Name : Vanessa D’mello
Roll No. : 8863
Branch : SE Computer A (Batch A)
Practical No. : 1

Bubble Sort

```
#include<stdio.h>
#define MAX 100
int main()
{
	int a[MAX],n,i,j,temp;
	printf("Enter the size of array: ");
	scanf("%d",&n);
	printf("Enter the array elements: ");
	for(i=0;i<n;i++)
		scanf("%d",&a[i]);
	for(i=1;i<n;i++)
	{
		for(j=0;j<n-i;j++)
		{
			if(a[j]>a[j+1])
			{
				temp=a[j];
				a[j]=a[j+1];
				a[j+1]=temp;
			}
		}
	}
	printf("\nArray after using bubble sort: ");
	for(i=0;i<n;i++)
		printf("%d ",a[i]);
	return 0;
}
```

C:\Users\dmell\OneDrive\Desktop\Subjects\AOA\BubbleSort.exe

```
Enter the size of array: 5
Enter the array elements: 8     25      1       4       6

Array after using bubble sort: 1 4 6 8 25
Process returned 0 (0x0)   execution time : 13.759 s
Press any key to continue.
```

Modifed Bubble Sort

```
#include<stdio.h>
#define MAX 100
int main()
{
	int a[MAX],n,i,j,temp,flag=-1;
	printf("Enter the size of array: ");
	scanf("%d",&n);
	printf("Enter the array elements: ");
	for(i=0;i<n;i++)
		scanf("%d",&a[i]);
	for(i=1;i<n && flag==-1;i++)
	{
		flag=0;
		for(j=0;j<n-i;j++)
		{
			if(a[j]>a[j+1])
			{
				temp=a[j];
				a[j]=a[j+1];
				a[j+1]=temp;
				flag=-1;
			}
		}
	}
	printf("\nArray after using bubble sort: ");
	for(i=0;i<n;i++)
		printf("%d ",a[i]);
	return 0;
}
```

C:\Users\dmell\OneDrive\Desktop\Subjects\AOA\ModifiedBubbleSort.exe

```
Enter the size of array: 5
Enter the array elements: 5     25      8       1       9

Array after using bubble sort: 1 5 8 9 25
Process returned 0 (0x0)   execution time : 7.009 s
Press any key to continue.
```

8863

① Space Complexity

The space complexity for bubble sort and modified bubble sort is $O(1)$ as extra space is required is a a single additional space for temp variable use while swapping 2 elements

② a) Bubble Sort Time Complexity

$$f(n) = \sum_{pass=1}^{n-1} \sum_{i=0}^{n-pass-1} 1$$

$$= \sum_{pass=1}^{n-1} (n-pass-1-0+1)$$

$$= \sum_{pass=1}^{n-1} (n-pass)$$

$$= (n-1) + (n-2) + \cdots + 1$$

$$= \sum_{i=1}^{n-1} i$$

$$= \frac{n(n-1)}{2}$$

$$= \frac{n^2}{2} - \frac{n}{2}$$

By ignoring the lower order terms and the constant coefficient of higher order terms we get

$f(n) = O(n^2)$

Thus, ~~Θ~~ time complexity is quadratic

b) Modified Bubble Sort Time Complexity.

i) Best Case

This case is when array is already sorted

eg. | 1 | 2 | 3 | 4 | 5 |

There will be 1 pass but there are 4 comparisons

∴ in 1 pass $(n-1)$ comparison

8863

PAGE No.
DATE

$\therefore$ By ignoring lower order terms and the constant coefficient of higher order terms, we get

$f(n) = \Omega(n)$

Thus, best case time complexity is linear

2] <u>Worst Case</u>

It happens when array is reverse order

eg

5	4	3	2	1

In this case, modified sort works as bubble sort

$\therefore$ $(n-1)$ passes takes place.

$\therefore$ $f(n) = O(n^2)$

$\therefore$ Worst case complexity is quadratic

3] <u>Average Case</u>

Here we have to consider all the input cases

let $c(i)$ denote array get sorted after pass i

Therefore

$c(i)$	comparisons
$c(1)$	$(n-1)$
$c(2)$	$(n-1)+(n-2)$
$\vdots$	$\vdots$
$(n-1)$	$(n-1)+(n-2)+\dots 1$

$$\therefore c(i) = \sum_{j=1}^{i} n-j$$

$$= \sum_{j=1}^{i} n - \sum_{j=1}^{i} j$$

$$= n * i - \frac{(i+1)(i)}{n}$$

Assuming all cases are occuring equally likely

$$\therefore \text{time taken} = \frac{c(1)}{n-1} + \dots + \frac{c(n-1)}{n-1}$$

8863

$$= \frac{1}{(n-1)} \sum_{i=1}^{n-1} c(i)$$

$$= \frac{1}{n-1} \sum_{i=1}^{n-1} \left(n * i - \frac{(i+1)}{2} i \right)$$

$$= \frac{1}{(n-1)} \frac{(n-1)n}{2} - \frac{1}{2(n-1)} \sum_{i=1}^{n-1} i^2 - \frac{1}{2(n-1)} \sum_{i=1}^{n-1} i$$

$$= \frac{n^2}{2} - \frac{1}{2(n-1)} \frac{(n-1)n(2n-1)}{6} - \frac{1}{2(n-1)} \frac{n(n-1)}{2}$$

$$= \frac{n^2}{2} - \frac{2n^2 - n}{2} - \frac{n}{4}$$

∴ By ignoring lower order terms and constant coefficient of higher order

$$f(n) = \theta(n^2)$$

∴ Average case complexity is quadratic