The “Classroom Problems” will be discussed and solved ex tempore at the tutorials in the same week as this problem set is handed out. These are intended to help you in getting started with the material, and will not yield any course credit. The “Homework Problems” you should solve at home, and attend one of the tutorial sessions in the following week, prepared to present your solutions at class if requested. You earn one tutorial credit point (= 0.33 course points, for a maximum total of 6) for each of these problems that you have solved, and marked as solved at the tutorial session. The “Demonstration Problems” illustrate further interesting aspects of the material. These will be discussed at the tutorials in the same week as the Homework Problems (time permitting) and printed solutions will be provided. You are encouraged to work on these, but no course credit is given for them. The “Challenge Problems” are extra problems related to the material, which may be exceptionally difficult and/or require tools beyond the course curriculum.

**Classroom problems:**

1. Show that for infinitely many $n$, there are graphs with $n$ vertices such that the maximal matching-based approximation algorithm for Vertex Cover presented at Lecture 11 may produce a cover that is twice the size of the optimum cover. Thus the approximation ratio 2 derived for this algorithm at Lecture 11 is tight.

2. Let $0 < \varepsilon_2 < \varepsilon_1 < 1$. Consider a Monte Carlo algorithm that gives the correct solution to a problem with probability at least $1 - \varepsilon_1$, regardless of the input. How many independent executions of this algorithm suffice to raise the probability of obtaining at least one correct solution to at least $1 - \varepsilon_2$, regardless of the input?

**Homework problems:**

3. [Dasgupta et al., Ex. 9.6] In the Minimum Steiner Tree problem, the input consists of: a complete graph $G = (V, E)$ with distances $d_{uv}$ between all pairs of nodes; and a distinguished set of terminal nodes $V' \subseteq V$. The goal is to find a minimum-cost tree that includes the vertices $V'$. The tree may or may not include nodes in $V \setminus V'$.

![Graph](image.png)

Suppose the distances in the input satisfy the conditions of a metric:

(i) $d_{uv} \geq 0$ for all $u, v$,
(ii) $d_{uv} = 0$ if and only if $u = v$,
(iii) $d_{uv} = d_{vu}$ for all $u, v$,
(iv) $d_{uv} \leq d_{uw} + d_{wv}$ for all $u, v, w$ (triangle inequality).
Show that an efficient ratio-2 approximation algorithm for Minimum Steiner Tree can be obtained by ignoring the nonterminal nodes and simply returning a minimum spanning tree on $V'$.  
(Hint: Recall the approximation algorithm for metric TSP presented at Lecture 11.)

4. [Dasgupta et al., Ex. 9.8] In the MAX-SAT problem, we are given a set of clauses, and we want to find an assignment that satisfies as many of them as possible.
   
   (a) Show that if this problem can be solved in polynomial time, then so can SAT.
   
   (b) Here's a very naive algorithm:
      
      for each variable:
      set its value to either 0 or 1 by flipping a coin

Suppose the input has $m$ clauses, of which the $j$th has $k_j$ literals. Show that the expected number of clauses satisfied by this simple algorithm is

$$\sum_{j=1}^{m} \left(1 - \frac{1}{2^{k_j}}\right) \geq \frac{m}{2}.$$  

In other words, this is a 2-approximation in expectation! And if the clauses all contain $k$ literals, then this approximation factor improves to $1 + 1/(2^k - 1)$.

5. Consider a Monte Carlo algorithm $A$ for a problem $\Pi$ whose running time is at most $O(T(n))$ on any instance of size $n$ and that outputs a correct solution with probability at least $p(n)$. Suppose further that, given a solution to $\Pi$, we can verify its correctness in time $O(t(n))$. Show how to obtain a Las Vegas algorithm that always outputs a correct solution and runs in expected time at most $O\left((T(n) + t(n))/p(n)\right)$.

**Demonstration problems:**

6. [Dasgupta et al., Ex. 5.34] Show that for any integer $n$ that is a power of 2, there is an instance of the Set Cover problem with the following properties:

   (i) There are $n$ elements in the base set.
   
   (ii) The optimal cover uses just two sets.
   
   (iii) The greedy algorithm may pick $\log_2 n$ sets.

Thus the logarithmic approximation ratio derived for the greedy set cover algorithm at Lecture 11 is tight.

7. Random permutations and random bits.

   (a) Suppose you have access to a subroutine that, given a positive integer $k$ as input, returns an integer drawn uniformly at random from $\{1, 2, \ldots, k\}$. Design an algorithm that, given a positive integer $n$ as input, returns a permutation of the integers $1, 2, \ldots, n$ drawn uniformly at random from the set of all such permutations.

   (b) Suppose you have access to a source of independent uniform random bits. Describe a (Las Vegas) implementation of the subroutine.

**Challenge problem:**

8. Can you make the MAX-SAT approximation algorithm in Problem 4 deterministic?  (Hint: Express the expected number of satisfied clauses using two conditional expectations. At least one of the conditional expectations has to be at least the expected number of satisfied clauses.)