# Finding Common Elements in Three Sorted Arrays

# Finding Common Elements in Three Sorted Arrays

---

There are three sorted arrays namely, A, B, C ; each having elements n1, n2, n3 respectively. Find the common elements in the arrays.
For eg.:
A[ ] = { 9, 21, 30, 70, 80, 99, 100, 120, 150 }
B[ ] = { 0, 2, 4, 21, 70, 100, 150 }
C[ ] = { 2, 14, 17, 21, 30, 50, 70, 120 }
Output: 21, 70

## 1.

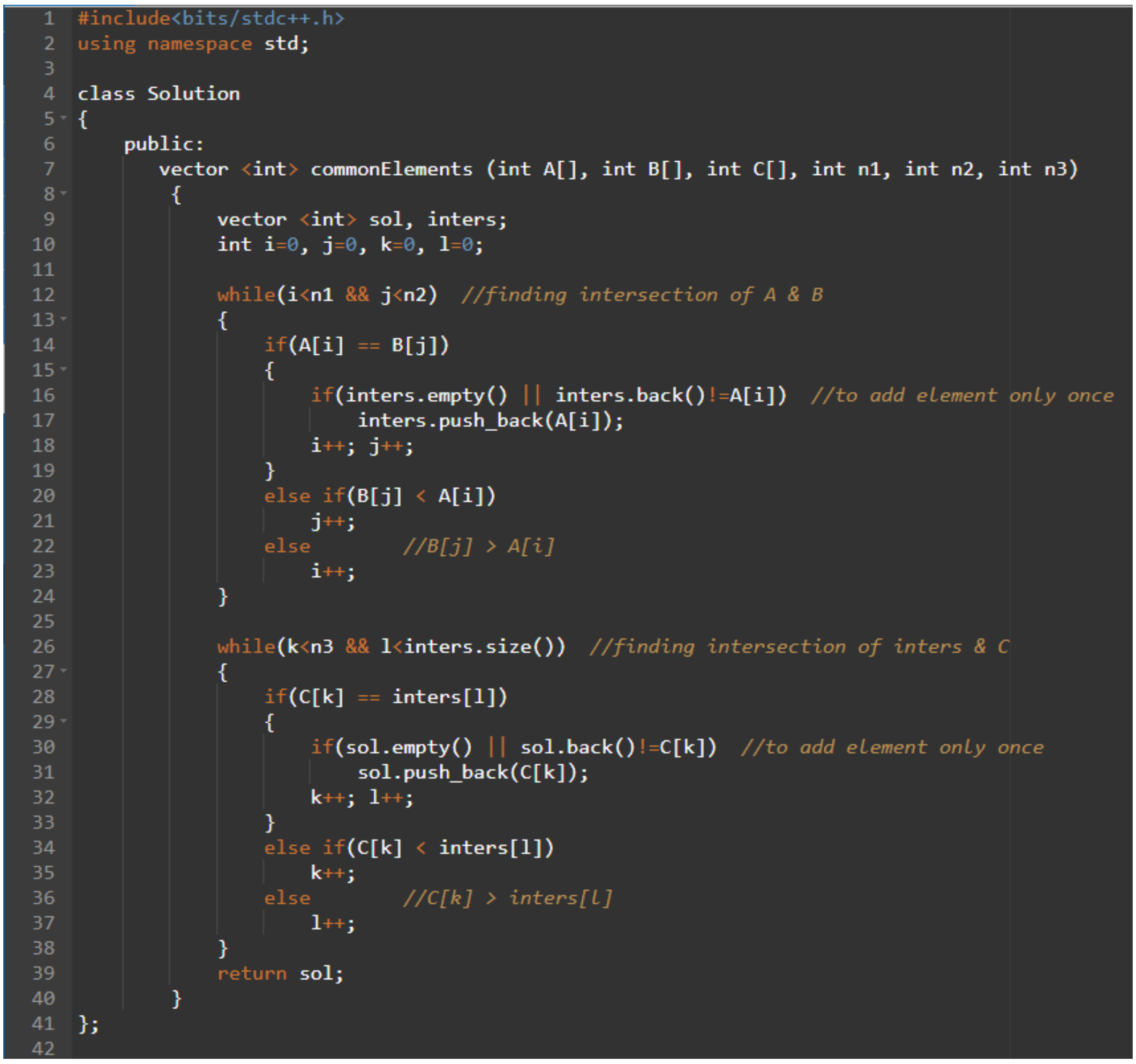
The brute force approach is by creating an intersection of the first two arrays (common elements)
And then finding its intersection with the third array. This way we can find the common elements in all the three arrays .
In the above example:
intersection of A and B will be { 21, 70, 100, 150 }
And the solution will be intersection of this and C i.e. { 21, 70 }

The code for the above approach would work as follows :

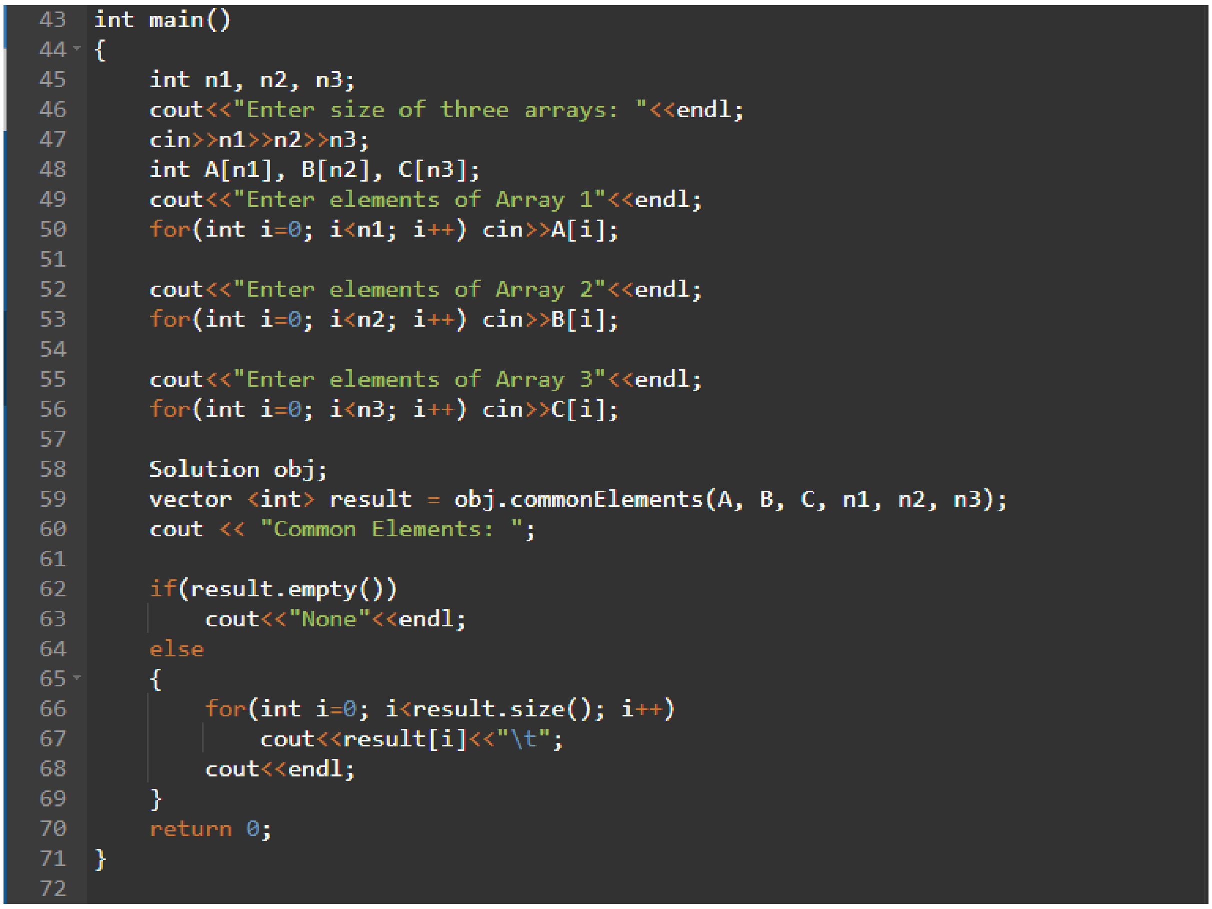
- Finding intersection of array A and B:
    - Initialise variables i, j at 0
    - Create a vector named inters to store the common elements of the 2 arrays
    - Create a while loop that will run till we reach the end of one of the arrays i.e. (i < n1 && j < n2)
    - Inside the loop : (let A[i] be a; and B[i] be b )

        - If a and b are equal push any one of them to the inters vector and increment i and j
        - If a > b ; b can not be the common element so increment j
        - Else (i.e. if a < b ) ; a cannot be the common element so increment i
- Find the intersection of the inters vector and C array
- Store it in the solution vector and return it.

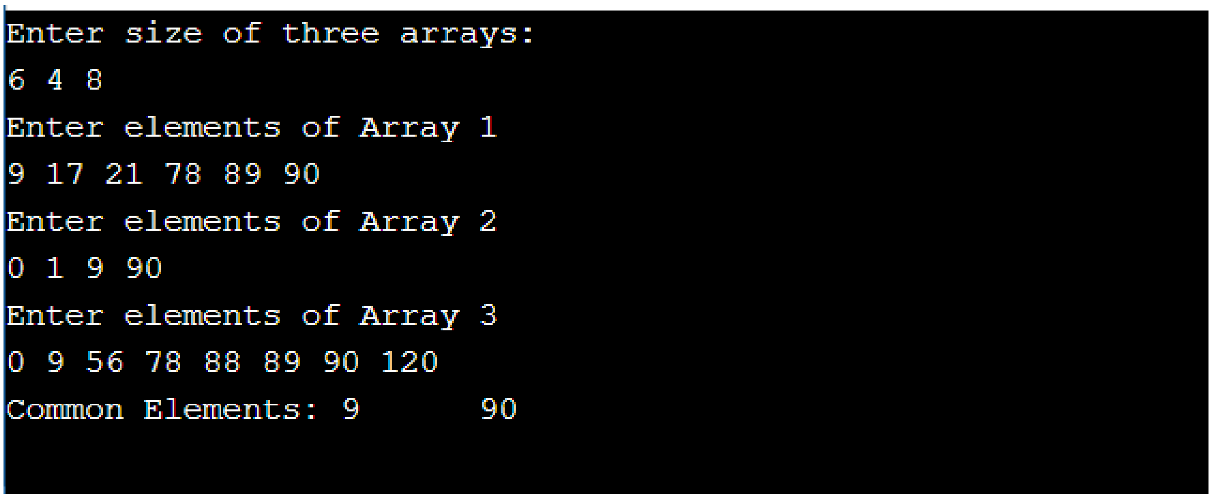
The time complexity is O(n1 + n2 + n3)
And Space complexity is O( min(n1, n2) + min(n1, n2, n3) )



```cpp
#include<bits/stdc++.h>
using namespace std;

class Solution
{
    public:
       vector <int> commonElements (int A[], int B[], int C[], int n1, int n2, int n3)
        {
            vector <int> sol, inters;
            int i=0, j=0, k=0, l=0;

            while(i<n1 && j<n2)  //finding intersection of A & B
            {
                if(A[i] == B[j])
                {
                    if(inters.empty() || inters.back()!=A[i])  //to add element only once
                        inters.push_back(A[i]);
                    i++; j++;
                }
                else if(B[j] < A[i])
                    j++;
                else        //B[j] > A[i]
                    i++;
            }

            while(k<n3 && l<inters.size())  //finding intersection of inters & C
            {
                if(C[k] == inters[l])
                {
                    if(sol.empty() || sol.back()!=C[k])  //to add element only once
                        sol.push_back(C[k]);
                    k++; l++;
                }
                else if(C[k] < inters[l])
                    k++;
                else        //C[k] > inters[l]
                    l++;
            }
            return sol;
        }
};
```



```cpp
int main()
{
    int n1, n2, n3;
    cout<<"Enter size of three arrays: "<<endl;
    cin>>n1>>n2>>n3;
    int A[n1], B[n2], C[n3];
    cout<<"Enter elements of Array 1"<<endl;
    for(int i=0; i<n1; i++) cin>>A[i];

    cout<<"Enter elements of Array 2"<<endl;
    for(int i=0; i<n2; i++) cin>>B[i];

    cout<<"Enter elements of Array 3"<<endl;
    for(int i=0; i<n3; i++) cin>>C[i];

    Solution obj;
    vector <int> result = obj.commonElements(A, B, C, n1, n2, n3);
    cout << "Common Elements: ";

    if(result.empty())
        cout<<"None"<<endl;
    else
    {
        for(int i=0; i<result.size(); i++)
            cout<<result[i]<<"\t";
        cout<<endl;
    }
    return 0;
}

```



```
Enter size of three arrays:
6 4 8
Enter elements of Array 1
9 17 21 78 89 90
Enter elements of Array 2
0 1 9 90
Enter elements of Array 3
0 9 56 78 88 89 90 120
Common Elements: 9      90
```

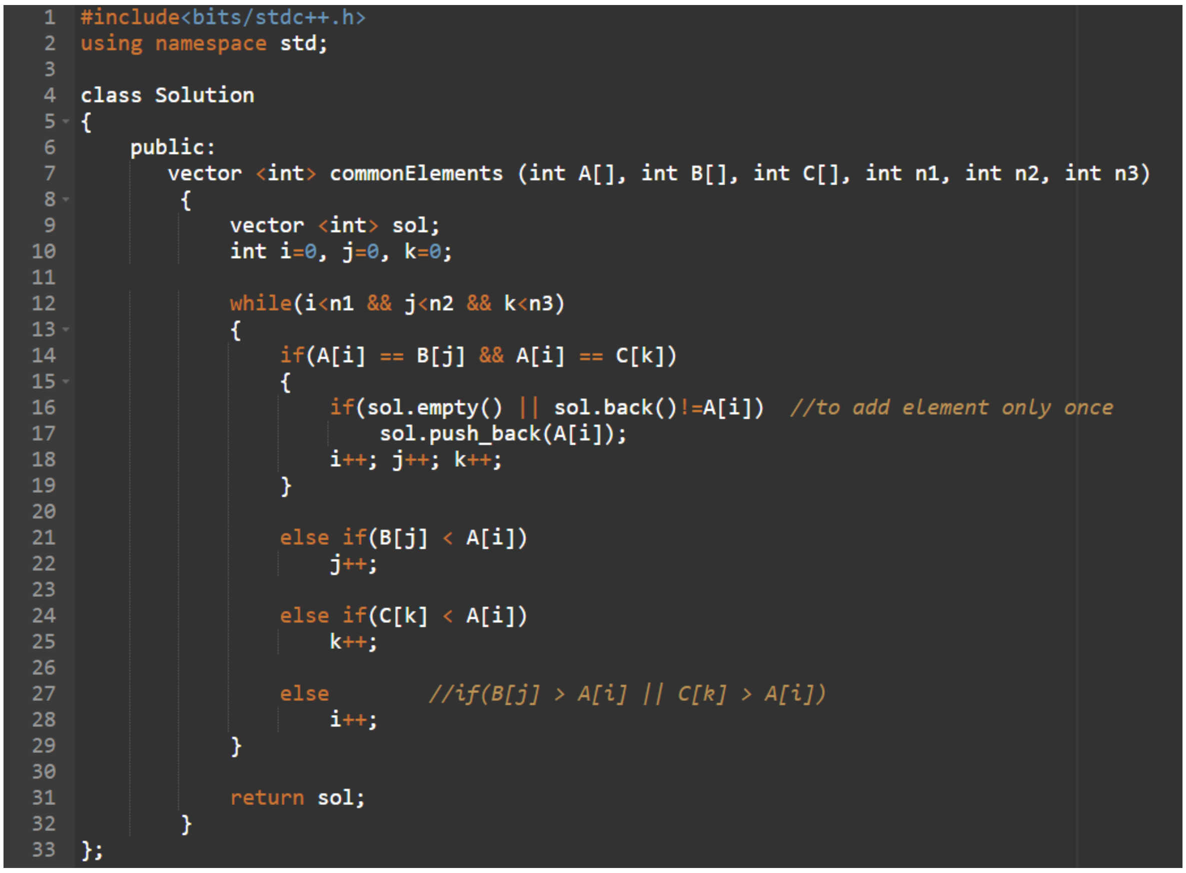
## 2.

The above approach uses two loops and extra space (the inters vector) . Instead of traversing two arrays at a time, we can traverse 3 arrays at the same time and without using the inters vector.

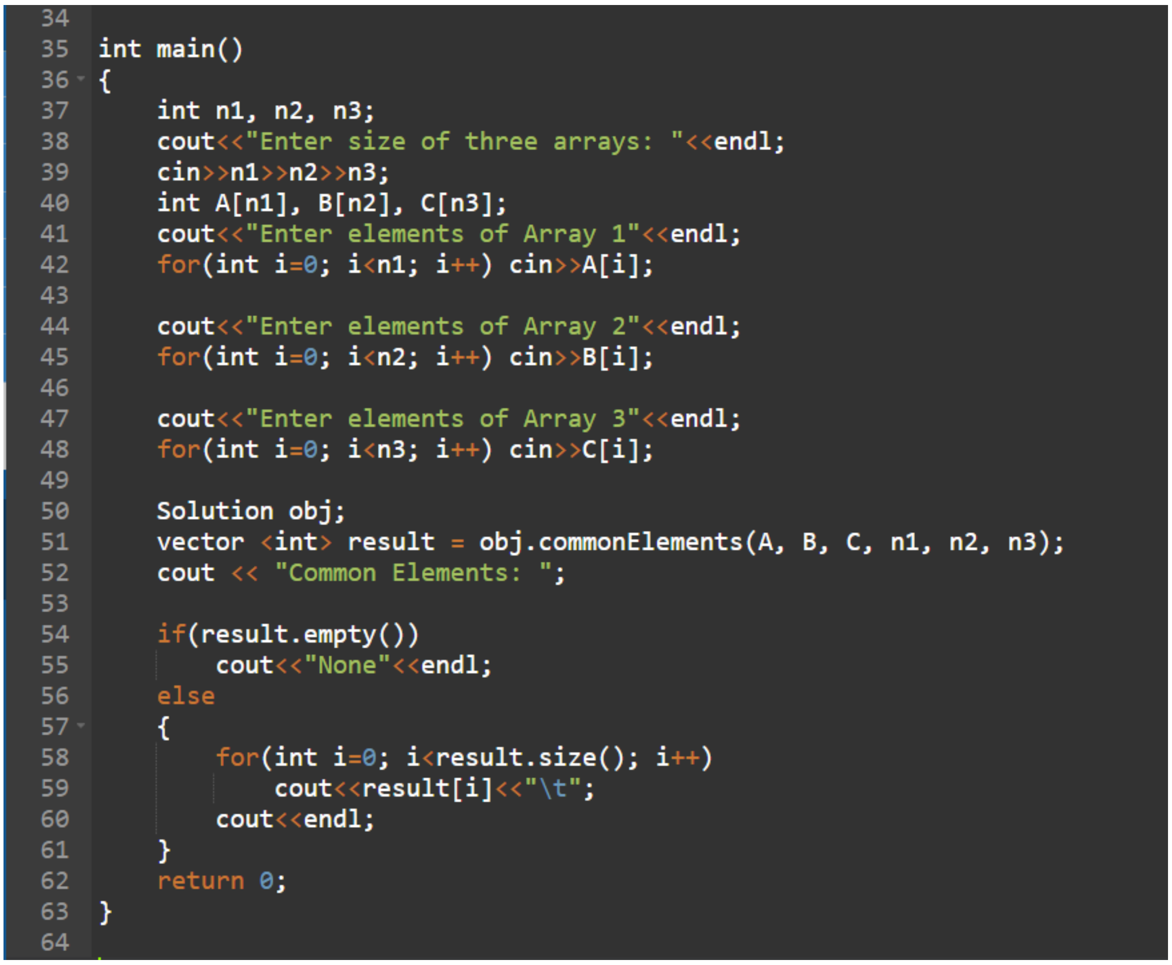
The code for this approach will work as follows :

- I will initiate variable i, j, k at 0.
  i = 0, j = 0, k = 0
- Create a vector named solution to store the common elements.
- I will create a while loop that will run till we reach the end of any one of the arrays i.e.
  ( i < n1 && j < n2 && k < n3 )
- Inside the loop we will consider the following:
  (let A[i] be a; B[j] be b; C[k] be c)
  - If a, b, c are equal : add any one of these to the solution vector
  - If a > b : then b cannot be the common element so increment j
  - If a > c : then c cannot be the common element so increment k
  - Else ( a < b or a < c) : then a cannot be the common element so increment i
- Return solution vector.
  (if the vector is empty, no common element is present)

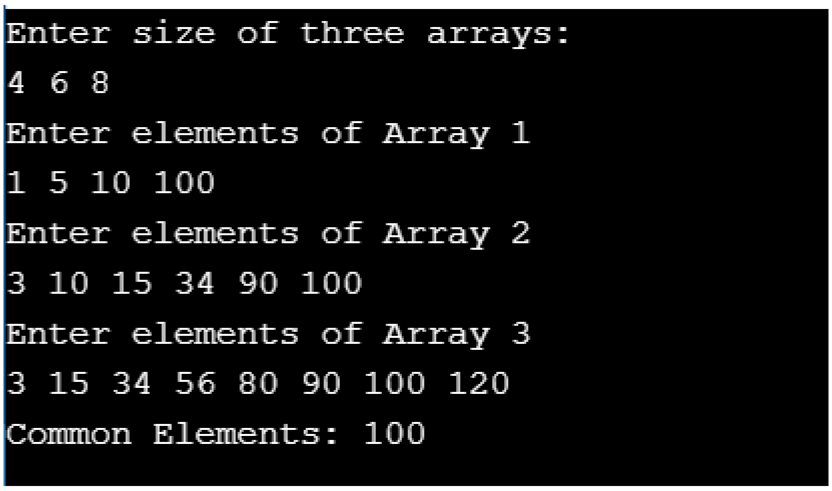
The time complexity is O( n1 + n2 + n3 )
And space complexity is O( min(n1, n2, n3) )



```cpp
#include<bits/stdc++.h>
using namespace std;

class Solution
{
    public:
       vector <int> commonElements (int A[], int B[], int C[], int n1, int n2, int n3)
        {
            vector <int> sol;
            int i=0, j=0, k=0;

            while(i<n1 && j<n2 && k<n3)
            {
                if(A[i] == B[j] && A[i] == C[k])
                {
                    if(sol.empty() || sol.back()!=A[i])  //to add element only once
                        sol.push_back(A[i]);
                    i++; j++; k++;
                }

                else if(B[j] < A[i])
                    j++;

                else if(C[k] < A[i])
                    k++;

                else        //if(B[j] > A[i] || C[k] > A[i])
                    i++;
            }

            return sol;
        }
};
```



```cpp

int main()
{
    int n1, n2, n3;
    cout<<"Enter size of three arrays: "<<endl;
    cin>>n1>>n2>>n3;
    int A[n1], B[n2], C[n3];
    cout<<"Enter elements of Array 1"<<endl;
    for(int i=0; i<n1; i++) cin>>A[i];

    cout<<"Enter elements of Array 2"<<endl;
    for(int i=0; i<n2; i++) cin>>B[i];

    cout<<"Enter elements of Array 3"<<endl;
    for(int i=0; i<n3; i++) cin>>C[i];

    Solution obj;
    vector <int> result = obj.commonElements(A, B, C, n1, n2, n3);
    cout << "Common Elements: ";

    if(result.empty())
        cout<<"None"<<endl;
    else
    {
        for(int i=0; i<result.size(); i++)
            cout<<result[i]<<"\t";
        cout<<endl;
    }
    return 0;
}

```



```
Enter size of three arrays:
4 6 8
Enter elements of Array 1
1 5 10 100
Enter elements of Array 2
3 10 15 34 90 100
Enter elements of Array 3
3 15 34 56 80 90 100 120
Common Elements: 100
```