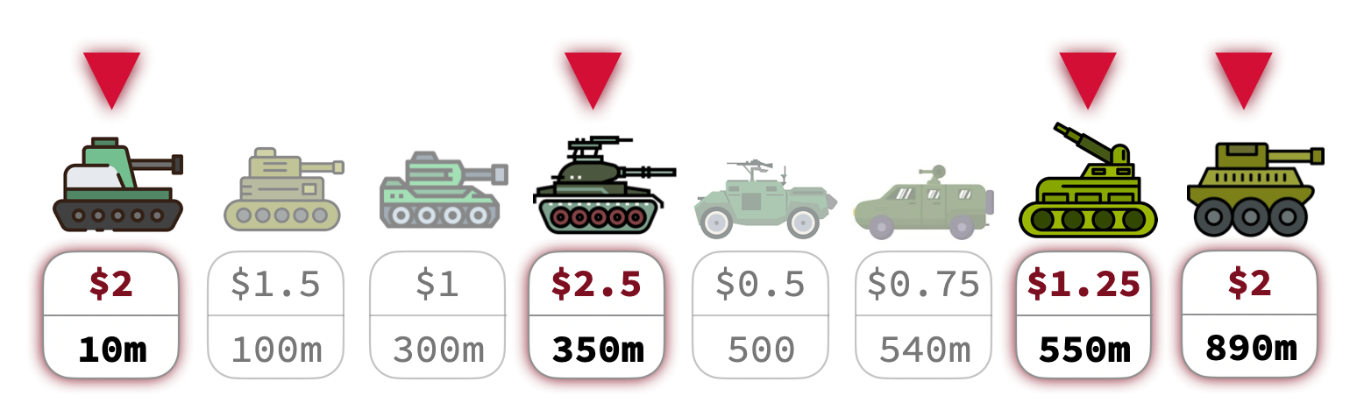
Ex1. Many Red Triangles

🍭 50 points



**Overview**

A group of Palestinian fighters are planning a coordinated attack on a group of Israeli tanks that are lingering close to the entrance of a tunnel. Each of these tanks has a different value (in million dollars). The fighters have orders to avoid targeting tanks that are very close to each other, as this increases the danger of the fighters themselves getting hurt. The fighters are deciding on a way to pick which tanks to destroy, such that monetary loss is maximized.

**Task Description**

You are given the following:

* A sequence of �*n* non-negative strictly-increasing integers {�1,�2,…,��}{*d*1​,*d*2​,…,*dn*​} representing the distances of the tanks (in meters) from the entrance of the tunnel the fighters are based at.
* A sequence of �*n* non-negative integers {�1,�2,…,��}{*v*1​,*v*2​,…,*vn*​} representing the monetary value of each tank.

Your task is to implement a dynamic programming algorithm (memoized or bottom-up) to find which tanks the fighters must destroy in order to maximize the value of the loss.

**Constraints**

No two picked tanks can be ⩽⩽ 100 meters away from each other.

Assignment Requirements

Modify **triangles.cpp** to implement the following functions:

* **double max\_loss(int n, int d[], double v[])** , where **n** is the number of tanks, **d[]** is the array of distances (**d[i]** is the distance of tank �*i* from the tunnel entrance) and **v[]** is the array storing the value of each tank (in millions).  
  The function must *return* the maximum total loss value possible for the given tanks.
* **string which\_tanks(int n, int d[], double v[])** : The function must return a string of bits representing which tanks that will become khorda, where 11 at position �*i* in the string means that tank �*i* will be destroyed and 00 means the tank will not be targeted for now.

**NOTE.** The above two functions are independent. Do not assume that **max\_loss** will be called before **which\_tanks**.

**Examples**

**Example 1**

**Input:**

d[] = {1, 2, 3, 4}

v[] = {1.5, 3, 0.5, 2.75}

**Result**:

max\_loss = 3 (million)

which\_tanks = 0100

**Explanation:** Since none of the tanks is more than 100 meters away from any other tank, only one tank can be targeted. The second tank is picked because it has the highest value. This makes the maximum loss possible to be 3 (million).

**Example 2**

**Input:**

d[] = {100, 150, 250, 300}

v[] = {1, 2, 0.75, 4}

**Result**:

max\_loss = 6 (million)

which\_tanks = 0101