PRACTICAL – 3

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: Selection sort

Answer:-

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

int count = 0;

int * selection_sort(int *ptr, int length)
{
	int *copy = ptr;
	int j, i, temp;

	for(i=0; i<length-1; i++)
	{
		for(j=i+1; j<length; j++)
		{
			count ++;
			if(*(ptr+i)>=*(ptr+j))
			{
				temp = *(ptr+j);
				*(ptr+j) = *(ptr+i);
				*(ptr+i) = temp;
			}
		}
	}

	return ptr;

}

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 = selection_sort(p, n);

	printf("\n\n");
	printf("The time complexity of Selection 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;
}
```

Output:-

Enter the number of elements you want to enter: 4

Enter the terms:
88
34
5
-99

The list before sorting:
88, 34, 5, -99.

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

According to given formula, theoretically, iterations are 16.

Practically performing, the number of iterations are 6.

The list AFTER sorting:
-99, 5, 34, 88.

--------_____--------_____--------_____--------_____--------_____--------_____--------

TIME ANALYSIS FOR SELECTION SORTING

Worst Case:-

From above program, in the function 'selection_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-1-i) 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:

$S(n) = (n-1) + (n-1-1) + (n-1-2) + ...+(n-1-(n-2))$
$S(n) = (n)(n-2) + (-1 -2 -3 -4 -5 ..... (n-2) \text{ times})$
$S(n) = (n)(n -2) + ((n-2)(n-1)/2)$
$S(n) = n^2 – 2n + (n^2 – 3n + 2)/2$
$S(n) = (2n^2 – 4n + n^2 – 3n + 2)/2$
$S(n) = (3n^2 – 7n + 2)/2$
$S(n) = (3n-1)(n-2)/2$

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

Therefore, the time complexity of worst case selection 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 ‘selection_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-1) times.

Let us consider the case, where the elements entered are 4, like: -11, 46, 11, 32.

In the first pass, the loop will run 3 times, and the list would look like: -11, 46, 11, 32.
In the second pass, the loop will run 2 times, and the list will remain same: -11, 11, 32, 46.
In the third pass, the loop will run 1 time and the list will remain same: -11, 11, 32, 46.

So, the total number of iterations were: S(n) = 3+2+1 = 6.
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 Selection sort method is $O(n^2)$.

Best Case:-

Although, the list given in ascending order, the iterations will be ((3n-1)(n-2)/2) times, so, the time complexity of best case is same as that of worst and average case.

Hence, the time complexity for best case Selection sort method is $O(n^2)$.

--------_________--------_________--------_________--------_________--------_________--------_________--------
_________--------_________--------_________--------_________--------_________--------_________--------_________
--------_________--------_________--------_________--------_________--------_________--------_________--------