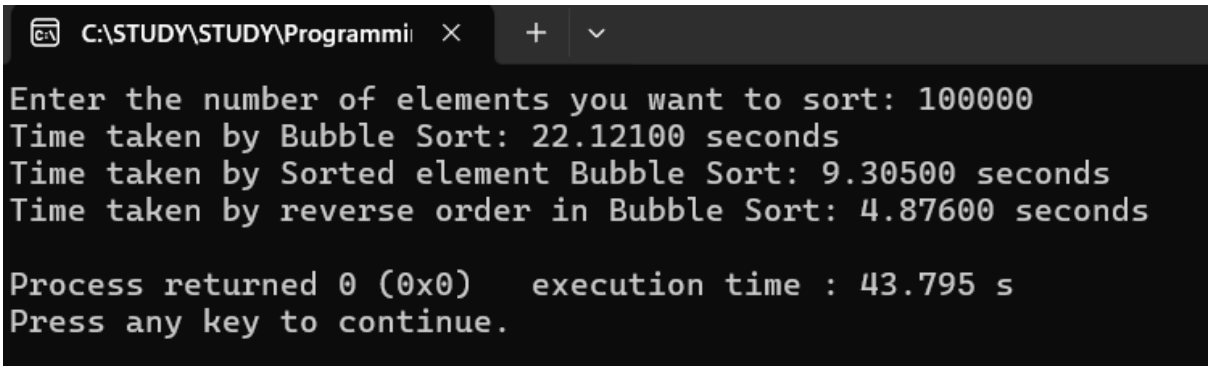
PRACTICAL 2

AIM: Write a program to sort given elements of an array in ascending order using Bubble sort. Analyze the time complexity for best, average and worst case.

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>

int main() {
        int SIZE;
   printf("Enter the number of elements you want to sort: ");
   scanf("%d", &SIZE);

   int* arr = (int*)malloc(SIZE * sizeof(int));
   for(int i=0;i<SIZE;i++){
      arr[i] = rand() % SIZE;
   }
   clock_t start, end;
        double cpu_time;

        start = clock();
        int temp;
        for(int i = 0; i < SIZE - 1; i++) {
                for(int j = 0; j < SIZE - i - 1; j++) {
                        if(arr[j] > arr[j + 1]) {
                                temp = arr[j];
                                arr[j] = arr[j + 1];
                                arr[j + 1] = temp;
                        }
                }
        }
        end = clock();

        cpu_time = ((double)(end - start)) / CLOCKS_PER_SEC;
        printf("Time taken by Bubble Sort: %.5f seconds\n", cpu_time);

   clock_t start2, end2;
   double cpu_time2;

   start2 = clock();
   for(int i = 0; i < SIZE - 1; i++) {
                for(int j = 0; j < SIZE - i - 1; j++) {
                        if(arr[j] > arr[j + 1]) {
```

```
                                temp = arr[j];
                                arr[j] = arr[j + 1];
                                arr[j + 1] = temp;
                        }
                }
        }
    end2 = clock();

    cpu_time2 = ((double)(end2 - start2))/CLOCKS_PER_SEC;
    printf("Time taken by Sorted element Bubble Sort: %.5f seconds\n", cpu_time2);

    clock_t start3, end3;
    double cpu_time3;

    start3 = clock();
    for(int i = SIZE - 2; i >= 0; i--) {
                for(int j = SIZE - i - 2; j >= i; j--) {
                        if(arr[j] > arr[j + 1]) {
                                temp = arr[j];
                                arr[j] = arr[j + 1];
                                arr[j + 1] = temp;
                        }
                }
        }
    end3 = clock();

    cpu_time3 = ((double)(end3 - start3))/CLOCKS_PER_SEC;
    printf("Time taken by reverse order in Bubble Sort: %.5f seconds\n",cpu_time3);

    free(arr);
        return 0;
}
```

OUTPUT:



Output for different size of input

	30000	50000	80000	100000	150000
Random Array	0.76500	5.25900	14.0110	22.12100	51.60600
Sorted Array	0.84500	2.34100	5.98300	9.30500	21.60500
Reverse Sorted Array	0.45700	1.25300	3.21000	4.87600	11.19900

Where elapsed time is in seconds.