1. Professor Tournesol proposes the following heuristic to solve vertex-cover problem:

- **Repeat** until there is no more edges:
  - select a vertex of largest degree, and remove all of its incident edges.

(a) Give an example to show that it is not a 2-approximation algorithm.
(b) Give an example for which the heuristic gives log \(n\) ration.

2. **Vertex cover in tree 1**

*The goal of this exercise is to compute efficiently an optimal solution for the vertex cover problem when the input is a tree.*

Let \(T\) be a tree.

(a) Prove that there exists an optimal solution for vertex cover that does not contain any leaf of \(T\).
(b) Describe a linear-time algorithm that compute an optimal solution for vertex cover when the input is a tree.

3. **Vertex cover in tree 2**

Same Question as in the previous exercise, but this time design an algorithm based on dynamic programming.

*FYI:* This method is of crucial importance to solve (almost all) problems on trees and more generally on graph with bounded treewidth, where the treewidth is a measurement of how much a graph looks like a tree.

4. **Tightness of the approximation ratio of the greedy algorithm for the set cover problem.**

Recall that we have seen in class a greedy heuristic giving a log \(n\) approximation ratio for the weighted set covering problem. But are we sure that this greedy heuristic does not actually compute a solution with a better ratio? We are going to prove that the answer is no.

(a) Find an (edge)-weighted hypergraph for which the greedy heuristic give a solution of size at least \((\log(n) - o(1)) \cdot OPT\)
(b) Find a (non-weighted) hypergraph for which the greedy heuristic give a solution of size \(\frac{\log(n) - 2}{2} OPT\).

**Observation:** it is often a hard question to find the tight approximation ratio of a greedy heuristic, recall the open question on the superstring problem!
5. **Greedy vertex coloring**

A vertex coloring of a graph $G$ is an assignment of a color to each vertex such that two adjacent vertices receive distinct colors. The chromatic number of a graph $G$, denoted by $\chi(G)$ is the minimum number of colors needed to color $G$. Traditionally, we color a graph with integers, i.e. a $k$-coloring of a graph is a coloring of the vertices with colors in $\{1, 2, \ldots, k\}$

(a) Give a greedy algorithm for coloring a graph $G$ with at most $\Delta(G) + 1$ colors.

(b) Is there graphs for which $\chi(G) = \Delta(G) + 1$?

(c) Give an algorithm for coloring a 3-colorable graph with $O(\sqrt{n})$ colors. (FYI: 3-coloring is a NP-complete problem, there is not even constant ratio approx algorithm).

6. **TSP and fake proof that $P = NP$**

(a) Show how in polynomial time we can transform an instance $(G, c)$ of the traveling-salesman problem into another instance whose cost function satisfies the triangle inequality. The two instances must have the same set of optimal tours.

(b) We proved in class that there was no constant ratio approx algorithm for the general TSP unless $P = NP$. Moreover, we have seen a 2-approx for the metric TSP. Does question (a) implies that $P = NP$?

7. **(Bottleneck traveling-salesman problem)**

We want to find the hamiltonian cycle that minimizes the cost of the most costly edge in the cycle. Assuming that the cost function satisfies the triangle inequality, show that there exists a polynomial-time approximation algorithm with approximation ratio 3 for this problem.

---

$\Delta(G)$ is the maximum degree of $G$