TIME AND SPACE COMPLEXITY

7 Time and Space Complexity 68% Completed

Bubble Sort

Easy

1. You are given an array(arr) of integers.
2. You have to sort the given array in increasing order using bubble sort.

Constraints

```
1 <= N <= 10000
-10^9 <= arr[i] <= 10^9
```

Format

Input

An Integer n
arr1
arr2..
n integers

Output

Check the sample ouput and question video.

Example

Sample Input

```
5
7
-2
4
1
3
```

Sample Output

```
Comparing -2 and 7
Swapping -2 and 7
Comparing 4 and 7
Swapping 4 and 7
Comparing 1 and 7
Swapping 1 and 7
Comparing 3 and 7
Swapping 3 and 7
Comparing 4 and -2
Comparing 1 and 4
Swapping 1 and 4
Comparing 3 and 4
Swapping 3 and 4
Comparing 1 and -2
Comparing 3 and 1
Comparing 1 and -2
-2
1
3
4
7
```

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


bool isSmaller(int arr[],int i,int j ){
    cout<<"Comparing "<<arr[i]<<" and "<<arr[j]<<endl;
    if(arr[i]<arr[j]){
        return true;
    }else{
        return false;
    }
}

void swap(int arr[],int i,int j){
    cout<<"Swapping "<<arr[i]<<" and "<<arr[j]<<endl;
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;

}


void bubbleSort(int arr[],int n){
    // write your code here
    for (int itr {1}; itr <= n-1; itr++){
        for (int i{}; i < n-itr; i++){
            if(isSmaller(arr,i+1,i)){
                swap (arr ,i+1,i);
            }
        }
    }
}

void print(int arr[],int n){
    for(int i=0;i<n;i++){
        cout<<arr[i]<<endl;
    }
}
```

```
int main(){
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }

    bubbleSort(arr,n);
    print(arr,n);

}
```

Selection Sort

Easy

1. You are given an array(arr) of integers.
2. You have to sort the given array in increasing order using selection sort.

Constraints

```
1 <= N <= 10000
-10^9 <= arr[i] <= 10^9
```

Format

Input

An Integer n
arr1
arr2..
n integers

Output

Check the sample ouput and question video.

Example

Sample Input

```
5
7
-2
4
1
3
```

Sample Output

```
Comparing -2 and 7
Comparing 4 and -2
Comparing 1 and -2
Comparing 3 and -2
Swapping 7 and -2
Comparing 4 and 7
Comparing 1 and 4
Comparing 3 and 1
Swapping 7 and 1
Comparing 7 and 4
Comparing 3 and 4
Swapping 4 and 3
```

```
Comparing 4 and 7
Swapping 7 and 4
-2
1
3
4
7
```

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


bool isSmaller(int arr[],int i,int j ){
    cout<<"Comparing "<<arr[i]<<" and "<<arr[j]<<endl;
    if(arr[i]<arr[j]){
        return true;
    }else{
        return false;
    }
}

void swap(int arr[],int i,int j){
    cout<<"Swapping "<<arr[i]<<" and "<<arr[j]<<endl;
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;

}


void selectionSort(int arr[],int n){
    //write your code
    for (int itr{0}; itr < n-1; itr++){
        int min_i = itr;
        for (int i{itr+1}; i < n ;i++){
            if (isSmaller(arr, i,min_i)){
                min_i = i;
            }
        }
        swap(arr,itr,min_i);

    }

}

void print(int arr[],int n){
    for(int i=0;i<n;i++){
        cout<<arr[i]<<endl;
    }
}
```

```
int main(){
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }

    selectionSort(arr,n);
    print(arr,n);

}
```

Insertion Sort

Easy

1. You are given an array(arr) of integers.
2. You have to sort the given array in increasing order using insertion sort.

Constraints

```
1 <= N <= 10000
-10^9 <= arr[i] <= 10^9
```

Format

Input

An Integer n
arr1
arr2..
n integers

Output

Check the sample output and question video.

Example

Sample Input

```
5
7
-2
4
1
3
```

Sample Output

```
Comparing -2 and 7
Swapping 7 and -2
Comparing 4 and 7
Swapping 7 and 4
Comparing 4 and -2
Comparing 1 and 7
Swapping 7 and 1
Comparing 1 and 4
Swapping 4 and 1
Comparing 1 and -2
Comparing 3 and 7
Swapping 7 and 3
Comparing 3 and 4
Swapping 4 and 3
Comparing 3 and 1
-2
1
3
4
7
```

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


bool isGreater(int arr[],int j,int i ){
    cout<<"Comparing "<<arr[i]<<" and "<<arr[j]<<endl;
    if(arr[i]<arr[j]){
        return true;
    }else{
        return false;
    }
}

void swap(int arr[],int i,int j){
    cout<<"Swapping "<<arr[i]<<" and "<<arr[j]<<endl;
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;

}


void insertionSort(int arr[],int n){

  // write your code
    for (int it {1}; it < n; it++){
        for (int j {it}; j > 0 ; j--){
            if (isGreater(arr , j-1, j)){
                swap (arr, j-1,j);
            }else{
                break;
            }
        }
    }
}

void print(int arr[],int n){
    for(int i=0;i<n;i++){
        cout<<arr[i]<<endl;
    }
}
```

```
int main(){
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }

    insertionSort(arr,n);
    print(arr,n);

}
```

Merge Two Sorted Arrays

Easy

1. You are given two sorted arrays(a,b) of integers.
2. You have to merge them and form one sorted array.
3. You have to do it in linear time complexity.

Constraints

```
1 <= N <= 10^8
-10^9 <= a[i],b[i] <= 10^9
```

Format

Input

An Integer n
a1
a2..n integers
An integer m
b1
b2..m integers

Output

Check the sample output and question video.

Example

Sample Input

```
4
-2
5
9
11
3
4
6
8
```

Sample Output

```
-2
4
5
6
8
9
11
```

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

vector<int> mergeTwoSortedArrays(vector<int> &A, vector<int> &B)
{   vector <int> result;

    int a = A.size();
    int b = B.size();

    int ai{};
    int bi{};
    while (ai<a && bi<b){
        if (A[ai]<B[bi]){
            result.push_back(A[ai]);
            ai++;
        }else{
            result.push_back(B[bi]);
            bi++;
        }
    }
    for( int i {bi};i < b;i++){
        result.push_back(B[i]);
    }
    for( int i {ai};i<a;i++){
        result.push_back(A[i]);
    }

    //my solution

    // while (true){
    //     if (A[ai]<B[bi]){
    //         result.push_back(A[ai]);
    //         ai++;
    //         if(ai == a){
    //             for( int i {bi};i < b;i++){
    //                 result.push_back(B[i]);
    //             }
    //             return result;
    //         }

    //     }else{
    //         result.push_back(B[bi]);
    //         bi++;
    //         if(bi == b){
    //             for( int i {ai};i<a;i++){
    //                 result.push_back(A[i]);
    //             }
    //             return result;
    //         }

    //     }
    // }

    return result;

}

void input(vector<int> &arr)
{
    for (int i = 0; i < arr.size(); i++)
    {
        cin >> arr[i];
    }
}

void output(vector<int> &arr)
{
    for (int i = 0; i < arr.size(); i++)
    {
        cout << arr[i] << endl;
    }
}

int main()
{
    int n, m;
    cin >> n;
    vector<int> A(n, 0);
    input(A);

    cin >> m;
    vector<int> B(m, 0);
    input(B);

    vector<int> ans = mergeTwoSortedArrays(A, B);
    output(ans);
    return 0;
}
```

Merge Sort
Easy

1. You are given an array(arr) of integers.
2. You have to sort the given array in increasing order using the merge sort.

Constraints

```
1 <= N <= 100000
-10^9 <= arr[i] <= 10^9
```

Format

Input

An Integer n
arr1
arr2..
n integers

Output

Check the sample output and question video.

Example

Sample Input

```
5
7
-2
4
1
3
```

Sample Output

```
Merging these two arrays
left array -> 7
right array -> -2
Merging these two arrays
left array -> -2 7
right array -> 4
Merging these two arrays
left array -> 1
right array -> 3
Merging these two arrays
left array -> -2 4 7
right array -> 1 3
Sorted Array -> -2 1 3 4 7
```

```
#include <iostream>

#include <vector>

using namespace std;
```

```cpp
void input(vector<int> &arr)
{
    for (int i = 0; i < arr.size(); i++)
    {
        cin >> arr[i];
    }
}

void print(vector<int> &arr)
{
    for (int i = 0; i < arr.size(); i++)
    {
        cout << arr[i] << " ";
    }
    cout << endl;
}

vector<int> mergeTwoSortedArrays(vector<int> &A, vector<int> &B)
{
    if (A.size() == 0 || B.size() == 0)
        return A.size() == 0 ? B : A;

    int n = A.size();
    int m = B.size();
    vector<int> ans(n + m, 0);

    cout << ("Merging these two arrays ") << endl;
    cout << ("left array -> ");
    print(A);
    cout << ("right array -> ");
    print(B);

    int i = 0, j = 0, k = 0;
    while (i < n && j < m)
    {
        if (A[i] < B[j])
            ans[k++] = A[i++];
        else
            ans[k++] = B[j++];
    }

    while (i < n)
        ans[k++] = A[i++];
    while (j < m)
        ans[k++] = B[j++];

    return ans;
}

vector<int> mergeSort(vector<int> &arr ,int si ,int ei)
{
    if(si == ei){
        vector <int> r ;
        r.push_back(arr[si]);
        return r;
    }
    int mid = (si+ei)/2;
    vector<int> A = mergeSort(arr, si,mid);
    vector<int> B = mergeSort(arr,mid+1,ei);
    return mergeTwoSortedArrays(A,B);

}

int main()
{
    int n;
    cin >> n;
    vector<int> A(n, 0);
    input(A);

    vector<int> ans = mergeSort(A ,0,n-1);
    cout << "Sorted Array -> ";
    print(ans);
    return 0;
}
```

Partition An Array

Easy

1. You are given an array(arr) of integers and a pivot.
2. You have to re-arrange the given array in such a way that all elements smaller or equal to pivot lie on the left side of pivot and all elements greater than pivot lie on its right side.
3. You have to achieve this in linear time.

Note -> For more information, watch question video.

Constraints

```
1 <= N <= 100000
-10^9 <= arr[i] <= 10^9
-10^9 <= pivot <= 10^9
```

Format

Input

An Integer n
arr1
arr2..
n integers
An integer pivot

Output

Check the sample output and question video.

Example

Sample Input

```
5
7
-2
4
1
3
3
```

Sample Output

```
Swapping -2 and 7
Swapping 1 and 7
Swapping 3 and 4
-2 1 3 7 4
```

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


void swap(int arr[],int i,int j){
    cout<<"Swapping "<<arr[i]<<" and "<<arr[j]<<endl;
    int temp=arr[i];
    arr[i]=arr[j];
    arr[j]=temp;
}

void partition(int arr[],int n,int pivot){
    // write your code

    int c = 0;
    int b = 0;

    while(c != n){
        if (arr[c] <= pivot ){
            swap (arr , c, b);
            c++;
            b++;
        }else
        {
            c++;
        }
    }
}

void print(int arr[],int n){
    for(int i=0;i<n;i++){
        cout<<arr[i]<<" ";
    }
    cout<<endl;
}
```

```
int main(){

    int n;
    cin>>n;

    int arr[n];
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }

    int pivot;
    cin>>pivot;

    partition(arr,n,pivot);
    print(arr,n);
}
```

Quick Sort

Easy

1. You are given an array(arr) of integers.
2. You have to sort the given array in increasing order using quick-sort.

Constraints

```
1 <= N <= 100000
-10^9 <= arr[i] <= 10^9
```

Format

Input

An Integer n
arr1
arr2..
n integers

Output

Check the sample output and question video.

Example

Sample Input

```
5
7
-2
4
1
3
```

Sample Output

```
pivot -> 3
Swapping -2 and 7
Swapping 1 and 7
Swapping 3 and 4
pivot index -> 2
pivot -> 1
Swapping -2 and -2
Swapping 1 and 1
```

```
pivot index -> 1
pivot -> -2
Swapping -2 and -2
pivot index -> 0
pivot -> 4
Swapping 4 and 7
pivot index -> 3
pivot -> 7
Swapping 7 and 7
pivot index -> 4
-2 1 3 4 7
```

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

void swap(vector<int> &arr, int i, int j){
    cout<<"Swapping " << arr[i] << " and " << arr[j] << endl;
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}

int partition(vector<int> &arr, int pivot, int lo, int hi){
    cout << "pivot -> " << pivot << endl;
    int curr = lo;
    int prev = lo - 1;
    while(curr <= hi){
        if(arr[curr] <= pivot){
            prev++;
            swap(arr, curr, prev);
        }
        curr++;
    }
    cout << "pivot index -> " << prev << endl;
    return prev;
}

void quicksort(vector<int> &arr, int lo, int hi){
    // write your code here
    if (lo>hi){     //lo>=hi is better here but did lo>hi to match
with ans
        return ;
    }

    int pivot = arr[hi];
    int p_index = partition(arr, pivot, lo ,hi);

    quicksort(arr ,lo ,p_index-1 );
    quicksort(arr ,p_index+1,hi );

    return ;
}
```

```
void Display(vector<int>& arr){
    for(int ele : arr){
        cout<< ele << " ";
    }
}

int main(){
    int n;
    cin >> n;
    vector<int> arr(n, 0);
    for(int i = 0; i < arr.size(); i++){
        cin >> arr[i];
    }
    quicksort(arr, 0, n - 1);
    Display(arr);
    return 0;
}
```

Quick Select

Easy

1. You are given an array(arr) of integers.
2. You have to find the k-th smallest element in the given array using the quick-select algorithm.

Constraints

```
1 <= N <= 100000
-10^9 <= arr[i] <= 10^9
1 <= k <= N
```

Format

Input

An Integer n
arr1
arr2..
n integers
An integer k

Output

Check the sample output and question video.

Example

Sample Input

```
5
7
-2
4
1
3
3
```

Sample Output

```
pivot -> 3
Swapping -2 and 7
Swapping 1 and 7
Swapping 3 and 4
pivot index -> 2
3
```

```
#include<iostream>
#include<vector>

using namespace std;


void swap(vector<int> &arr, int i, int j){
    cout<<"Swapping " << arr[i] << " and " << arr[j] << endl;
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}

int partition(vector<int> &arr, int pivot, int lo, int hi){
    cout << "pivot -> " << pivot << endl;
    int curr = lo;
    int prev = lo - 1;
    while(curr <= hi){
        if(arr[curr] <= pivot){
            prev++;
            swap(arr, curr, prev);
        }
        curr++;
    }
    cout << "pivot index -> " << prev << endl;
    return prev;
}


int quickselect(vector<int>& arr, int lo, int hi, int k) {
    // write your code here
    int pivot = arr[hi];
    int pi = partition(arr ,pivot, lo ,hi);
    int ans {};
    if(pi == k){
        ans = arr[k];
    }else if (pi > k){
        ans = quickselect(arr , lo , pi-1 ,k);
    }else if (pi < k){
        ans = quickselect(arr , pi+1 ,hi ,k);
    }
    return ans;
}
```

```
int main() {
  int n;
  cin >> n;
  vector<int> arr(n, 0);
  for (int i = 0; i < n; i++) {
    cin >> arr[i];
  }
  int k;
  cin >> k;

  int ans = quickselect(arr, 0, n - 1, k-1);
  cout << ans << endl;
  return 0;
}
```

Count Sort

Easy

1. You are given an array(arr) of integers.
2. You have to sort the given array in increasing order using count sort.

Constraints

```
1 <= N <= 10000
-10^8 <= arr[i] <= 10^8
```

Format

Input

An Integer n
arr1
arr2..
n integers

Output

Check the sample ouput and question video.

Example

Sample Input

```
5
7
-2
4
1
3
```

Sample Output

```
-2
1
3
4
7
```

```cpp
#include<iostream>
#include<vector>
#include<climits>
#include<algorithm>

using namespace std;

void Display(vector<int>& arr){
    for(int ele : arr){
        cout<< ele << endl;
    }
}

void countsort(vector<int> &arr, int max, int min){
    // write your code here

    int range  = max - min + 1 ;  //size of frequency array
    vector <int> f(range );

    for (auto a:arr){
        int i = a-min;       //filling frequency array
        f[i]++;
    }

    f[0] = f[0] - 1;      //making in form of index first ele
                         //other will get convert while making
prefix sum array
    int n = arr.size();
    for (int i{1} ; i < range ; i++ ) {         //making frequency
arryay prefix sum array
        f[i] = f[i] + f[i - 1] ;
    }
    vector <int> ans(n);

    for(int i{n-1}; i>=0 ; i--) {
        int index = f[arr[i]-min];
        ans[index] = arr[i];
        f[arr[i]-min]--;
    }
    for (int i{}; i<n ;i++){
        arr[i] = ans[i];
    }
    return;

}
```

```
int main(){
    int n;
    cin >> n;
    vector<int> arr(n, 0);

    for(int i = 0; i < n; i++){
        cin >> arr[i];
    }

    int maxi = (int)-1e9;
    int mini = (int)1e9;

    for(int i = 0; i < n; i++){
        mini = min(mini, arr[i]);
        maxi = max(maxi, arr[i]);
    }

    countsort(arr, maxi, mini);
    Display(arr);

}
```

Radix Sort

Easy

1. You are given an array(arr) of integers.
2. You have to sort the given array in increasing order using radix sort.

Constraints

```
1 <= N <= 10000
0 <= arr[i] <= 10^8
```

Format

Input

An Integer n
arr1
arr2..
n integers

Output

Check the sample ouput and question video.

Example

Sample Input

```
5
7
2
4
1
3
```

Sample Output

```
After sorting on 1 place -> 1 2 3 4 7
1 2 3 4 7
```

```
#include<iostream>
#include<vector>
#include<climits>
#include<algorithm>

using namespace std;

void Display(vector<int>& arr){
    for(int ele : arr){
        cout<< ele << " ";
    }
}

void countsort(vector<int> &arr, int d){
    // write your code here
    int n = arr.size();


    vector <int> f(10);

    for (auto a:arr){
        int i = ((a/d)%10);        //filling frequency array
        f[i]++;
    }

    f[0] = f[0] - 1;

    for (int i{1} ; i < 10 ; i++ ) {          //making frequency
arryay prefix sum array
        f[i] = f[i] + f[i - 1] ;
    }
    vector <int> ans(n);

    for(int i{n-1}; i>=0 ; i--) {
        int at = ((arr[i]/d)%10);
        int index = f[at];
        ans[index] = arr[i];
        f[at]--;
    }
    for (int i{}; i<n ;i++){
        arr[i] = ans[i];
    }

    cout<< "After sorting on " << d << " place -> ";
    Display(arr);
    cout << endl;
}
```

```
void radixSort(vector<int> &arr){
    // write your code here
    int max = arr[0];
    for(int i{} ; i< arr.size() ;i++){
        if(arr[i] > max) {
            max = arr[i];
        }
    }
    int d{1};
    while (max != 0){
        max /= 10;
        countsort(arr,d);
        d *= 10;
    }
}


int main(){
    int n;
    cin >> n;
    vector<int> arr(n, 0);

    for(int i = 0; i < n; i++){
        cin >> arr[i];
    }

    radixSort(arr);
    Display(arr);
    return 0;
}
```

Sort Dates

Easy

1. You are given an array(arr) of different dates in format DD-MM-YYYY.
2. You have to sort these dates in increasing order.

Constraints

```
1 <= N <= 10000
All dates are between year 0 to year 2500
```

Format

Input

An Integer N
arr1
arr2..
n integers

Output

Check the sample output and question video.

Example

Sample Input

```
5
12041996
20101996
05061997
12041989
11081987
```

Sample Output

```
11081987
12041989
12041996
20101996
05061997
```

```cpp
#include <iostream>

#include <vector>

using namespace std;

void input(vector<string> &arr)
{
    for (int i = 0; i < arr.size(); i++)
    {
        cin >> arr[i];
    }
}

void print(vector<string> &arr)
{
    for (int i = 0; i < arr.size(); i++)
    {
        cout << arr[i] << endl;
    }
    cout << endl;
}
```

```
void countSort(vector <string> &arr,int div, int mod, int range) {
    // write your code here
    int n = arr.size();

    vector <int> f(range );

    for (auto a:arr){
        int i = (stoi(a)/div) % mod ;       //filling frequency 
array
        f[i]++;
    }

    f[0] = f[0] - 1;
    for (int i{1} ; i < range ; i++ ) {
        f[i] = f[i] + f[i - 1] ;
    }
    vector <string> ans(n);

    for(int i{n-1}; i>=0 ; i--) {
        int b = (stoi(arr[i])/div) % mod;
        int index = f[b];
        ans[index] = arr[i];
        f[b]--;
    }
    for (int i{}; i<n ;i++){
        arr[i] = ans[i];
    }

    return;

}

void sort01(vector<string> &arr)
{
    countSort(arr,1000000,100,32);
    countSort(arr ,10000,100,13);
    countSort(arr ,1, 10000,2501);
    return;
}

int main()
{
    int n;
    cin >> n;
    vector<string> A(n, "");
    input(A);

    sort01(A);
    print(A);
    return 0;
}
```

Sort 01

Easy

1. You are given an array(arr) containing only 0's and 1's.
2. You have to sort the given array in increasing order and in linear time.

Constraints

```
1 <= N <= 10000
arr[i] = 0,1
```

Format

Input

An Integer N
arr1
arr2..
n integers

Output

Check the sample output and question video.

Example

Sample Input

```
5
0
1
0
1
0
```

Sample Output

```
Swapping index 0 and index 0
Swapping index 2 and index 1
Swapping index 4 and index 2
0
0
0
1
1
```

```
#include<iostream>
#include<vector>

using namespace std;


void swap(vector<int> &arr, int i, int j){
    cout<<"Swapping index " << i << " and index " << j << endl;
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}
```

```
void sort01(vector<int>& arr) {
    // write your code here
  int i{};
  int j{};
  while (i != arr.size()){
    if (arr[i] == 1){
      i++;
    }else{
      swap (arr ,i,j);
      i++;
      j++;
    }
  }

}


void Display(vector<int>& arr){
    for(int ele : arr){
        cout<< ele << endl;
    }
}

int main() {
  int n;
  cin >> n;
  vector<int> arr(n, 0);
  for (int i = 0; i < n; i++) {
    cin >> arr[i];
  }

  sort01(arr);
  Display(arr);
  return 0;
}
```

Sort 012

Easy

1. You are given an array(arr) containing only 0's, 1's, and 2's.
2. You have to sort the given array in increasing order and in linear time.

Constraints

```
1 <= N <= 10000
arr[i] = 0,1,2
```

Format

Input

An Integer N
arr1
arr2..

n integers

Output

Check the sample output and question video.

Example

Sample Input

```
10
1
0
2
2
1
0
2
1
0
2
```

Sample Output

```
Swapping index 1 and index 0
Swapping index 2 and index 9
Swapping index 2 and index 8
Swapping index 2 and index 1
Swapping index 3 and index 7
Swapping index 5 and index 2
Swapping index 6 and index 6
0
0
0
1
1
1
2
2
2
2
```

```
#include <iostream>
#include <vector>

using namespace std;

void input(vector<int> &arr)
{
    for (int i = 0; i < arr.size(); i++)
    {
        cin >> arr[i];
    }
}
```

```
void print(vector<int> &arr)
{
    for (int i = 0; i < arr.size(); i++)
    {
        cout << arr[i] << endl;
    }
    cout << endl;
}

// used for swapping ith and jth elements of array
void swap(vector<int> &arr, int i, int j)
{
    cout << ("Swapping index " + to_string(i) + " and index " +
to_string(j)) << endl;
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}

void sort012(vector<int> &arr)
{
    int i{};
    int j{};
    int k{arr.size()-1};
  while(i <= k){
      if(arr[i] == 2){
          swap(arr, i,k);
          k--;
      }else if(arr[i] == 1){
          i++;
      }else{
          swap(arr ,i,j);
          i++;
          j++;
      }

  }


}

int main()
{
    int n, m;
    cin >> n;
    vector<int> A(n, 0);
    input(A);

    sort012(A);
    print(A);
    return 0;
}
```

Target Sum Pair 1

Easy

1. You are given an array(arr) of distinct integers and a target. 2. You have to print all the pairs having their sum equal to the target.

Constraints

1 <= N <= 10000 -10^9<= arr[i] <= 10^9

Format

Input

An Integer N arr1 arr2.. n integers An integer target

Output

Check the sample output and question video.

Example

Sample Input

```
12
9
-48
100
43
84
74
86
34
-37
60
-29
44
160
```

Sample Output

```
60, 100
74, 86
```

```cpp
#include <bits/stdc++.h>
using namespace std;
void input(vector<int> &arr)
{
    for (int i = 0; i < arr.size(); i++)
    {
        cin >> arr[i];
    }
}
vector<int> mergeTwoSortedArrays(vector<int> &A, vector<int> &B)
{
    vector <int> result;

    int a = A.size();
    int b = B.size();

    int ai{};
    int bi{};


    while(ai <a && bi < b){
        if (A[ai]<B[bi]){
            result.push_back(A[ai]);
            ai++;
        }else{
            result.push_back(B[bi]);
            bi++;
        }
    }
    for( int i {bi};i < b;i++){
        result.push_back(B[i]);
    }
    for( int i {ai};i<a;i++){
        result.push_back(A[i]);
    }
    return result;
}

vector<int> mergeSort(vector<int> &arr ,int si ,int ei)
{
    if(si == ei){
        vector <int> r ;
        r.push_back(arr[si]);
        return r;
    }
    int mid = (si+ei)/2;
    vector<int> A = mergeSort(arr, si,mid);
    vector<int> B = mergeSort(arr,mid+1,ei);
    return mergeTwoSortedArrays(A,B);

}
void targetSumPair(vector<int> &arr, int target)
```

```
{
    //write your code here

    int n = arr.size();
    arr = mergeSort(arr ,0,n-1); //sorted


    int s{};
    int l{n-1};
    while(s<l){
        if(arr[s]+arr[l] == target){
            cout<<arr[s]<<", "<<arr[l]<<endl;
            s++;
            l--;
        }else if (arr[s]+arr[l] < target){
            s++;
        }else if (arr[s]+arr[l] > target){
            l--;
        }

    }


}
int main()
{
    int n, target;
    cin >> n;
    vector<int> vec(n, 0);
    input(vec);
    cin >> target;
    targetSumPair(vec, target);
    return 0;
}
```

Pivot In Sorted And Rotated Array

Easy

1. You are given an array(arr) of distinct integers, which is sorted and rotated around an unknown point.
2. You have to find the smallest element in O(logN) time-complexity

Constraints

```
1 <= N <= 10000
-10^9 <= arr[i] <= 10^9
```

Format

Input

An Integer N
arr1
arr2..
n integers

Output
The smallest element
Example
Sample Input
```
9
15
16
19
21
23
24
1
2
12
```
Sample Output
```
1
```
```cpp
#include <iostream>
#include<vector>
using namespace std;
void input(vector<int> &arr)
{
    for (int i = 0; i < arr.size(); i++)
    {
        cin >> arr[i];
    }
}
int findpivot(vector<int> &arr)
{
    //write your code here
    int n = arr.size();
    int l = 0;
    int r = n-1;
     int mid {(r+l)/2};
    while (l<r){
        // int mid = (r+l)/2;
        if(arr[mid]< arr[r]){
            r = mid;
        }else if(arr[mid] > arr[r]){
            l = mid+1;
        }
        mid = (r+l)/2;
    }
    return arr[l];
}
int main()
{
    int n;
    cin >> n;
    vector<int> vec(n, 0);
    input(vec);
    int pivot = findpivot(vec);
    cout << pivot;
    return 0;
}
```

Linear Search vs Binary Search