Aryan   **22BCS10190**   **KPIT-901(A)**

common element in 3 sorted arrays

```cpp
#include <iostream>
#include <vector>
using namespace std;

void findCommonElements(const vector<int>& arr1, const vector<int>& arr2, const vector<int>& arr3) {
    int i = 0, j = 0, k = 0;

    cout << "Common elements: ";
    while (i < arr1.size() && j < arr2.size() && k < arr3.size()) {
        if (arr1[i] == arr2[j] && arr2[j] == arr3[k]) {
            cout << arr1[i] << " ";
            i++;
            j++;
            k++;
        }
        else if (arr1[i] < arr2[j]) {
            i++;
        }
        else if (arr2[j] < arr3[k]) {
            j++;
        }
        else {
            k++;
        }
    }
    cout << endl;
}

int main() {
    vector<int> arr1 = {1, 5, 10, 20, 40, 80};
    vector<int> arr2 = {6, 7, 20, 80, 100};
    vector<int> arr3 = {3, 4, 15, 20, 30, 70, 80, 120};

    findCommonElements(arr1, arr2, arr3);

    return 0;
}
```

```
Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\class>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\" && g++ day5ques1.cpp -o day5ques1 && "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\"day5ques1
Common elements: 20 80

c:\Users\Gaurav Kumar\OneDrive\Desktop\class>
```

search in 2d matrix

```cpp
#include <iostream>
#include <vector>
using namespace std;

bool searchMatrix(const vector<vector<int>>& matrix, int target) {
    if (matrix.empty() || matrix[0].empty()) {
        return false;
    }

    int rows = matrix.size();
    int cols = matrix[0].size();
    int left = 0, right = rows * cols - 1;

    // Binary search in the flattened view of the matrix
    while (left <= right) {
        int mid = left + (right - left) / 2;
        int midValue = matrix[mid / cols][mid % cols];

        if (midValue == target) {
            return true;
        } else if (midValue < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }

    return false;
}

int main() {
    vector<vector<int>> matrix = {
        {1, 3, 5, 7},
        {10, 11, 16, 20},
        {23, 30, 34, 50}
    };
    int target = 47;

    if (searchMatrix(matrix, target)) {
        cout << "Element found!" << endl;
    } else {
        cout << "Element not found." << endl;
    }

    return 0;
}
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    SEARCH ERROR    SPELL CHECKER

Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\class>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\" && g++ day5ques2.cpp -o day5ques2 && "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\"day5ques2
Element not found.

c:\Users\Gaurav Kumar\OneDrive\Desktop\class>
```

Find First and Last Position of Element in Sorted Array.

```cpp
#include <iostream>
#include <vector>
using namespace std;

int findPosition(const vector<int>& nums, int target, bool findFirst) {
    int left = 0, right = nums.size() - 1;
    int result = -1;

    while (left <= right) {
        int mid = left + (right - left) / 2;
        if (nums[mid] == target) {
            result = mid;
            if (findFirst) {
                right = mid - 1;
            } else {
                left = mid + 1;
            }
        } else if (nums[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return result;
}

vector<int> findFirstAndLastPosition(const vector<int>& nums, int target) {
    int firstPos = findPosition(nums, target, true);
    int lastPos = findPosition(nums, target, false);
    return {firstPos, lastPos};
}

int main() {
    vector<int> nums = {5, 7, 7, 8, 8, 10};
    int target = 8;

    vector<int> result = findFirstAndLastPosition(nums, target);
    cout << "First Position: " << result[0] << ", Last Position: " << result[1] << endl;

    return 0;
}
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   SEARCH ERROR   SPELL CHECKER

```
Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\class>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\" && g++ day5ques3.cpp -o day5ques3 && "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\"day5ques3
First Position: 3, Last Position: 4

c:\Users\Gaurav Kumar\OneDrive\Desktop\class>
```

Pair Sum Closet to 0.

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <climits>
using namespace std;

pair<int, int> closestPairSumToZero(vector<int>& arr) {
    if (arr.size() < 2) {
        return {0, 0};
    }

    sort(arr.begin(), arr.end());

    int left = 0, right = arr.size() - 1;
    int closestSum = INT_MAX;
    pair<int, int> closestPair;

    while (left < right) {
        int sum = arr[left] + arr[right];

        if (abs(sum) < abs(closestSum)) {
            closestSum = sum;
            closestPair = {arr[left], arr[right]};
        }
        if (sum < 0) {
            left++;
        } else {
            right--;
        }
    }

    return closestPair;
}

int main() {
    vector<int> arr = {1, 60, -10, 70, -80, 85};
    pair<int, int> result = closestPairSumToZero(arr);

    cout << "The pair closest to zero sum is: (" << result.first << ", " << result.second << ")" << endl;
    return 0;
}
```

Merge k Sorted Lists.

```cpp
#include <iostream>
#include <vector>
#include <queue>
using namespace std;

struct ListNode {
    int val;
    ListNode* next;
    ListNode(int x) : val(x), next(nullptr) {}
};

struct compare {
    bool operator()(ListNode* a, ListNode* b) {
        return a->val > b->val;
    }
};

ListNode* mergeKLists(vector<ListNode*>& lists) {
    priority_queue<ListNode*, vector<ListNode*>, compare> minHeap;

    for (auto list : lists) {
        if (list) {
            minHeap.push(list);
        }
    }

    ListNode* dummy = new ListNode(0);
    ListNode* tail = dummy;

    while (!minHeap.empty()) {
        ListNode* smallest = minHeap.top();
        minHeap.pop();

        tail->next = smallest;
        tail = tail->next;

        if (smallest->next) {
            minHeap.push(smallest->next);
        }
    }

    ListNode* result = dummy->next;
    delete dummy;
    return result;
}

void printList(ListNode* head) {
    while (head) {
        cout << head->val << " -> ";
        head = head->next;
    }
    cout << "NULL" << endl;
}
```

```cpp
int main() {
    ListNode* list1 = new ListNode(1);
    list1->next = new ListNode(4);
    list1->next->next = new ListNode(5);

    ListNode* list2 = new ListNode(1);
    list2->next = new ListNode(3);
    list2->next->next = new ListNode(4);

    ListNode* list3 = new ListNode(2);
    list3->next = new ListNode(6);

    vector<ListNode*> lists = {list1, list2, list3};

    ListNode* result = mergeKLists(lists);

    cout << "Merged list: ";
    printList(result);

    return 0;
}
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   SEARCH ERROR   SPELL CHECKER                                          ⌷ Code

Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\class>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\" && g++ day5ques5.cpp -o day5ques5 && "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\"day5ques5
Merged list: 1 -> 1 -> 2 -> 3 -> 4 -> 4 -> 5 -> 6 -> NULL

c:\Users\Gaurav Kumar\OneDrive\Desktop\class>
```

median of two sorted arrays.

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <climits>
using namespace std;

double findMedianSortedArrays(const vector<int>& nums1, const vector<int>& nums2) {
    vector<int> merged(nums1.size() + nums2.size());
    merge(nums1.begin(), nums1.end(), nums2.begin(), nums2.end(), merged.begin());

    int n = merged.size();
    if (n % 2 == 0) {
        return (merged[n / 2 - 1] + merged[n / 2]) / 2.0;
    } else {
        return merged[n / 2];
    }
}

int main() {
    int n1, n2;
    cout << "Enter the number of elements in the first array: ";
    cin >> n1;

    vector<int> nums1(n1);
    cout << "Enter the elements of the first sorted array: ";
    for (int i = 0; i < n1; i++) {
        cin >> nums1[i];
    }

    cout << "Enter the number of elements in the second array: ";
    cin >> n2;

    vector<int> nums2(n2);
    cout << "Enter the elements of the second sorted array: ";
    for (int i = 0; i < n2; i++) {
        cin >> nums2[i];
    }

    double median = findMedianSortedArrays(nums1, nums2);
    cout << "Median: " << median << endl;

    return 0;
}
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    SEARCH ERROR    SPELL CHECKER

Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\class>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\" && g++ day5ques6.cpp -o day5ques6 && "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\"day5ques6
Enter the number of elements in the first array: 2
Enter the elements of the first sorted array: 1 2
Enter the number of elements in the second array: 1
Enter the elements of the second sorted array: 3
Median: 2

c:\Users\Gaurav Kumar\OneDrive\Desktop\class>
```

searching a number

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main() {
    int n, target;

    cout << "Enter the number of elements in the array: ";
    cin >> n;

    vector<int> arr(n);
    cout << "Enter the elements of the array: ";
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    cout << "Enter the number to search: ";
    cin >> target;

    bool found = false;
    for (int i = 0; i < n; i++) {
        if (arr[i] == target) {
            cout << "Number " << target << " found at position " << i + 1 << "." << endl;
            found = true;
            break;
        }
    }

    if (!found) {
        cout << "Number " << target << " not found in the array." << endl;
    }

    return 0;
}
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   SEARCH ERROR   SPELL CHECKER

Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\class>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\" && g++ day5ques7.cpp -o day5ques7 && "c:\Users\Gaurav Kumar\OneDrive\Desktop\class\"day5ques7
Enter the number of elements in the array: 5
Enter the elements of the array: 1 2 3 4 5
Enter the number to search: 3
Number 3 found at position 3.

c:\Users\Gaurav Kumar\OneDrive\Desktop\class>
```