TUSHAR PANJETA CS20B1098

1 Assignment 1, Due: 9/Aug
Problem: Finding MIN, MAX, Second MIN, Second MAX in an integer array
Present 3 different logic. NO logic should use sorting. For example, one of the logic is to divide the array into
two equal halves, find min/max/smin/smax for each, recursively update to get the final min/max/smin/smax.
Implementation using C++ classes and objects.
Some practice questions, not for submission: Given a number, list all its prime factors, Given m, n, find GCD(m, n), LCM(m, n) (you may think of more than one logic for each).

```
//Program to find MIN, MAX, Second MIN, Second MAX in an integer array
#include<iostream>
#include<ctime>
#include<cstdlib>
using namespace std;
#define SIZE 11
class Array {
    private:
         int max, min, smax, smin, start, end;

    public:
        int array[SIZE] ;
        //for creating an array
        void createarrayay() {
            srand(time(0));
            for(int i=0; i<SIZE; ++i) {
                array[i] = rand()%100 ;
                cout << array[i] <<" ";
            }
            cout << endl;
        }
        Array()
        {
            max=INT_MIN;
            min=INT_MAX;
            smax=INT_MIN;
            smin=INT_MAX;
        }
        void byLinear();
        void recusivelyDivideInTwoParts(int array[], int start, int end, int &min, int &max);
```

```
        void byComparisonInPairs();
};

void Array::byLinear() {

    //program for MAX and MIN
    for(int i=0; i<SIZE; ++i) {
        if(array[i] > max) {
            max = array[i];
        }
        if(array[i] < min) {
            min = array[i];
        }
    }

    //for finding Second MAX and Second MIN
    for(int i=0; i<SIZE; ++i) {
        if(array[i] != max) {
            if(array[i] > smax) {
                smax = array[i];
            }
        }
        if(array[i] != min) {
            if(array[i] < smin) {
                smin = array[i];
            }
        }

    }

    cout<<"\nMAX = "<<max<<"\nMIN = "<<min<<"\nSecond MAX =
"<<smax<<"\nSecond MIN = "<<smin<<endl;
}

void Array::recusivelyDivideInTwoParts(int array[], int start, int
end, int &min, int &max)
{
    // if the array contains only one element
    if (start == end)
    {
```

```
        if (max < array[start]) {
            max = array[start];
        }

        if (min > array[end]) {
            min = array[end];
        }

        return;
    }

    // if the array contains only two elements
    if (end - start == 1)
    {
        if (array[start] < array[end])
        {
            if (min > array[start]) {
                min = array[start];
            }

            if (max < array[end]) {
                max = array[end];
            }
        }
        else {
            if (min > array[end]) {
                min = array[end];
            }

            if (max < array[start]) {
                max = array[start];
            }
        }
        return;
    }

    int mid = (start + end) / 2;
    //recursion in first half
    recusivelyDivideInTwoParts(array, start, mid, min, max);
    //recursion in second half
```

```
    recusivelyDivideInTwoParts(array, mid + 1, end, min, max);
}

void Array::byComparisonInPairs() {
    int i=0, max=INT_MIN, min=INT_MAX, smax=INT_MIN, smin=INT_MAX;

    // for max and min
    if(SIZE % 2 == 1) {
        max = array[0];
        min = array[0];
        i=1;
    }
    else {
        if(array[0] < array[1]) {
            max = array[1];
            min = array[0];
        }
        else {
            max = array[0];
            min = array[1];
        }
        i=2;
    }

    while(i < SIZE) {
        if ( array[i] < array[i+1] ) //comparing continuous two
elements
        {
            if ( array[i] < min ) //comparing with min
                min = array[i];
            if ( array[i+1] > max ) //comparing with max
                max = array[i+1];
        }
        else
        {
            if ( array[i] > max )
                max = array[i];
            if ( array[i+1] < min )
                min = array[i+1];
        }
```

```
        i = i + 2;
    }

    // for smax and smin
    i=0;
    if(SIZE % 2 == 1) {
        if(array[0]==max || array[0]==min)
        {
            if(array[1]==max || array[1]==min)
            {
                smax = array[2];
                smin = array[2];
                i=3;
            }
            else
            {
                smax = array[1];
                smin = array[1];
                i=1;
            }
        }
        else
        {
            smax = array[0];
            smin = array[0];
            i=1;
        }
    }

    else {
        if(array[0]==max || array[0]==min)
        {
            if(array[1]==max || array[1]==min)
            {
                if(array[2] < array[3]) {
                   smax = array[3];
                   smin = array[2];
                }
                else {
                   smax = array[2];
```

```
                    smin = array[3];
                }
                i=4;
            }
            else{
                smax=array[1];
                smin=array[1];
                i=2;
            }
        }
        else
        {
            if(array[1]==max || array[1]==min)
            {
                smax = array[0];
                smin = array[0];
                i=2;

            }
            else {
                if(array[0] < array[1]) {
                    smax = array[1];
                    smin = array[0];
                }
                else {
                    smax = array[0];
                    smin = array[1];
                    i=2;
                }

            }

        }

    }

    while(i < SIZE) {
        if ( array[i] < array[i+1] )
        {
            if(array[i]==min)
```

```
            {
                if(array[i+1]<smin)
                    smin = array[i+1];
            }

            if ( array[i] < smin && min != array[i])
                smin = array[i];

            if(array[i+1]==max)
            {
                if(array[i]>smax)
                smax = array[i];
            }

            if ( array[i+1] > smax && array[i+1] != max )
                smax = array[i+1];

        }
        else
        {
            if(array[i]==max)
            {
                if(array[i+1]>smax)
                smax = array[i+1];
            }
            if ( array[i] > smax && array[i] != max )
                smax = array[i];

            if(array[i+1]==min)
            {
                if(array[i]<smin)
                smin = array[i];
            }

            if ( array[i+1] < smin && min != array[i+1])
                smin = array[i+1];
        }
        i = i + 2;
    }
```

```
    cout<<"\nMAX = "<<max<<"\nMIN = "<<min<<"\nSecondMAX =
"<<smax<<"\nSecondMIN = "<<smin<<endl;

}


int main() {
    int choice, max = INT_MIN, min=INT_MAX , smax = INT_MIN,
smin=INT_MAX ;
    Array array1;
    array1.createarrayay();

    cout<<"\n-----By Linear Searching-----"<<endl;
    array1.byLinear();

    cout<<"\n-----By dividing array in two parts-----"<<endl;

    array1.recusivelyDivideInTwoParts(array1.array, 0, SIZE-1,
min, max);

    cout << "\nMax = "<<max << endl;
    cout << "Min = "<<min << endl;

    int count=0, i;
    for( i=0 ; i<SIZE ; i++)
    {
        if(array1.array[i] == max || array1.array[i] == min)
        {
            count++;
        }
    }
    int n = SIZE - count;
    int arr[n];
    count=0;

    for( i=0 ; i<SIZE ; i++)
    {
        if(array1.array[i] == max || array1.array[i] == min)
        {
            continue;
```

```
        }
        else
        {
            arr[count] = array1.array[i];
            count++;
        }
    }

    array1.recusivelyDivideInTwoParts(arr, 0, n-1, smin, smax);
    cout << "SecondMax = "<<smax << endl;
    cout << "SecondMin = "<<smin << endl;

    cout<<"\n-----By Comparison in Pairs-----"<<endl;
    array1.byComparisonInPairs();

    return 0;
}
```