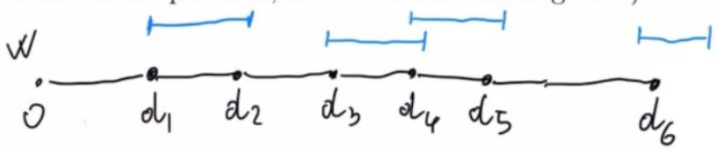
Solve the following exercises.

1. Let $G(V, E, w)$ be an undirected weighted graph, that is, let V be the graph's set of nodes, let E be the graph's set of edges, and let $w : E \to \mathbf{R}^+$ be the function that assigns positive weights to edges. (For instance, $V = \{1, 2, 3\}$, $E = \{\{1, 2\}, \{2, 3\}\}$, $w(\{1, 2\}) = 1.5$ and $w(\{2, 3\}) = 1$.)

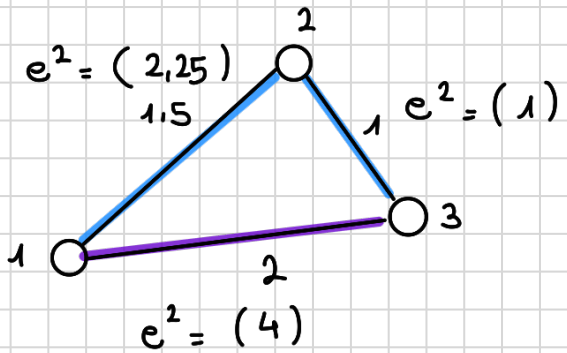
   Consider the following claim: *"For any two nodes v, v' of V, if π is a shortest path in $G(V, E, w)$ from v to v', then π is a shortest path from v to v' even in the graph $G(V, E, w')$, where $w'(e) = w(e)^2$ for each $e \in E$. (That is, the sequence of nodes of a shortest path does not change, if we square the weight of the edges.)"*

   Your task is to determine whether the claim is true or false. I.e., either prove the claim, or give a counterexample.

2. Consider a long rectilinear road, of length t km, on the west-to-east axis. There are k houses along this road, the first of which is at distance d_1 km from the western endpoint w of the road, the second of which at distance d_2 from w, ..., and the last of which is at distance d_k from w. A phone company would like to build antennas along the road so that each house can be served by its GSM network. If each antenna covers a radius of 10 km, what is the smallest number of antennas required for covering each house? Give an efficient algorithm for this problem (prove that it solves the problem, and bound its running time).



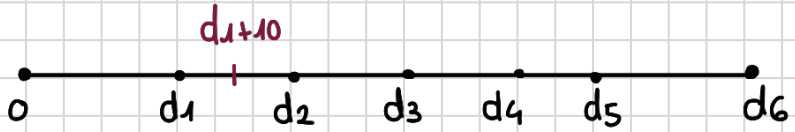
ex. 1. FALSE. Counterexample:



— shortest path before e^2

— // // after e^2

ex. 2



Put the antenna at $d_1 + 10$ and remove the covered houses. Look at the 1st house on the left that needs to be covered and push the antenna as right as possible, but being sure that the leftmost house is covered and keep repeating this (pushing the solution as much as possible to the right). ~ Greedy

$S = \{s_1, \dots, s_k\}$ # set of antennas placed by our greedy algo

$T = \{t_1, \dots t_m\}$ # set of antennas positions given by an optimal solution (sorted from w to E).

We want to prove that $k = m$. (Greedy stays ahead)

claim: $s_i \geq t_i$ for each i.

The claim is true for $i = 1$ since we go as far as possible to east before placing the 1st antenna.

Let's assume that it's true for some $i \geq 1$; this means that our algorithm's first i antennas cover all the houses covered by the first i antennas $(t_1, \dots, t_i)$.

If we add for example t_{i+1} to $\{s_1, \dots, s_i\}$, there won't be any house between s_i and t_{i+1} uncovered.

The $(i+1)$ first step of our greedy sol. chooses s_{i+1} to be as large as possible but with the condition to cover the houses between s_i and s_{i+1}; so $s_{i+1} \geq t_{i+1}$.

Induction.

If $k > m$, then $\{s_1, \ldots, s_m\}$ fails to cover all houses. But $s_m \geq t_m$, and so $\{t_1, \ldots, t_m\} = T$ also fails to cover all houses, a contradiction.

July 7, 2021

Solve the following exercises.

1. Consider the following statement: "*Let $G(V, E, w)$ be a connected undirected graph with a distinct* positive *weight $w(e)$ on each edge $e \in E$. Let x, y be two distinct nodes of $G(V, E, w)$. Then, if e^* is the edge of minimum weight in E, then each shortest path from x to y contains the edge e^*.*"

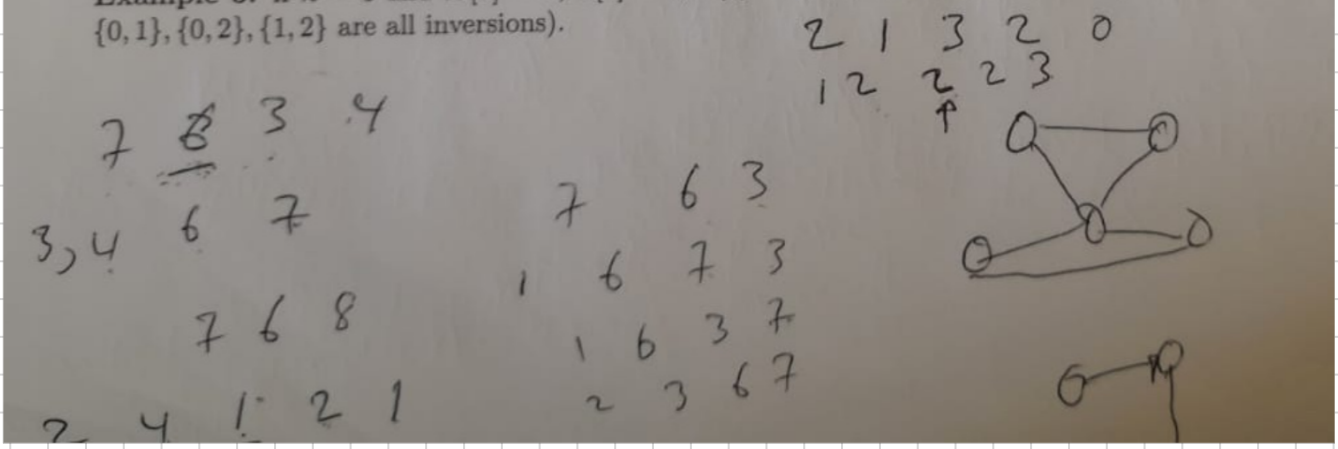
   Determine whether the statement is true or false: if it is true, prove it; if it is false, give a counterexample.

2. Let X be an array of n distinct integers. An inversion of X is a pair of indices $0 \leq i < j \leq n-1$ such that $X[i] > X[j]$. (Clearly, X can have anywhere from 0 to $\frac{n^2-n}{2}$ inversions). Give an algorithm to compute the number of inversions of X. Prove that your algorithm is correct, and bound its running time. *Larger scores will be awarded to faster solutions.*

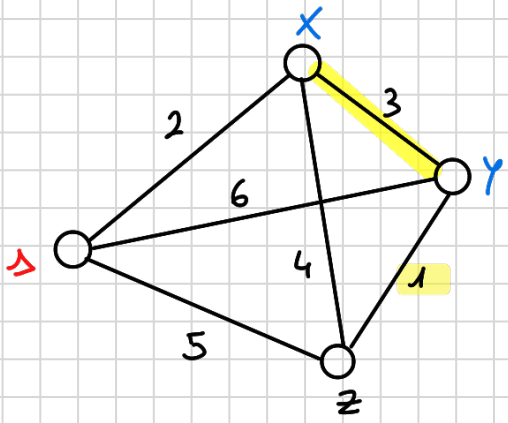
   Example 1: if $n = 3$ and $X[0] = 5, X[1] = 7, X[2] = 6$, then X has 1 inversion (the one given by the pair of indices $\{1, 2\}$).

   Example 2: if $n = 3$ and $X[0] = 2, X[1] = 4, X[2] = 9$, then X has 0 inversions (indeed, X is sorted increasingly).

   Example 3: if $n = 3$ and $X[0] = 7, X[1] = 6, X[2] = 3$, then X has 3 inversions (that is, $\{0, 1\}, \{0, 2\}, \{1, 2\}$ are all inversions).

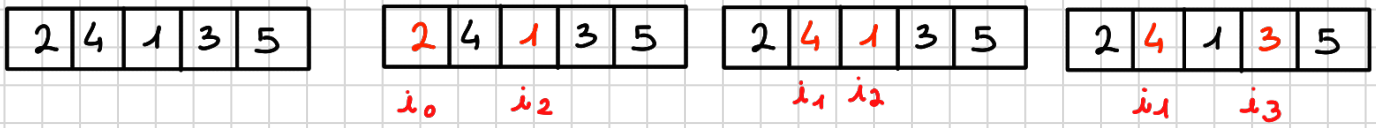


ex. 1 FALSE. Counterexample:



ex. 2 Two elements a[i] and a[j] form an inversion if a[i] > a[j] and i < j

Example: the sequence 2, 4, 1, 3, 5 has three inversions (2,1), (4,1), (4,3)



Inversion count for an array indicates how far (or close) the array is from being sorted.

If the array is already sorted, then the inversion count is 0.
If the array is sorted in reverse order, the inversion count is maximum.

```
int InversionsCounter(int arr[], int n)
{
    int inv_count = 0;
    for (int i = 0; i < n-1; i++)
        for (int j = i+1; j < n; j++)
            if (arr[i] > arr[j])
                inv_count++;
    return inv_count;
}
```

iterative method

$O(n^2)$