EXPERIMENT 3

Experiment No: 3 **Date:** 12/04/2021

Aim: Implementation of MaxMin Algorithm and obtain its step count

Theory:

MaxMin

- MaxMin Algorithm is an algorithm that helps to search for the maximum and minimum element for the array.
- There are two methods by which this can be performed.
- The first method is to use a for loop to traverse the array and we compare if the maximum_element is smaller than arr[i], if it is true than value of arr[i] is assigned to maximum_element.
- “arr” here is the array while arr[i] is the element of the array at index i. maximum_element refers to the maximum element of the array.
- Similarly for the maximum_element, while the array is being traversed we check if maximum_element is greater than arr[i].
- If so then value of arr[i] is assigned to maximum_element.
- The second method is to use the divide and conquer strategy to find the maximum and minimum element of the array.
- In this method, the array is recursively divided into smaller halves and the maximum and minimum elements of each of these subproblems(or halves) are determined.
- We keep on dividing the array recursively until we get a half having only one element(in which case both the maximum and minimum element is that one element) or a half having only two elements, of which one is the maximum element and the other is the minimum element.

- The maxima and minima values of these smaller subproblems are returned and the two maxima and the two minima are compared to obtain the maximum and minimum element of the previous subproblem.
- This process continues till we get the maximum and minimum elements of the whole array.

Algorithm

```
Algorithm MaxMin(i, j,  max, min)
//a[1:n] is a global array. Parameters I and j are integers,
// 1<= i <=j <=n. The effect is to set max and min to the largest
// and smallest values in a[i:j] respectively.
{
    if(i==j) then max := min := a[i]; //Small(P)
    else if(i=j-1) then // Another case of Small(P)
        {
            if(a[i]<a[j]) then
            {
                max := a[j]; min := a[i];
            }
            else
            {
                max := a[i]; min := a[j];
            }
```

EXPERIMENT 3

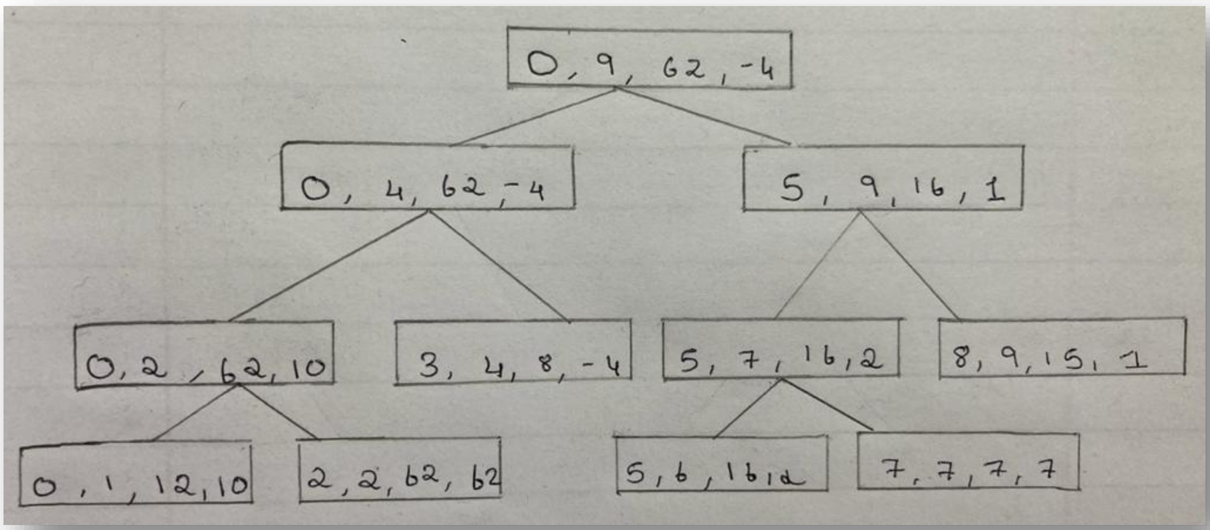
```
        }
        else
        {
                // if P is not small, divide P into subproblems.
                //Find where to split the set.
                        mid := (i+j)/2;
                //Solve the subproblems.
                        MaxMin(i,  mid, max, min);
                        MaxMin(mid+1, j, max1, min1);
                //Combine the solutions.
                if(max<max1) then max := min1;
                if(min>min1) then min := min1;
        }
}
```

EXPERIMENT 3

Tracing with Examples

Consider the following array:

Values:	12	10	62	-4	8	16	2	7	1	15
Index:	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]



Recurrence Relation

Let T(n) represent the number of comparisons needed for MaxMin, then the recurrence relation is:

$$T(n) = \begin{cases} 0 & , n = 1 \\ 1 & , n = 2 \\ T(n/2) + T(\lceil n/2 \rceil) + 2 & , n > 2 \end{cases}$$

$$\therefore T(n) = \begin{cases} 0 & , n = 1 \\ 1 & , n = 2 \\ 2T(\lceil n/2 \rceil) + 2 & , n > 2 \end{cases}$$

EXPERIMENT 3

Applying Repeated Substitution Method:

$$T(n) = 2T(n/2) + 2$$

$$= 2(2T(n/4)+2)+ 2$$

$$= 4T(n/4) +4 + 2$$

.

.

.

$$= 2k-1\ T(2) + \sum i\le 1\le k-1\ 2i$$

$$= 2k-1 + 2k -2$$

$$\therefore T(n) = 3n/2 - 2$$

$$= O(n)$$

PROGRAM

```
#include<iostream>

using namespace std;

int a[100];

int steps=0;

void maxmin(int,int,int&,int&);

int main()

{

   int n,s,loc,x,y;

   cout<<"------------------------------------"<<endl;
```

EXPERIMENT 3

```
cout<<"Enter Number of Terms for the Array:\n";

cout<<"------------------------------------"<<endl;

steps++;

cin>>n;

steps++;

cout<<"-----------------------------------"<<endl;

cout<<"Enter "<<n<<" Terms for the Array:\n";

cout<<"-----------------------------------"<<endl;

steps++;

for(int i=0;i<n;i++)

{

   steps++;

   cin>>a[i];

   steps++;

}

steps++;

maxmin(0,n-1,x,y);

steps++;

cout<<"\n\nThe Maximum Element is: "<<x<<endl;

cout<<"The Minimum Element is: "<<y<<endl;

steps+=2;

cout<<"\n***********"<<endl;
```

EXPERIMENT 3

```
    cout<<"Total Steps= "<<steps<<endl;

    cout<<"*************"<<endl;

}

void maxmin(int i,int j,int& max,int& min)

{

    int max1,min1,mid;

    steps++;

    if(i==j)

    {

        max=min=a[i];

        steps+=2;

    }

    else if(i==j-1)

    {

        steps++;

        steps++;

        if(a[i]<a[j])

        {

            max=a[j];

            min=a[i];

            steps+=2;

        }
```

EXPERIMENT 3

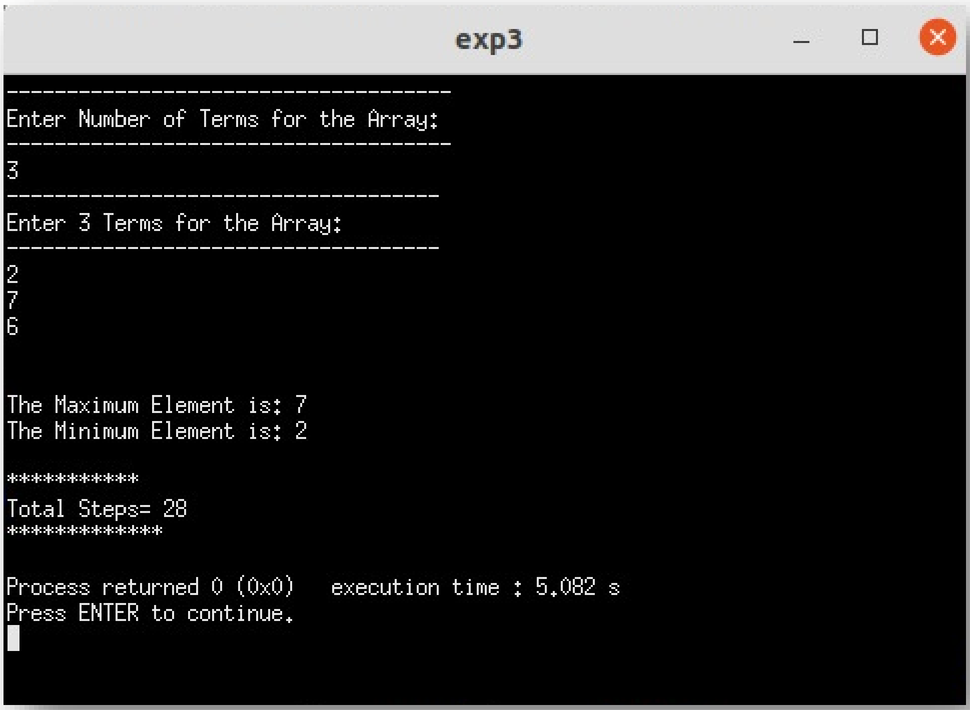
```
  else
  {
    max=a[i];
    min=a[j];
    steps+=2;
  }
}
else
{
  steps++;
  mid=(i+j)/2;
  steps++;
  maxmin(i,mid,max,min);
  steps++;
  maxmin(mid+1,j,max1,min1);
  steps++;
  steps++;
  if(max<max1)
  {
    max=max1;
    steps++;
  }
```

EXPERIMENT 3

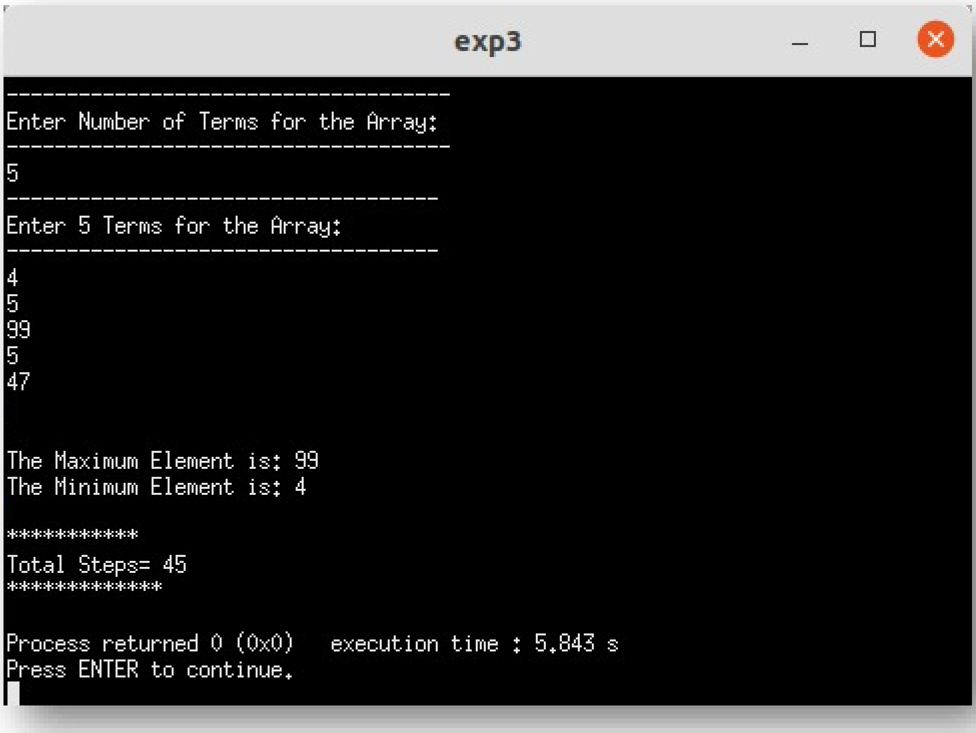
```
        steps++;

        if(min>min1)

        {

            min=min1;

            steps++;

        }

    }

}
```

EXPERIMENT 3

OUTPUT

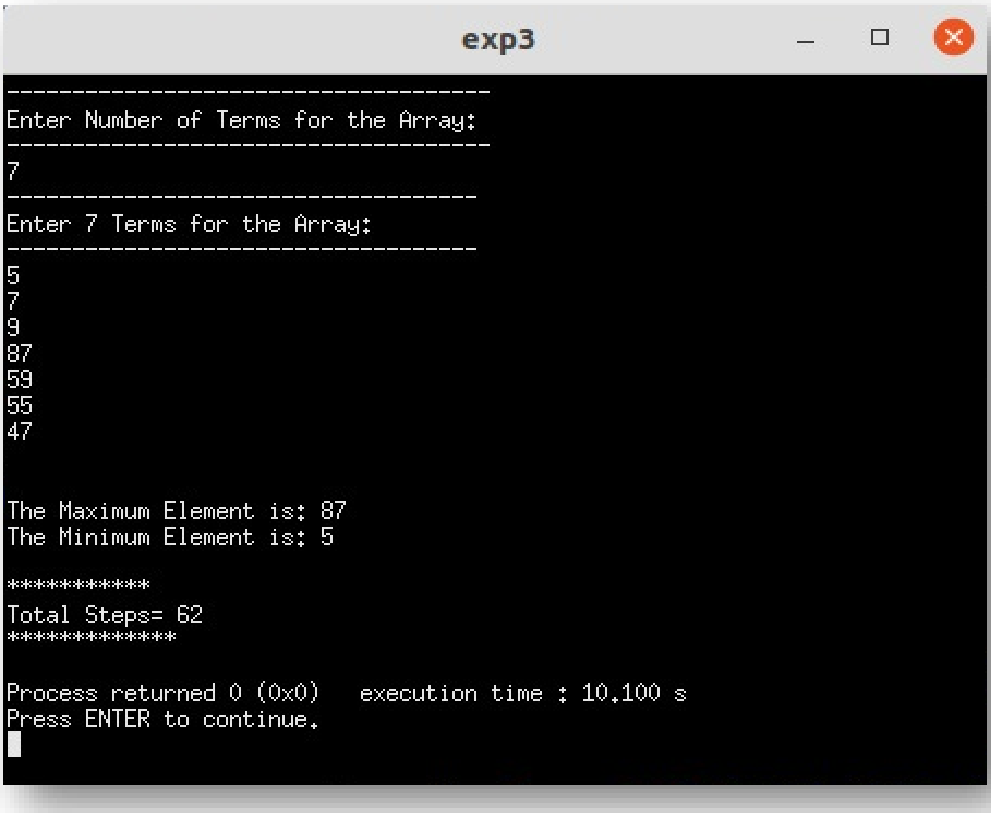


```
-------------------------------------
Enter Number of Terms for the Array:
-------------------------------------
3
------------------------------------
Enter 3 Terms for the Array:
------------------------------------
2
7
6


The Maximum Element is: 7
The Minimum Element is: 2

***********
Total Steps= 28
*************

Process returned 0 (0x0)   execution time : 5.082 s
Press ENTER to continue.
```



```
-------------------------------------
Enter Number of Terms for the Array:
-------------------------------------
5
------------------------------------
Enter 5 Terms for the Array:
------------------------------------
4
5
99
5
47


The Maximum Element is: 99
The Minimum Element is: 4

***********
Total Steps= 45
*************

Process returned 0 (0x0)   execution time : 5.843 s
Press ENTER to continue.
```

EXPERIMENT 3



```
-------------------------------------
Enter Number of Terms for the Array:
-------------------------------------
7
------------------------------------
Enter 7 Terms for the Array:
------------------------------------
5
7
9
87
59
55
47


The Maximum Element is: 87
The Minimum Element is: 5

***********
Total Steps= 62
*************

Process returned 0 (0x0)   execution time : 10.100 s
Press ENTER to continue.
```

CONCLUSION

- Detailed concept of MaxMin algorithm was studied successfully.
- MaxMin program were executed successfully.
- The step count for the MaxMin algorithm was obtained for different cases.