), and her payoff.

payoff matrix M	goalie L	goalie R
shooter L	$(4, -4)$	$(3, -3)$
shooter R	$(1, -1)$	$(2, -2)$

- **Shooter** plays L with probability 1 and R with probability 0.

  The maxmin-optimal payoff to the shooter is 3.

  Answer: In both cases, the payoff for the shooter is better in row L: $4 > 1$, $3 > 2$. (Action L is the "dominant strategy".) The goalie minimizes his payoff for that row, giving $(3, -3)$ as the equilibrium.

  A lot of you just mechanically set the expected payoffs for both actions equal to each other, and got the incorrect answer. Note that in this case you want to maximize $\min\{4p + 1(1-p), 3p + 2(1-p)\}$*, subject to* $p \in [0, 1]$*.* ***In this case the maximum is not achieved at the intersection of the two lines*** $4q + 3(1-q)$ ***and*** $1q + 2(1-q)$***.***

6. In the following 2-player zero-sum game, give the maxmin-optimal strategy of the column-player (goalie), and his payoff.

payoff matrix M	goalie L	goalie R
shooter L	$(3, -3)$	$(2, -2)$
shooter R	$(1, -1)$	$(5, -5)$

- **Answer:** Goalie plays L with probability .6 and R with probability .4.

  The maxmin-optimal payoff to the *goalie* is -2.6.

  We want to minimize $\max\{3q + 2(1-q), 1q + 5(1-q)\}$ subject to $q \in [0, 1]$. The first equation comes from your expected payoff when the shooter chooses L, the second from your expected payoff when the shooter chooses R. In this case the optimal point is indeed obtained by setting both the constraints to be equal.