**Mozilla Firefox Club**

**Domain :CP**

**Task : Implement a simple sorting Algorithm and Compare Performance**

**1.BUBBLE SORT**

#include<iostream>

using namespace std;

void bubble\_sort(int \*arr,int n){

for(int i=0;i<=n-1;i++){

bool swaped=false;

for(int j=0;j<n-1-i;j++){

if(arr[j+1]>arr[j]){

swap(arr[j],arr[j+1]);

swaped=true;

}}

if(swaped==false){

break;

}

}

}

int main(){

int arr[]={5,0,2,0,4,1};

int n=sizeof(arr)/sizeof(arr[0]);

bubble\_sort(arr,n);

for(int i=0;i<=n-1;i++){

cout<<arr[i]<<" ";

}

}

**OUPUT**

5 4 2 1 0 0

**CHARACTERISTICS**

Time Complexity :O(n^2)

Stability :stable(does not change the relative order of elements with equal keys)

Space Complexity : O(1)

**2.INSERTION SORT**

#include <iostream>

using namespace std;

void insertion\_sort(int arr[], int n) {

for (int i = 1; i < n; i++) {

int key = arr[i];

int j = i - 1;

while (j >= 0 && arr[j] > key) {

arr[j + 1] = arr[j];

j--;

}

arr[j + 1] = key;

}

}

int main() {

int arr[] = {5, 3, 7, 2, 1};

int n = sizeof(arr) / sizeof(arr[0]);

insertion\_sort(arr, n);

for (int i = 0; i < n; i++) {

cout << arr[i] << " ";

}

cout << endl;

return 0;

}

**OUPUT**

1 2 3 5 7

**CHARACTERISTICS**

Time Complexity :O(n^2)

Stability : Stable (does not change the relative order of elements with equal keys)

Space Complexity : O(1)

**3.SELECTION SORT**

#include <iostream>

using namespace std;

void selection\_sort(int arr[], int n) {

for (int i = 0; i < n - 1; i++) {

int min\_index = i;

for (int j = i + 1; j < n; j++) {

if (arr[j] < arr[min\_index]) {

min\_index = j;

}

}

swap(arr[i], arr[min\_index]);

}

}

int main() {

int arr[] = {5, 3, 7, 2, 1};

int n = sizeof(arr) / sizeof(arr[0]);

selection\_sort(arr, n);

for (int i = 0; i < n; i++) {

cout << arr[i] << " ";

}

cout << endl;

return 0;

}

**OUPUT**

1 2 3 5 7

**CHARACTERISTICS**

Time Complexity :O(n^2)

Stability : Not stable (does not change the relative order of elements with equal keys)

Space Complexity : O(1)