KIET Group of Institutions, Ghaziabad

Department of Computer Applications

(An ISO – 9001: 2015 Certified & 'A' Grade accredited Institution by NAAC)

Design and Analysis of Algorithm

RCA 352: Session 2020-21

DAA Lab

Experiment-No.9

Objective: Implement the **Bubble sort** algorithm to sort the given list of N numbers and plot graph

Scheduled Date:	Compiled Date:	Submitted Date:
25-09-20	25-09-20	25-09-20

Algorithm:

BubbleSort(Input: Array A, Size N)

N: Number of values to be sort

A: Array of Size N

Temp, Pass,J : extra variable

1. Pass=1
2. while(pass<=n) do:
3. J :=1;
2. while(j<=n-pass) do:
3. if(a[j]>a[j+1])
4. temp :=a[j];
5. a[j] :=a[j+1];
6. a[j+1] :=temp;
7. j := j+1
8. end while
9. pass := pass+1
10. end while

Program file bubble_sort.c :

```
#include<stdio.h>
#include<conio.h>
#include<process.h>
#include<alloc.h>
int count=0;
void main()
{
      void getdata(int[10],int);
      void putdata(int[10],int);
      void bubble_sort(int a[],int);
      int i,a[100],n;
      clrscr();
      printf("enter the value of n\n");
      scanf("%d",&n);
      getdata(a,n);
      printf("\nbefore soring\n");
      putdata(a,n);
      bubble_sort(a,n);
```

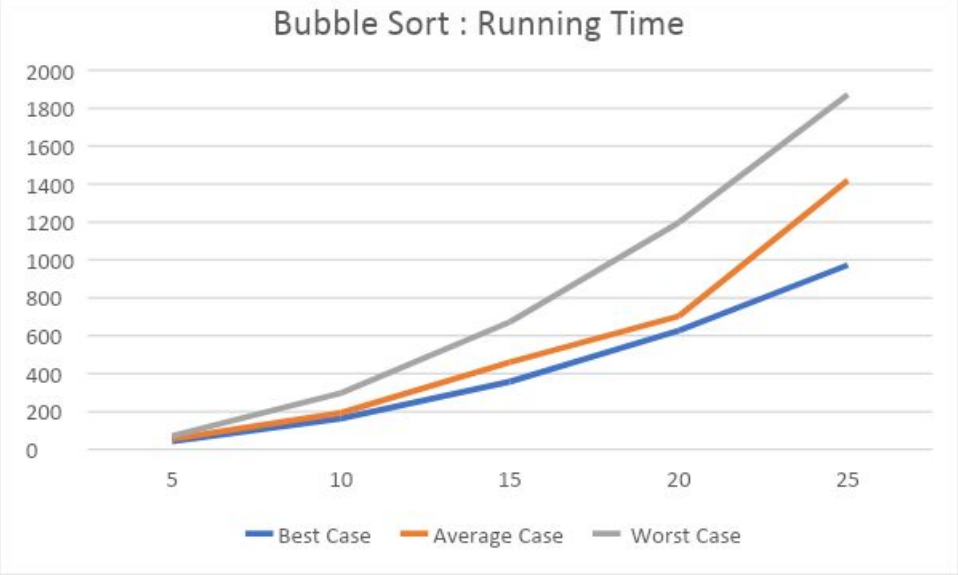
```
      printf("\nafter sorting\n");
      putdata(a,n);
      printf("\n for n = %d value of count is  %d",n,count);
      getch();
}
void getdata(int x[10],int n)
{
     int k;
     printf("enter the value  for sorting\n");
     for(k=0;k<n;k++)
     {
      scanf("%d",&x[k]);
     }
}
void putdata(int x[10], int n)
{
      int k;
      for(k=0;k<n;k++)
      {
            printf("%d\t",x[k]);
       }
       printf("\n");
}
void bubble_sort(int a[],int n)
{
       int pass,j,temp;
       count++;

       for(pass=1;pass<=n-1;pass++)
       {
         count++;
         count++;
         for(j=0;j<n-pass;j++)
         {
          count++;
          count++;
          if(a[j]>a[j+1])
          {
           count++;
           temp=a[j];
           count++;
           a[j]=a[j+1];
           count++;
           a[j+1]=temp;
          }
          count++;
```

```
            }
            count++;
          }
}
```

Output

Input Size	Best Case	Average Case	Worst Case
5	43	55	72
10	163	193	298
15	358	460	673
20	628	703	1198
25	973	1420	1873

Graph



Conclusion

Case	Running Time : Growth of Function mathematically	Running Time : Growth of Function after observing graph
Best Case	O(n^2)	O(n^2)
Average Case	O(n^2)	O(n^2)
Worst Case	O(n^2)	O(n^2)