PRACTICAL – 2

Name:- Manish M Prajapati (190420116054)

Subject:- Analysis and Design of Algorithms (3150703)

Class:- IT 3rd Year, 5th semester.

Question:- Implementation and Time analysis of sorting algorithms: Bubble sort

Answer:-

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

//PRACTICAL 2
//Implementation and Time analysis of sorting algorithms: Bubble sort

int count = 0;

int * bubble_sort(int *cptr, int length)
{
	int *copy = cptr;
	int i=0, j=0, s;
	int swap;


	for(i=0; i<length-1; i++)
	{
		swap = 0;
		for(j=0; j<length-1-i; j++)
		{
			count++;
			if(*(cptr+j)>=*(cptr+j+1))
			{
				swap = 1;
				s = *(cptr+j);
				*(cptr+j) = *(cptr+j+1);
				*(cptr+j+1) = s;
			}
		}
		if(swap==0)
		{
			break;
		}
	}

	return cptr;
```

```
}

int main()
{
    int n, i;
    int *p;

    printf("Enter the number of elements you want to enter: ");
    scanf("%d", &n);

    p = (int *) malloc(n*(sizeof(int)));

    printf("Enter the terms: \n");
    for(i=0; i<n; i++)
    {
        scanf("%d", p+i);
    }
    printf("\n");

    printf("The list before sorting:\n");

    for(i=0; i<n; i++)
    {
        if(i==n-1)
        {
            printf("%d.", *(p+i));
        }
        else
        {
            printf("%d, ", *(p+i));
        }
    }

    p = bubble_sort(p, n);

    printf("\n\n");
    printf("The time complexity of bubble sorting is O(n^2).\n\n");
    printf("According to given formula, theoritically, iterations are %d.\n\n", n*n);
    printf("Practically performing, the number of iterations taken are %d.\n\n", count);
    printf("The list AFTER sorting:\n");

    for(i=0; i<n; i++)
    {
        if(i==n-1)
        {
            printf("%d.", *(p+i));
        }
        else
```

```
                {
                        printf("%d, ", *(p+i));
                }
        }


        return 0;
}
```

<u>Output</u>:-

```
Enter the number of elements you want to enter: 4

Enter the terms:
12
-33
64
56

The list before sorting:
12, -33, 64, 56.

The time complexity of bubble sorting is O(n^2).

According to given formula, theoretically, iterations are 16.

Practically performing, the number of iterations are 5.

The list AFTER sorting:
-33, 12, 56, 64.
```

--------________--------________--------________--------________--------________--------________--------

<u>TIME ANALYSIS FOR BUBBLE SORTING</u>

<u>Worst Case</u>:-

It mainly occurs when the elements in the list are entered in the descending order. Secondarily, when the list is huge to be sorted.

From above program, in the function ‘bubble_sort’, the outer ‘for’ loop will run (n-2) times and inner ‘for’ loop will run in decreasing order of iterations, depending on increasing value of ‘i’, i.e., (n-i-2) times.

Let us consider, the number is large, i.e., n. In that case, the outer loop will run (n-2)times and for inner loop, it would be:

$S(n) = (n-2) + (n-2-1) + (n-2 -2) + \ldots\ldots + (n-2 -(n-3))$
$S(n) = (n)(n-2) + (-2 -3 -4 -5 \ldots\ldots (n-3) \text{ times})$
$S(n) = (n)(n -2) + ((n-3)(n-2)/2)$
$S(n) = n^2 - 2n + (n^2 - 5n + 6)/2$
$S(n) = (2n^2 - 4n + n^2 - 5n + 6)/2$

$S(n) = (3)(n^2 – 3n + 2)/2$
$S(n) = (3)(n-1)(n-2)/2$

From above equation, it is clear that time complexity is almost n^2.

Therefore, the time complexity of worst case bubble sort method is $O(n^2)$.

Average Case:-

In the case of less number of elements in the list, it is easier to find out the average time complexity of the program.

From above program, in the function ‘bubble_sort’, the outer ‘for’ loop will run (n-2) times and inner ‘for’ loop will run in decreasing order of iterations, depending on increasing value of ‘i’, i.e., (n-i-2) times.

Let us consider the case, where the elements entered are 4, like: -12, 45, 12, 33.

In the first pass, the loop will run 3 times, and the list would look like: -12, 12, 33, 45.
In the second pass, the loop will run 2 times, and the list will remain same: -12, 12, 33, 45. Here, as there is no swapping done, so, the ‘bubble_sort’ function will return the list as the sorting is done.

So, the total number of iterations were: $S(n) = 3+2 = 5$.
And, n^2 is 16. The iterations are less than 16, hence follow the time complexity $O(n^2)$.

Hence, the average case time complexity of bubble sort method is $O(n^2)$.

Best Case:-

In this program, the best case scenario will occur when the elements entered would be in ascending order.

Let us consider the case of a list containing: 1, 2, 3, 4, 5.

Passing above elements to function ‘bubble_sort’, for only one time in inner loop, the list will be seen with no swapping of elements, leading to conclusion that list being already sorted, hence returning the list without performing any operations.

In first pass, the inner loop will make comparisons of 4 pairs, and list will be: 1, 2, 3, 4, 5.
Hence, there will be only 4 iterations, i.e., $S(n) = 4 \Rightarrow S(n) = n – 1 \approx n$

Hence, the time complexity for best case bubble sort method is $O(n)$.