Sorts

1

```cpp
//bubble sort
#include <iostream>
using namespace std;
void bubble_sort(int a[], int n){
    for(int times=1; times<=n-1; times++){
        //repeated swapping
        for (int i = 0; i <= n-times-1; ++i)
        {
            if(a[i] > a[i+1]){
                swap(a[i],a[i+1]);
            }
        }
    }
}
int main(){
    int arr[] = {-2,3,4,-1,-5,-12,6,1,3};
    int n = sizeof(arr)/sizeof(n);
    bubble_sort(arr,n);
    for(auto x : arr){
        cout<<x<<" , ";
    }
    return 0;
}
```

2

```cpp
#include<iostream>
using namespace std;
void insertionSort(int arr[], int n){
    for(int i=1; i<n; i++){
        int current=arr[i];
        int prev = i - 1;
        while(prev>=0 and arr[prev]>current){
            arr[prev+1] = arr[prev];
            prev = prev - 1;
        }
           arr[prev+1] = current;
    }
}
int main(){
    int n ;
    cin>>n;
    int arr[n];
    for(int i = 0; i<n; i++){
        cin>>arr[i];
```

```
    }
    insertionSort(arr,n);
    for(auto x : arr){
        cout << x << " ";
    }
}
```

3

```
#include <iostream>
using namespace std;
void insertion_sort(int a[],int n){
    for (int i = 1; i < n; i++)
    {
        int current=a[i];
        int prev=i-1;
        //loop to find the right index where the element
        //current should be inserted
        while(prev>=0 and a[prev] > current){
            a[prev+1]=a[prev];
            prev = prev - 1;
        }
        a[prev+1] = current;
    }
}
int main(){
int arr[] = {-2,3,4,-1,-5,-8,6,1,3};
    int n = sizeof(arr)/sizeof(n);
    insertion_sort(arr,n);
    for(auto x : arr){
        cout<<x<<" , ";
    }
    return 0;
}
```

4

```
//selection sort
#include <iostream>
using namespace std;
//sort the elements in increasing order
void selection_sort(int a[],int n){
    for(int pos=0; pos <= n-2; pos++){
        int current = a[pos];
        int min_position = pos;
        //find out the element
        for (int j = pos; j < n; j++)
        {
```

```
            if(a[j] < a[min_position]){
                min_position = j;
            }
        }
        //swapping outside the loop
        swap(a[min_position],a[pos]);
    }
}
int main(){
    int arr[]={-2,4,-1,-5,-12,6,1,3};
    int n=sizeof(arr)/sizeof(int);
    selection_sort(arr,n);
    for(auto x:arr){
        cout<<x<<",";
    }
    return 0;
}
```

5

```
#include <iostream>
#include <algorithm>
using namespace std;
bool compare(int a, int b ){
    return a>b;
}
int main(){
    int arr[]={2,4,0,-1,3,22,33,11,4,5};
    int n = sizeof(arr)/sizeof(int);
    sort(arr,arr + n,compare);
    //compare is not a function call it is passing a function as parameter
    // reverse(arr, arr + n); //or we can use the compare
    //print the output
    sort(arr,arr + n, greater<int>());
    // aa greater nu pan lakhi ne reverse ma lakhi sakiae
    for(int x:arr){
        cout<<x<<" ";
    }
    return 0;
}
```

6

```
//counting sort
//when data is in range then we can use this sort
// complexity O(n) linear + range in terms of the range
//if data in the range complexity  O(n)
#include <bits/stdc++.h>
```

```cpp
using namespace std;
void counting_sort(int a[], int n){
    //largest element
    int largest = INT_MIN;
    for(int i = 0; i<n; i++){
        largest = max(largest,a[i]);
    }
    //creating the count array/vector
    vector<int> freq(largest+1,0);

    //update the frequent array
    for (int i = 0; i < n; i++)
    {
        freq[a[i]]++;
    }
    //put back the elements from freq into original array
    int j = 0;
    for (int i = 0; i <= largest; ++i)
    {
        while(freq[i]>0){
            a[j]=i;
            freq[i]--;
            j++;
        }
    }
    return;
}
int main(){
    int arr[] = {1,0,56,3,5,7,2222222};
    int n = sizeof(arr)/sizeof(int);
    counting_sort(arr,n);
    for (int i = 0; i < n; i++)
    {
        cout << arr[i] <<" ";
    }
    return 0;
}
```