1. Let $P$ be a simple (but not necessarily convex) polygon with $n$ vertices. Explain how to find a triangulation’s so the sum of squares of the triangles is as small as possible.

2. Let $G(A \cup B, E)$ be a bipartite and let $M^* \subseteq E$ be a maximum-cardinality matching. We suggest the following greedy algorithms for finding a large matching $M$. We start with $M = \emptyset$, and at each itetions, we find, we find an edge $(u, v) \in E$ where neither $u$ nor $v$ are matched. If no such edge is found, the algorithm terminates. Otherwise, we add $(u, v)$ to $M$, and repeat.

   (a) show how to implement the algorihtm in $O(m+n)$ time. (hint - similar to the greedy vertex cover algorithm)

   (b) Prove that when the algorithm terminates $M^*/2 \leq M \leq M^*$.

   Hint: recall that $M^*$ exists. Let edges of $M$ be charged for edges of $M^*$ that are deleted.

3. Let $G(A \cup B, E)$ be a bipartite and let $M^* \subseteq E$ be a maximum-cardinality matching. weight graph (each edge $(u, v) \in E$ is given with a weight $w(u, v) > 0$). The maximum weighted bipartite matching is to pick a matching $M^* \subseteq E$ whose sum of weights is as larger as possible. Show that a variant the algorithm from the previous question would give a factor $1/2$ approximation. That is, it will find a matching $M$ such that

   $$w(M) \leq 2w(M^*)$$

where $w(M)$ is the sum of weights of edges in $M$.

4. Let $X, Y, Z$ be the sequences of characters (E.g. $X = "aaabcadex"$). Assume $n = |X| = |Y| = |Z|$. Suggest an algorithm that computes a longest sequence $W$ such that $W$ is a subsequence of $X$, subsequence of $Y$ and a subsequence of $Z$. What is the running time of the algorithm? What is the space (memory) requirement? Will you change your answer is instead of seeking $W$, we just want to know $|W|$?

   What would be the running time, as a function of both $m$ and $n$, if the input contains $m > 3$ sequences (rather than 3)?

5. Let $P = \{p_1 \ldots p_n\}, Q = \{q_1 \ldots q_n\}$ be given $n$-gons, and let $k$ be a given value. Explain how to find $DTW(P, Q)$ in $O(nk)$ time, if it is known that for every $i$, when the person is on $p_i$, the dogs could be only on one of the vertices

   $$\{q_{i-k}, q_{i-k+1}, \ldots q_{i+k}\}$$

That is, on a vertex of $Q$ between $q_{i-k}$ and $q_{i+k}$.

6. Let $X, Y$ be two given sequences of $n$ characters each. Let $t$ be your lucky number. We say that $X$ is similar to $Y$ iff $ed(X, Y) \leq t$.

   Here we assume that each operation (insert/delete/replace) cost 1 unit.
So given $X, Y$ suggest an $O(tn)$ time algorithm that either reports that $X$ and $Y$ are not similar, or if they are similar, compute what is exactly $ed(X,Y)$.

Hint: Use the previous question.

7. Let $X,Y$ be two given sequences, each of length $n$. Suggest an $O(nk)$ algorithm for computing $ed(X,Y)$. Here $k = ed(X,Y)$ and is not known to you in advance.

Hint: Use the previous question.