



# [CS 11] Prac 10a – Match Making II

## Problem Statement

An Esports team wants to continue training their members for the popular game Protection of the Ancients (PotA).

The team consists of  $n$  players with varying skill levels. And again, even though the game is team-based, this particular training will focus on one-on-one combat. So the plan is for the players to be paired up.

Each player can only participate in one match for this training, but because time is limited, the number of pairs must be maximized. Furthermore, for the training to be effective, the gap between the skills of players in the match-ups should be small.

Among all possible ways to pair the players up with the maximum number of pairs, what is the minimum *sum* of the absolute differences between the paired-up players' skill levels?

[Submit solution](#)

[CS 11]

Practice 10

✓ Points: 215 (partial)

⌚ Time limit: 5.0s

💻 Memory limit: 1G

✉ Author:

kvatiienza (Kevin Atienza)

➤ Problem type

▼ Allowed languages

NONE, py3

## Task Details

Your task is to implement a function called `optimal_matchups`. This function has a single parameter, a `tuple` / `list` of  $n$  `int`s representing the skill levels of the players.

The function must return an `int` denoting the minimum *sum* of the absolute differences between the paired-up players' skill levels, among all possible ways to pair the players up with the maximum number of pairs.

## Restrictions

In this lab session, some names are banned. Here are a few of them: `input`, `type`. This is *not* an exhaustive list. (If you accidentally use a variable name that turns out to be banned, please rename it.)

Anonymous functions are now allowed.

For this problem in particular:

- The source code limit is 2000.

## Example Calls

### Example 1 Function Call

```
optimal_matchups((1, 7, 2, 10, 4, 7))
```

[Copy](#)

### Example 1 Return Value

```
7
```

[Copy](#)

### Example 1 Explanation

In this example, there are 6 players. The maximum number of pairs is 3. If we do the following pairing:

- first and third players.
- fourth and fifth players.
- second and sixth players.

Then we achieve a *sum* of absolute differences of  $|1 - 2| + |10 - 4| + |7 - 7| = 1 + 6 + 0 = 7$ . It can be shown that this is the minimum such sum that can be obtained.

### Example 2 Function Call

```
optimal_matchups([3, 1, 4])
```

[Copy](#)

### Example 2 Return Value

```
1
```

[Copy](#)

### Example 2 Explanation

In this example, there are 3 players. The maximum number of pairs is 1. If we do the following pairing:

- first and third players.

Then we achieve a *sum* of absolute differences of  $|3 - 4| = 1$ . It can be shown that this is the minimum such sum that can be obtained.

## Constraints

- The function `optimal_matchups` will be called at most 60,000 times.
- The sum of  $ns$  across all calls will be  $\leq 250,000$ .
- $0 \leq n \leq 250,000$
- Each skill level is an integer between 1 and  $10^{10}$ .

## Scoring

- You get 30 points if you solve all test cases where:

- $n \leq 50$

- $n$  is even.

- The sum of the  $ns$  across all calls will be 500.

- You get 40 points if you solve all test cases where:

- $n \leq 50$

- The sum of the  $ns$  across all calls will be 500.

- You get 25 points if you solve all test cases where:

- $n \leq 4,000$

- The sum of the  $ns$  across all calls will be 8,000.

- You get 45 points if you solve all test cases where:

- $n$  is even.

- You get 75 points if you solve all test cases.

## Clarifications

[Report an issue](#)

No clarifications have been made at this time.