PROBLEM STATEMENT:

To find Max and Min element from the given array of elements using Divide and Conquer Approach.

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];}
    }
    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:=max1;
            if (min>min1) then min:=min1;
        }
}
```

PROGRAM CODE:

```
x=input("Enter the array giving space between elements:").split(" ")
mx=0
mn=0
def MaxMin(i,j,mx,mn):
  if(i==j):
    mx=int(x[i])
    mn=int(x[i])
    return (mx,mn)
  else:
    if(i==j-1):
      if(int(x[i])<int(x[j])):
        mx=int(x[j])
        mn=int(x[i])
        return (mx,mn)
      else:
```

```
            mx=int(x[i])
            mn=int(x[j])
            return (mx,mn)
        else:
            mid=(i+j)//2
            (mx,mn)=MaxMin(i,mid,mx,mn)
            (max1,min1)=MaxMin(mid+1,j,0,0)
            if(mx<max1):
                mx=max1
            if(mn>min1):
                mn=min1
            return(mx,mn)
(mx,mn)=MaxMin(0,len(x)-1,mx,mn)
print("Maximum value in array",mx)
print("Minimum value in array",mn)
```

OUTPUT:

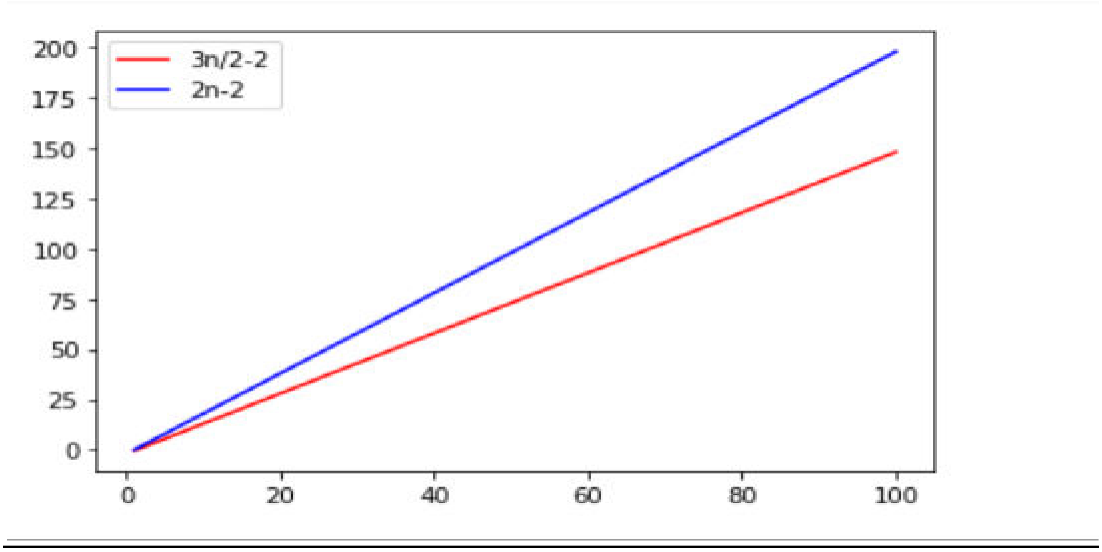
```
===== RESTART: C:/Users/pushp/AppData/Local/Programs/Python/Python38/df.py =====
Enter the array giving space between elements:20 30 7 12 57
Maximum value in array 57
Minimum value in array 7
>>>
========================================== RESTART: C:/Users/pushp/AppData/Local/Programs/Python/Python38
Enter the array giving space between elements:32
Maximum value in array 32
Minimum value in array 32

========================================== RESTART: C:/Users/pushp/AppData/Local/Programs/Python/Pyt
Enter the array giving space between elements:24 53 67 12 32
Maximum value in array 67
Minimum value in array 12
>>>
```

ANALYSIS:

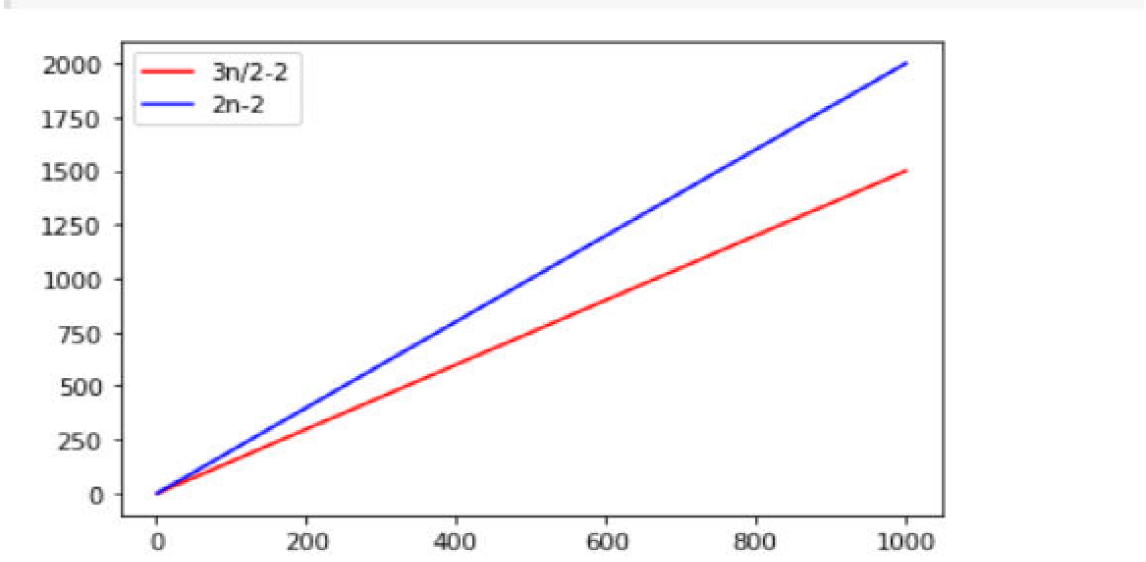
Straight Forward Method or Traditional method takes 2n-2 comparisons and This Method MaxMin takes 3n/2-2 which is less than 2n-2. If we compare the traditional method and MaxMin method, we can easily say MaxMin method takes less comparisons .

GRAPH:

Input Array Size(1-100)



Input Array Size(1-1000)



CONCLUSION:

As we increase the value of n, distance between 2n-2 and 3n/2-2 gets increasing which leads to more number of comparisons for Traditional method compared to this method for large size of array. So for large number of elements i.e for large array , It is better to use MaxMin method.