INSTRUCTIONS:

1. All the solutions are given with the problems.
2. This assignment will not be graded. It is exercise only.

Problem 1 (0 points).

You are given an undirected graph $G = (V, E)$, and you need to color each vertex with one of the given four colors. We say an edge $(u, v) \in E$ is *satisfied* if u and v are assigned different colors. Given $G = (V, E)$, the problem is to color all vertices so that the number of satisfied edges is maximized.

1. Design a randomized $4/3$-approximation algorithm for this problem, that is, the *expected* number of satisfied edges returned by your algorithm should be at least $3/4$ fraction of the number of the satisfied edges in the optimal solution. Prove your algorithm can achieve such randomized approximation ratio. Your algorithm should run in polynomial-time.
2. Design a (deterministic) $4/3$-approximation algorithm for this problem, that is, the number of satisfied edges returned by your algorithm should be at least $3/4$ fraction of the number of the satisfied edges in the optimal solution. Prove your algorithm can achieve such approximation ratio. Your algorithm should run in polynomial-time.

Solution.

1. The algorithm independently assigns one of the 4 colors with probability of $1/4$ for each vertex. We now show its expected performance. Let Z be the random variable indicates the total number of satisfied edges. Let Z_e be the binary random variable indicates whether edge $e \in E$ is satisfied. We have that $Z = \sum_{e \in E} Z_e$ and therefore $\mathbb{E}(Z) = \sum_{e \in E} \mathbb{E}(Z_e)$. Further, edge $e = (u, v)$ is satisfied if and only if u and v are colored differently, and with probability of $4 \cdot 1/4 \cdot 1/4$, they are colored the same. Hence, $\Pr(Z_e = 1) = 1 - \Pr(Z_e = 0) = 1 - 1/4 = 3/4$. Combined, $\mathbb{E}(Z) = \sum_{e \in E} \mathbb{E}(Z_e) = 3 \cdot |E|/4$. As the optimal solution satisfies at most $|E|$ edges, this algorithm is a randomized $4/3$-approximation algorithm.
2. We can use the conditional expectation technique to derandomize above randomized algorithm to get a deterministic approximation algorithm with the same approximation ratio. Let $\{c_1, c_2, c_3, c_4\}$ be the four given colors. Number all vertices as $V = \{v_1, v_2, \cdots, v_n\}$. The derandomization algorithm is given below. In the main for-loop, we determine the color for vertex v_i by comparing the four conditional expectations.

```
Algorithm DERAND
    for i = 1 to n
        compute A_1 = E(Z | v_1 = v_1^*, v_2 = v_2^*, ..., v_{i-1} = v_{i-1}^*, v_i = c_1)
        compute A_2 = E(Z | v_1 = v_1^*, v_2 = v_2^*, ..., v_{i-1} = v_{i-1}^*, v_i = c_2)
        compute A_3 = E(Z | v_1 = v_1^*, v_2 = v_2^*, ..., v_{i-1} = v_{i-1}^*, v_i = c_3)
        compute A_4 = E(Z | v_1 = v_1^*, v_2 = v_2^*, ..., v_{i-1} = v_{i-1}^*, v_i = c_4)
        among the above 4 values find the largest one A_k and set v_i^* = c_k
    end for
    return {v_1^*, v_2^*, ..., v_n^*}
end DERAND
```

To compute the conditional expectation, for example $A_1 = E(X \mid v_1 = v_1^*, v_2 = v_2^*, \cdots, v_{i-1} = v_{i-1}^*, v_i = c_1)$, we use $A_1 = \sum_{e \in E} E(Z_e \mid v_1 = v_1^*, v_2 = v_2^*, \cdots, v_{i-1} = v_{i-1}^*, v_i = c_1)$. Let $e = (u, v)$. If both u and v are colored, i.e., $u, v \in \{v_1, v_2, \cdots, v_i\}$, then we have $E(Z_e \mid v_1 = v_1^*, v_2 = v_2^*, \cdots, v_{i-1} = v_{i-1}^*, v_i = c_1) = 1$ if they are colored differently, and equals to 0 if not. If either u or v is not colored yet, then we have that $E(Z_e \mid v_1 = v_1^*, v_2 = v_2^*, \cdots, v_{i-1} = v_{i-1}^*, v_i = c_1) = 3/4$.

Using the same argument (in the MAX-3SAT problem), we can prove that this algorithm keeps the invariant that after i-th iteration, $E(Z \mid v_1 = v_1^*, v_2 = v_2^*, \cdots, v_{i-1} = v_{i-1}^*, v_i = v_i^*) \geq E(Z) = 3 \cdot |E|/4$. Hence after n iterations (i.e., when the algorithm terminates), all vertices are colored (with $v_1^*, v_2^*, \cdots, v_n^*$), and the number of satisfied edges is at least $3 \cdot |E|/4$.

Problem 2 (0 points).

Exercise 7 (page 787, *Algorithm Design* [KT]).

Solution. (a) We have proved that $E(Z) = k/2$. An tight example (with 1 variable and 2 clauses) is as follows. $C_1 : x_1$ and $C_2 : \overline{x_1}$.

(b) Let A be the set of single-literal clauses. For each clause in A we independently at random satisfy it with probability $p \geq 0.5$ (we will determine the actual value of p later on). For examples, if such a clause is x_2 then we set $x_2 = 1$ with probability p; if such a clause is $\overline{x_4}$ then we set $x_4 = 0$ with probability p. For all other variables that have not been assigned, we independently at random set them to 1 with probability of exactly $1/2$.

We now analyze this algorithm. For each clause C in A, clearly $\Pr(C = 1) = p$. For each clause C that is not in A, we know that C contains at least 2 literals. The worst case is that C contains exactly 2 literals for which their opporsite literals have been assigned in processing A. (For example, C is $\overline{x_2} \vee x_4$, and x_2 and $\overline{x_4}$ are two single-literal clauses in A.) In this case, $\Pr(C = 1) = 1 - \Pr(C = 0) = 1 - p^2$. This is the worse case. In other words, for any clause C that not in A we must have that $\Pr(C = 1) \geq 1 - p^2$.

To get the actual value of p, we simply let $p = 1 - p^2$, i.e., let each clause have the same (worse-case) probability of being satisfied. This gives $p = 0.62$. Hence, we have $E(Z) = 0.62k$.

(c) For each pair of single-literal clauses x and $\bar{x}$, we simply keep one of them and discard the other. Let k_1 be the number of removed clauses in this way. We then run the algorithm given in (b) on the remaining $k - k_1$ clauses.

We now analyze this algorithm. According to (b), the expected number of satisfied clauses equals to $0.62 \cdot (k - k_1)$. On the other side, consider the optimal solution. Note that for any pair of conflicting clauses, at most one of them can be satisfied in any assignment. Therefore, the maximized number of clauses that can be satisfied is $(k - k_1)$. Combined, in expectation, this randomized algorithm gives at least 0.62 fraction of the satisfied clauses in optimal solution.