Hacker Devils Easy Meeting

September 17, 2025

Meeting Agenda

- AMI Presentation
- Setting Up LeetCode
- Problems
- Pizza!

Attendance:



Arizona Microcredit Initiative

AMI
ARIZONA MICROCREDIT INITIATIVE

JOIN THE TEAM!

We're a **student-led nonprofit organization** dedicated to empowering local small businesses through:

- Microloans
- Free Consulting
- Free Business Education

Why Join?

Work Directly with Small Business Owners to Build Your Consulting and Financial Knowledge

EMAIL FAIZAN@AZMICROCREDIT.ORG FOR ANY QUESTIONS!

APPLY BY SEPT. 21ST

LINK IN BIO!

I WEEK TABLING

- *DEAN'S PATIO,* SEPT 15TH - *18TH, (11:30 AM-2:30 PM)*

1ST INFO SESSION

- *COTTONWOOD 101,* SEPT. *17TH (5:30-6:30 PM)*

2ND INFO SESSION

- *GOOGLE MEETS,* SEPT. *19TH (5:30-6:30 PM)*

Join HKD Discord:



Setting Up HackerRank

- https://leetcode.com
- Create an account if you don't have one



Warm-Up: Compare the Triplets

Alice and Bob each created one problem for HackerRank. A reviewer rates the two challenges, awarding points on a scale from 1 to 100 for three categories: problem clarity, originality, and difficulty.

The rating for Alice's challenge is the triplet a = (a[0], a[1], a[2]), and the rating for Bob's challenge is the triplet b = (b[0], b[1], b[2]).

The task is to calculate their comparison points by comparing each category:

- If a[i] > b[i], then Alice is awarded 1 point.
- If a[i] < b[i], then Bob is awarded 1 point.
- If a[i] = b[i], then neither person receives a point.

https://www.hackerrank.com/challenges/compare-the-triplets/problem?isFullScreen=true

- Read the description of the problem on Hackerrank

Warm-Up: Compare the Triplets (explaining the problem)

- The input consists of two arrays of integers:
  - One array contains 3 ratings for different aspects of Alice's problem
  - The other array contains 3 ratings for the same aspects of Bob's problem
- The task is to give a score to Alice and a score Bob based on who got a higher rating in each aspect of the problem
- The scores of Alice and Bob are returned by the function in a third array containing of 2 integers (each integer representing a score)

BUT WAIT! We need to learn about List<>s first

Warm-Up: Compare the Triplets (List<>s explained)

- List<> is what is called a generic type in the Java Collections Framework
- We don't need to dive much into details, but here is what you need to know for this problem:
  - A List<> object is an ordered sequence of elements (basically an array)
  - To initialize an object with this type, consisting of integer elements, named NewList:
    - List<Integer> NewList = new ArrayList<Integer>();
  - To add a new number, say 1, to NewList:
    - NewList.add(1);
  - To obtain an element from NewList at a certain index(eg.3):
    - element = NewList.get(3);
  - To replace a number at that index with another number:
    - NewList.set(3, 6);

This is similar to element = arr[3] When you're getting an element from an array

Warm-Up: Compare the Triplets (List<>s explained)

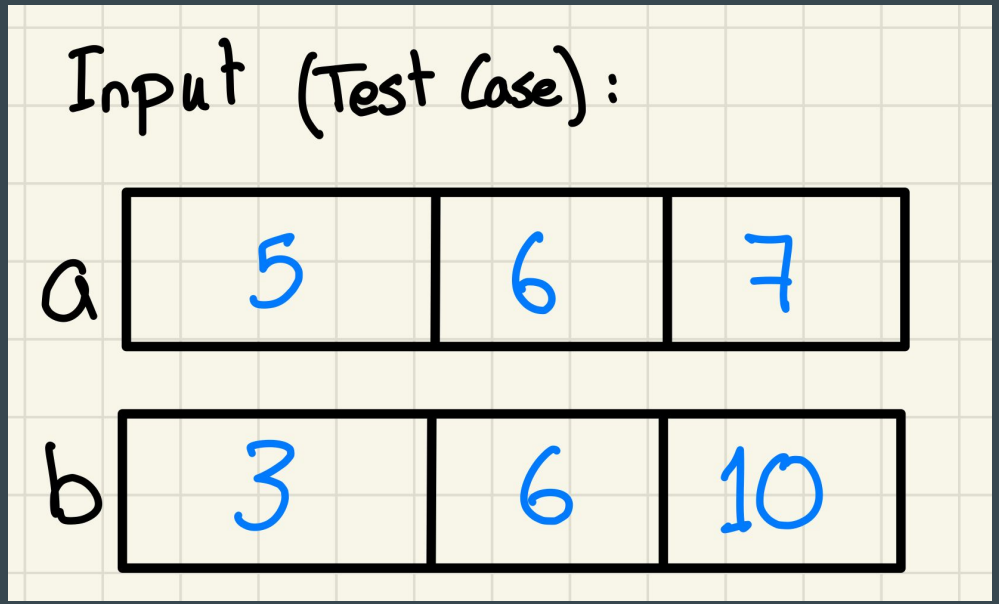
ALSO,

Bear in mind that Lists<> when initialized are empty, so you have to populate them using the .add(n) method (where n is an integer)

Good Luck!

Ps. if you want to learn more about lists in Java:
https://www.geeksforgeeks.org/java/initializing-a-list-in-java/

Warm-Up: Compare the Triplets (Solution Steps)



We will run the solution on this test case.

Warm-Up: Compare the Triplets (Solution Steps)

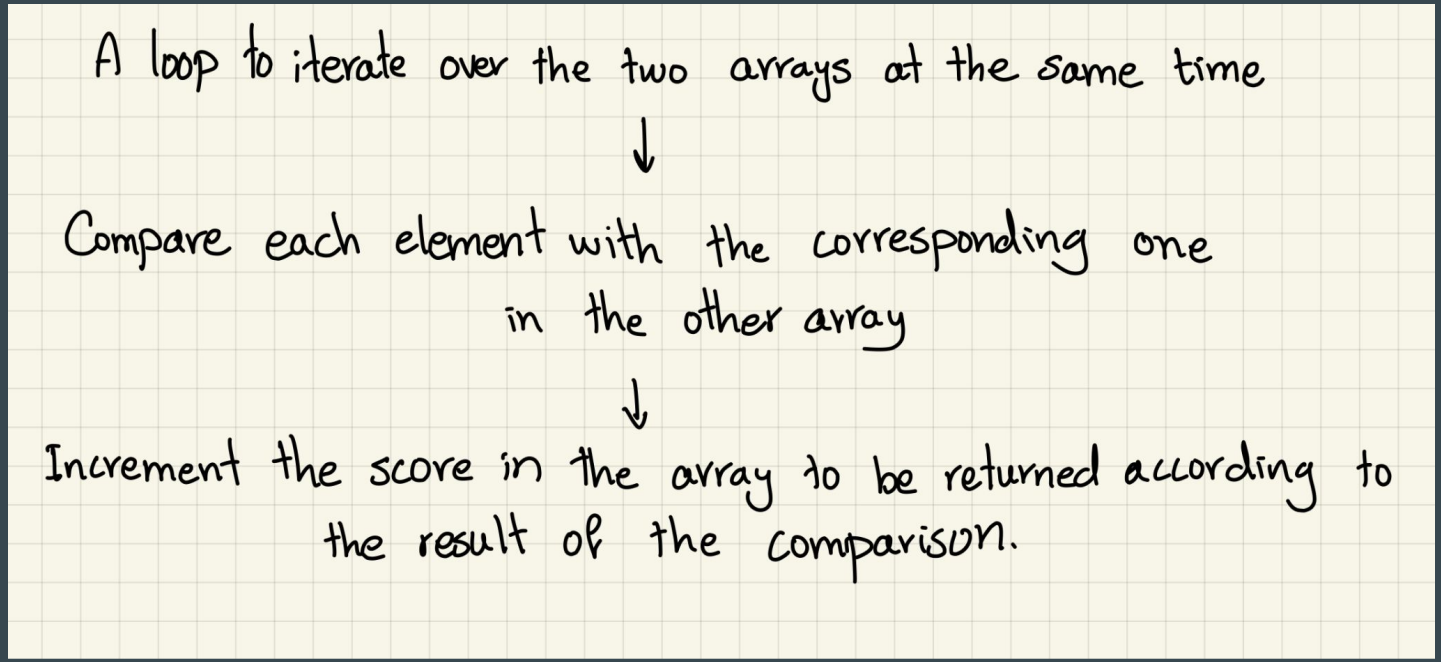
First of all, let's declare, initialize, and populate the List that we will return at the end of the function.

```
List<Integer> arr = new ArrayList<Integer>();
arr.add(0); arr.add(0);
```

We populated it with 2 elements according to the problem description. The first element represents Alice's score, and the second element represents Bob's score

Warm-Up: Compare the Triplets (Solution Steps)

These are the overall steps of the solution in plain English:

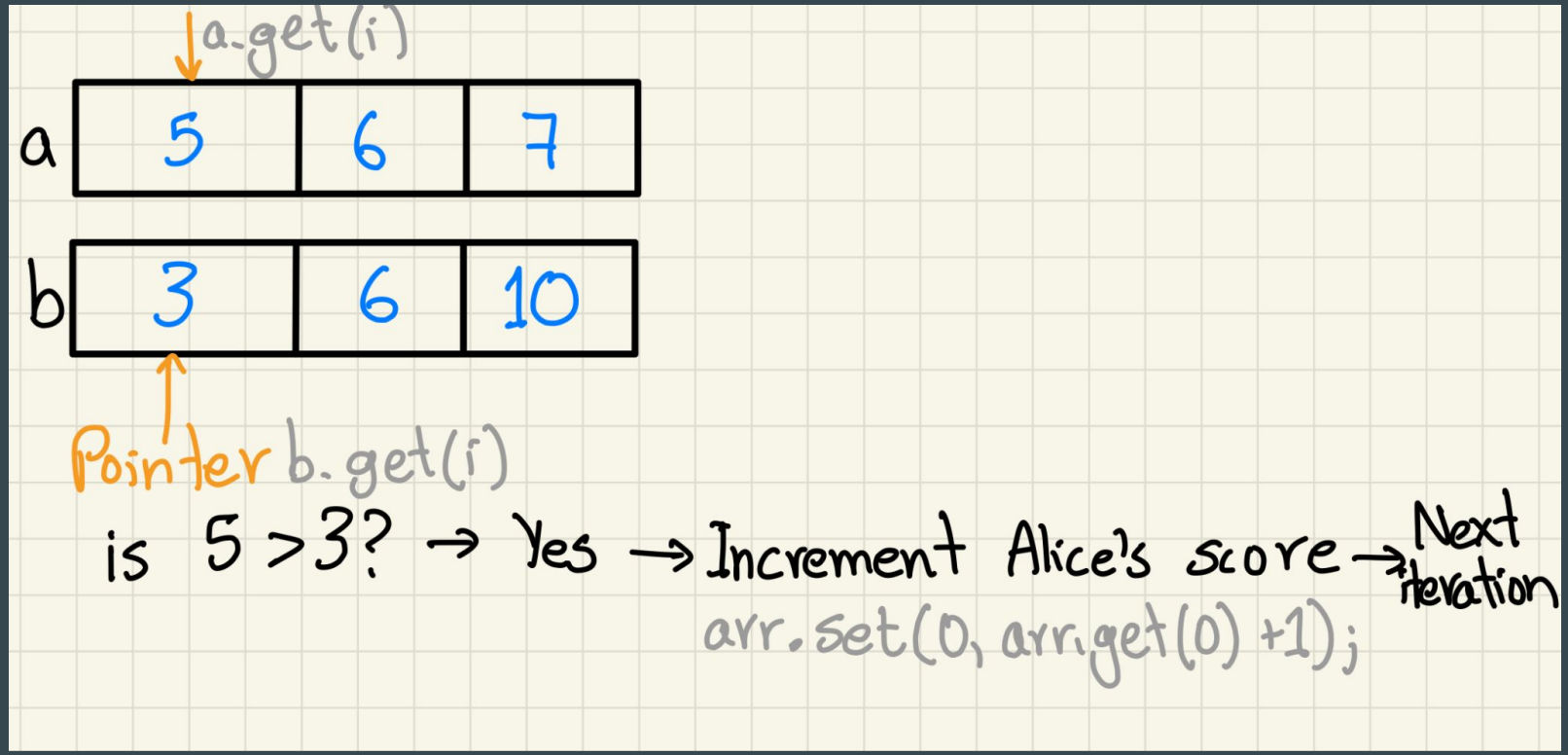


Warm-Up: Compare the Triplets (Solution Steps)

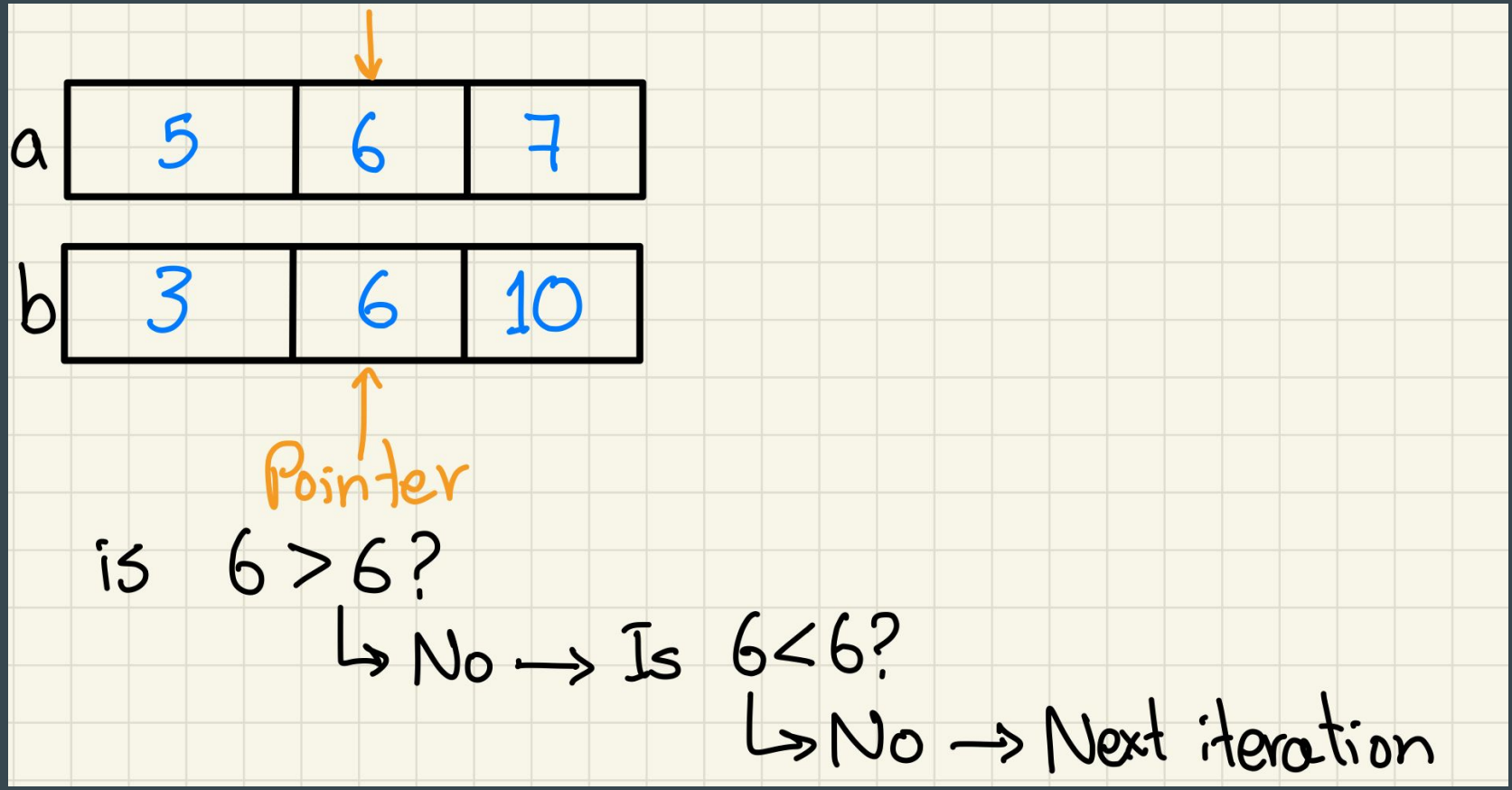
We will use a for loop, with an index (i) that starts at zero and ends before 3 (at 2), since we have 3 elements

```
for (int i = 0; i < 3; i++){
```

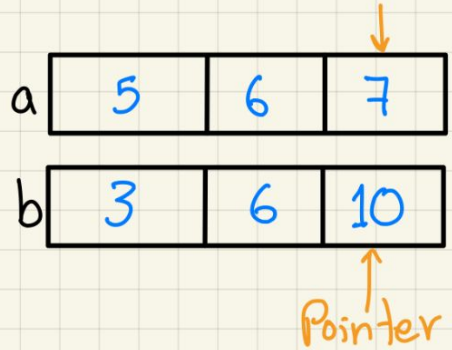
Warm-Up: Compare the Triplets (Solution Steps)



Warm-Up: Compare the Triplets (Solution Steps)



Warm-Up: Compare the Triplets (Solution Steps)



is 7 > 10?

↳ No → Is 7 < 10? → Yes → Increment Bob's score → Next iteration (Loop is over since i = 2)

```
arr.set(1, arr.get(1)+1);
```

↓

```
return arr;
```

Warm-Up: Compare the Triplets (Final Solution)

```java
public static List<Integer> compareTriplets(List<Integer> a, List<Integer> b) {
        List<Integer> arr = new ArrayList<Integer>();
        arr.add(0); arr.add(0);
        for (int i = 0; i < 3; i++){
            if (a.get(i) > b.get(i)){
                arr.set(0, arr.get(0) + 1);
            }else if(a.get(i) < b.get(i)){
                arr.set(1, arr.get(1) + 1);
            }
        }
        return arr;
}
```

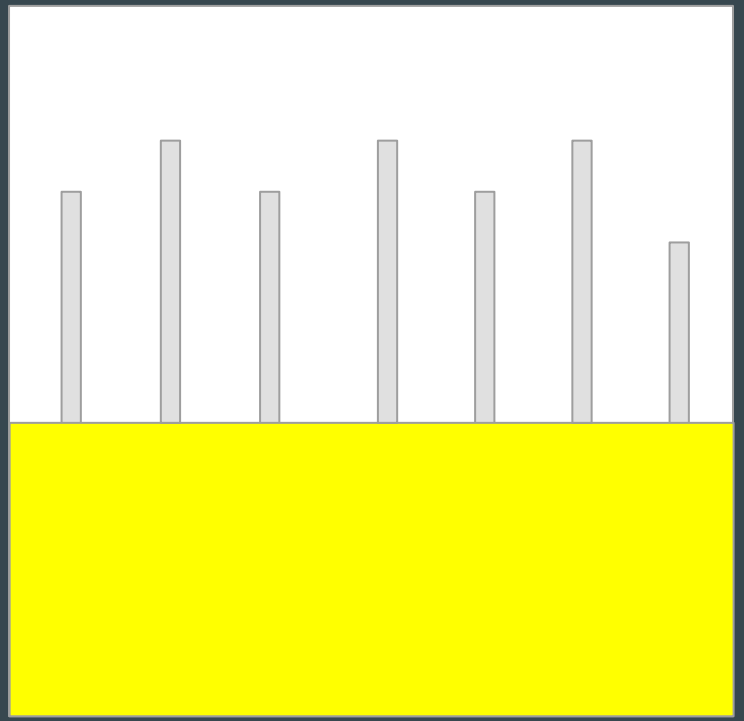
Problem 1: Birthday Cake Candles

https://www.hackerrank.com/challenges/birthday-cake-candles/problem?isFullScreen=true

Look at the candles on the right. How many candles are the tallest? When looking left to right, were they always the tallest candles?

You can use candles.get(index) to find a specific candle in the List.

Template Code for for loop:



Problem 1: Birthday Cake Candles

What information is needed to solve this problem?

- Height of the tallest candle
- Number of times that height appears in the list

```
int tallest = 0;
int count = 0;
```

We can acquire this information with for loops:

- One to get the tallest candle
- One to get the number of times it appears

For loop template:

```
for (int i = 0; i < candles.size(); ++i) {
        //Your code here
}
```

Can we get this information with just one for loop?

Problem 1 Solution

```
public static int birthdayCakeCandles(List<Integer> candles) {
// Write your code here
int tallest = 0;
int count = 0;
for (int i = 0; i < candles.size(); ++i) {
    if (candles.get(i) > tallest) {
        tallest = candles.get(i);
        count = 1;
    } else if (candles.get(i) == tallest) {
        ++count;
    }
}
return count;

}
```

A single for loop can solve this solution too! Just remember to reset the count of the tallest candle back to 1 when a new tallest candle is found.

PIZZA TIME!!!!