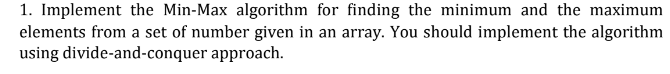
## **SUBMITTED BY:D.VEERA HARSHA VARDHAN REDDY ;210C2030061**



CODE:

#include <iostream>

using namespace std;

class dac

{

    public:

    int min1,min2,max1,max2,mid;

    void first\_half(int a[],int n)

    {

        min1=a[0];

        max1=a[0];

        for(int i=1;i<=mid;i++)

        {

          if(min1>a[i])

          {

              min1=a[i];

          }

          else if(max1<a[i])

          {

              max1=a[i];

          }

        }

    }

    void second\_half(int a[],int n)

    {

        min2=a[mid+ 1];

        max2=a[mid+ 1];

        for(int i=mid+ 2;i<n;i++)

        {

          if(min2>a[i])

          {

              min2=a[i];

          }

          else if(max2<a[i])

          {

              max2=a[i];

          }

        }

    }

    void final()

    {

        if(min1>min2)

        {

            cout<<"Minimum values is "<<" "<<min2<<endl;

        }

        else

        {

            cout<<"Minimum values is "<<" "<<min1<<endl;

        }

        if(max1>max2)

        {

            cout<<"Maximum values is "<<" "<<max1<<endl;

        }

        else

        {

            cout<<"Maximum values is "<<" "<<max2<<endl;

        }

    }

};

int main()

{

    int n,a[n],mid;

    cout<<"Enter the size of array"<<endl;

    cin>>n;

    cout<<"Enter the elements"<<endl;

    for(int i=0;i<n;i++)

    {

        cin>>a[i];

    }

    mid=n/2;

    dac obj;

    obj.first\_half(a,n);

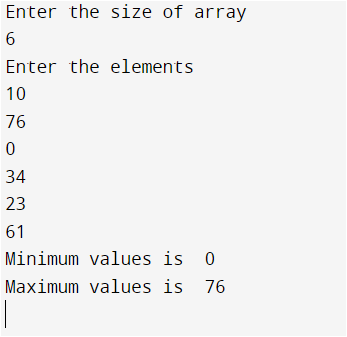
    obj.second\_half(a,n);

    obj.final();

    return 0;

}

OUTPUT:





CODE:

#include <iostream>

#include <vector>

using namespace std;

void swap(int \*a, int \*b)

{

    int temp = \*b;

    \*b = \*a;

    \*a = temp;

}

void heapify(vector<int> &hT, int i)

{

    int size = hT.size();

    int largest = i;

    int l = 2 \* i + 1;

    int r = 2 \* i + 2;

    if (l < size && hT[l] > hT[largest])

        largest = l;

    if (r < size && hT[r] > hT[largest])

        largest = r;

    if (largest != i)

    {

        swap(&hT[i], &hT[largest]);

        heapify(hT, largest);

    }

}

void insert(vector<int> &hT, int newNum)

{

    int size = hT.size();

    if (size == 0)

    {

        hT.push\_back(newNum);

    }

    else

    {

        hT.push\_back(newNum);

        for (int i = size / 2 - 1; i >= 0; i--)

        {

            heapify(hT, i);

        }

    }

}

void deleteNode(vector<int> &hT, int num)

{

    int size = hT.size();

    int i;

    for (i = 0; i < size; i++)

    {

        if (num == hT[i])

            break;

    }

    swap(&hT[i], &hT[size - 1]);

    hT.pop\_back();

    for (int i = size / 2 - 1; i >= 0; i--)

    {

        heapify(hT, i);

    }

}

void printArray(vector<int> &hT)

{

    for (int i = 0; i < hT.size(); ++i)

        cout << hT[i] << " ";

    cout << "\n";

}

int main()

{

    int n, a, b;

    cout << "Enter Number of Elements to be Inserted: ";

    cin >> n;

    vector<int> heapTree;

    for (int i = 1; i <= n; i++)

    {

        cout << "Enter Number[" << i << "]: ";

        cin >> a;

        insert(heapTree, a);

    }

    cout << "Max-Heap array: ";

    printArray(heapTree);

    cout << "Enter Element which should be Deleted: ";

    cin >> b;

    deleteNode(heapTree, b);

    cout << "After deleting an element: ";

    printArray(heapTree);

}

