

## Experiment 4(A)

**Student Name: Tanmaya Kumar Pani**

**UID: 22BCS12986**

**Branch: BE-CSE**

**Section/Group: IOT-613B**

**Semester: 5**

**Date of Performance: 13/8/24**

**Subject Name: Advanced Programming Lab-1**

**Subject Code: 22CSP-314**

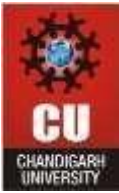
### **1. Title: Missing Numbers**

### **2. Objective:**

Given two arrays of integers, find which elements in the second array are missing from the first array.

### **3. Algorithm:**

- a) Input: Two vectors, arr and brr.
  - arr: A subset of brr with some elements missing.
  - brr: The full array with all elements.
- b) Sort Both Arrays:
  - Sort both arr and brr to compare them easily.
- c) Initialize Pointers and Result:
  - Create an empty vector missing to store missing numbers.
  - Initialize two pointers, i for arr and j for brr.
- d) Iterate through Both Arrays:
  - Compare elements at arr[i] and brr[j].
    - If arr[i] == brr[j], move both pointers forward.
    - If arr[i] < brr[j], increment pointer i because the current element in arr is smaller and isn't missing.
    - If arr[i] > brr[j], brr[j] is missing in arr. Add brr[j] to the missing vector if it's not a duplicate and move j forward.
- e) Continue Until All Elements in brr Are Checked:
  - Repeat the comparison and pointer increment until all elements of brr are processed.
- f) Return Result:
  - The missing vector contains all the numbers present in brr but missing from arr.



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## 4. Implementation/Code:

```
16  */
17
18  vector<int> missingNumbers(vector<int> arr, vector<int> brr) {
19  // Sort both arrays
20  sort(arr.begin(), arr.end());
21  sort(brr.begin(), brr.end());
22
23  vector<int> missing;
24  int i = 0, j = 0;
25
26  // Iterate through both sorted arrays
27  while (j < brr.size()) {
28  if (i < arr.size() && arr[i] == brr[j]) {
29  // If elements match, move both pointers
30  i++;
31  } else if (i < arr.size() && arr[i] < brr[j]) {
32  // If arr[i] is smaller, just move pointer i
33  i++;
34  } else {
35  // Otherwise, brr[j] is missing in arr
36  if (missing.empty() || missing.back() != brr[j]) {
37  missing.push_back(brr[j]);
38  }
39  }
40  j++;
41  }
42
43  return missing;
44  }
45
```

Line: 17 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

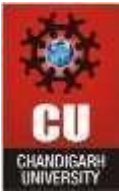
Person 1

Tanmaya Pani  
tanmaya0730@gmail.com

Sync is on  
Manage your Google Account  
Close 2 windows

Other profiles

- Tanmaya (Work)
- Tanmaya Kumar
- Guest
- Add



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## 5. Output:

Upload Code as File

☐ Test against custom input

Run Code

Submit Code

### Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

✓ Sample Test case 0

✓ Sample Test case 1

Input (stdin)

Download

1	10
2	203 204 205 206 207 208 203 204 205 206
3	13
4	203 204 204 205 206 207 205 208 203 206 205 206 204

Your Output (stdout)

Download

1	204 205 206
---	-------------

Expected Output

Download

1	204 205 206
---	-------------

## 6. Learning Outcomes:

- **Understand Sorting and Comparison:** Learn how sorting helps in efficiently comparing two arrays to find discrepancies, like missing numbers.
- **Efficient Iteration Techniques:** Understand the use of two-pointer techniques to iterate over sorted arrays for optimized performance.
- **Handling Duplicates:** Learn how to avoid adding duplicate missing elements by checking the previous entries.

7. Time Complexity:  $O(n \log n + m \log m)$

8. Space Complexity:  $O(m)$