Sorting Comparisons

Q-1. Compare Bubble sort and Selection sort based on the following:

a. Number of comparisons

b. Number of swaps

c. In-place and Out-place implementations

A-1.

a. No of comparisons in selection sort is less than no of comparisons in bubble sort. For an example take array= [10,5,4,12,20,6]. Now number of comparisons needed to sort this array through selection sort is 5 and number of comparisons needed to sort this array through bubble sort is 12.

b. For an example take array= [10,5,4,12,20,6]. Number of swaps needed to sort this array through selection sort is 3 and number of swaps needed to sort this array through bubble sort is 6.

c. Code- Bubble Sort

```
// C++ program for implementation
// of Bubble sort
#include <bits/stdc++.h>
using namespace std;
// A function to implement bubble sort
void bubbleSort(int arr[], int n)
{
int i, j;
for (i = 0; i < n - 1; i++)
// Last i elements are already
// in place
for (j = 0; j < n - i - 1; j++)
if (arr[j] > arr[j + 1])
swap(arr[j], arr[j + 1]);
}
// Function to print an array
void printArray(int arr[], int size)
{
int i;
for (i = 0; i < size; i++)
cout << arr[i] << " ";
cout << endl;
}
// Driver code
int main()
{
int arr[] = { 5, 1, 4, 2, 8};
int N = sizeof(arr) / sizeof(arr[0]);
bubbleSort(arr, N);
cout << "Sorted array: \n";
printArray(arr, N);
return 0;
}
```

Code- Selection Sort

```
// Selection sort
#include <iostream>
using namespace std;
// function to swap the the position of two elements
```

```
void swap(int *a, int *b) {
 int temp = *a;
 *a = *b;
 *b = temp;
}
// function to print an array
void printArray(int array[], int size) {
 for (int i = 0; i < size; i++) {
 cout << array[i] << " ";
 }
 cout << endl;
}
void selectionSort(int array[], int size) {
 for (int step = 0; step < size - 1; step++) {
 int min_idx = step;
 for (int i = step + 1; i < size; i++) {
 // To sort in descending order, change > to < in this line.
 // Select the minimum element in each loop.
 if (array[i] < array[min_idx])
 min_idx = i;
 }
 // put min at the correct position
 swap(&array[min_idx], &array[step]);
 }
}
// driver code
int main() {
 int data[] = {20, 12, 10, 15, 2};
 int size = sizeof(data) / sizeof(data[0]);
 selectionSort(data, size);
 cout << "Sorted array in Ascending Order:\n";
 printArray(data, size);
}
```

```
//Bonus
import time
start_time=time.time()
for i in range(0,100000):
    selectionSort(data1, size1)
end_time=time.time()
tim=end_time-start_time
print('time taken by selection sort')
print(tim)
```

```
import time
start_time=time.time()
for i in range(0,100000):
     bubbleSort(data2)
end_time=time.time()
tim1=end_time-start_time
print('time taken by bubble sort')
print(tim1)
```