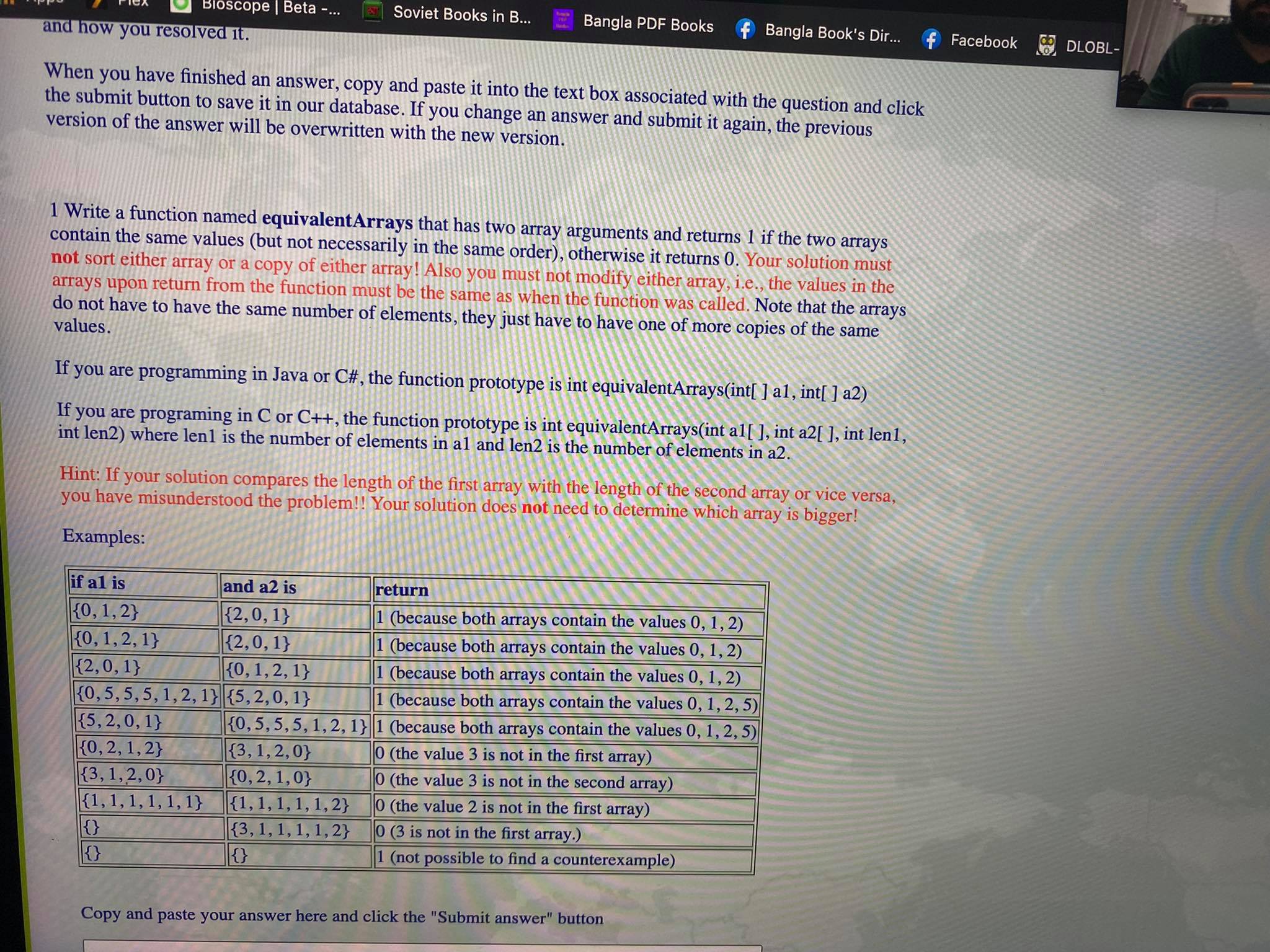
#moharishiSolveWithExplanations :



Brute Force :

package com.test;  
  
public class equivalentArraysFirstQuestion {  
  
 public static int equivalentArrays(int[]a1, int[]a2){  
  
 for(int i=0; i<a1.length; i++){  
 boolean matched = false;  
 for(int j=0; j<a2.length; j++){  
 if(a1[i]==a2[j]){  
 matched=true;  
 break; // so that no more checking is needed anymore for a1[i] with a2[j]  
  
 }  
 }  
 if(!matched){ // we kept this inside this loop of i cause if any of j not matched only then this will be needed and no further checking is required.  
 return 0;  
 }  
 }  
  
 for(int i=0; i<a2.length; i++){  
 boolean matched =false;  
 for(int j=0; j<a1.length; j++){  
 if(a2[i]==a1[j]){  
 matched=true;  
 break;  
 }  
  
 }  
 if(!matched){  
 return 0;  
 }  
 }  
 return 1;  
  
 }  
 public static void main(String[] args) {  
  
 System.*out*.println(*equivalentArrays*(new int[]{0,1,2}, new int[]{2,0,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{0,1,2,1}, new int[]{2,0,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{2,0,1}, new int[]{0,1,2,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{0,5,5,5,1,2,1}, new int[]{5,2,0,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{5,2,0,1}, new int[]{0,5,5,5,1,2,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{0,2,1,2}, new int[]{3,1,2,0}));  
 System.*out*.println(*equivalentArrays*(new int[]{3,1,2,0}, new int[]{0,2,1,0}));  
 System.*out*.println(*equivalentArrays*(new int[]{1,1,1,1,1,1}, new int[]{1,1,1,1,1,2}));  
 System.*out*.println(*equivalentArrays*(new int[]{}, new int[]{3,1,1,1,1,2}));  
 System.*out*.println(*equivalentArrays*(new int[]{}, new int[]{}));  
  
  
 }  
  
}

/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home/bin/java -javaagent:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/lib/idea\_rt.jar=54743:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/bin -Dfile.encoding=UTF-8 -classpath /Users/zayedabdullah/IdeaProjects/moharishiProblems/out/production/moharishiProblems com.test.equivalentArraysFirstQuestion

1

1

1

1

1

0

0

0

0

1

Process finished with exit code 0

package com.test;  
import java.util.HashMap;  
  
public class equivalentArraysFirstQuestionOptimized {  
  
 public static int equivalentArrays(int[]a1, int[]a2){  
  
 HashMap<Integer,Boolean>memory1 = new HashMap<Integer,Boolean>();  
 HashMap<Integer,Boolean>memory2 = new HashMap<Integer,Boolean>();  
 int counter1=0,counter2=0;  
  
 for(int i=0; i<a1.length; i++){  
 if(!memory1.containsKey(a1[i])){  
 counter1++;  
 memory1.put(a1[i],true);  
 }  
 }  
 for(int i=0; i<a2.length; i++){  
 if(!memory1.containsKey(a2[i])){  
 return 0;  
 }  
 else {  
 if(!memory2.containsKey(a2[i])){  
  
 counter2++;  
 }  
 }  
 memory2.put(a2[i],true);  
  
 }  
 if(counter1!=counter2){  
 return 0;  
 }  
 return 1;  
 }  
 public static void main(String[] args) {  
  
 System.*out*.println(*equivalentArrays*(new int[]{0,1,2}, new int[]{2,0,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{0,1,2,1}, new int[]{2,0,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{2,0,1}, new int[]{0,1,2,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{0,5,5,5,1,2,1}, new int[]{5,2,0,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{5,2,0,1}, new int[]{0,5,5,5,1,2,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{0,2,1,2}, new int[]{3,1,2,0}));  
 System.*out*.println(*equivalentArrays*(new int[]{3,1,2,0}, new int[]{0,2,1,0}));  
 System.*out*.println(*equivalentArrays*(new int[]{1,1,1,1,1,1}, new int[]{1,1,1,1,1,2}));  
 System.*out*.println(*equivalentArrays*(new int[]{}, new int[]{3,1,1,1,1,2}));  
 System.*out*.println(*equivalentArrays*(new int[]{}, new int[]{}));  
  
  
 }  
  
}

/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home/bin/java -javaagent:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/lib/idea\_rt.jar=55108:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/bin -Dfile.encoding=UTF-8 -classpath /Users/zayedabdullah/IdeaProjects/moharishiProblems/out/production/moharishiProblems com.test.equivalentArraysFirstQuestionBruteForce

1

1

1

1

1

0

0

0

0

1

Process finished with exit code 0

package com.test;  
  
public class equivalentArrayThirdWay {  
  
 public static int equivalentArrays(int[]a1, int[]a2){  
 int[] flag = new int[a2.length];  
 for(int i=0; i<flag.length; i++){  
 flag[i]=0;  
 }  
  
 for(int i=0; i<a1.length; i++){  
 int temp=0;  
 for(int j=0; j<a2.length; j++){  
 if(a1[i]==a2[j]){  
 flag[j]=1;  
 temp=1;  
 }  
 }  
 if(temp==0){  
 return 0;  
 }  
 }  
 for( int i=0; i<flag.length; i++){  
 if(flag[i]==0){  
 return 0;  
 }  
 }  
 return 1;  
  
 }  
 public static void main(String[] args) {  
  
 System.*out*.println(*equivalentArrays*(new int[]{0,1,2}, new int[]{2,0,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{0,1,2,1}, new int[]{2,0,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{2,0,1}, new int[]{0,1,2,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{0,5,5,5,1,2,1}, new int[]{5,2,0,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{5,2,0,1}, new int[]{0,5,5,5,1,2,1}));  
 System.*out*.println(*equivalentArrays*(new int[]{0,2,1,2}, new int[]{3,1,2,0}));  
 System.*out*.println(*equivalentArrays*(new int[]{3,1,2,0}, new int[]{0,2,1,0}));  
 System.*out*.println(*equivalentArrays*(new int[]{1,1,1,1,1,1}, new int[]{1,1,1,1,1,2}));  
 System.*out*.println(*equivalentArrays*(new int[]{}, new int[]{3,1,1,1,1,2}));  
 System.*out*.println(*equivalentArrays*(new int[]{}, new int[]{}));  
  
 }  
  
}

/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home/bin/java -javaagent:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/lib/idea\_rt.jar=56331:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/bin -Dfile.encoding=UTF-8 -classpath /Users/zayedabdullah/IdeaProjects/moharishiProblems/out/production/moharishiProblems com.test.equivalentArrayThirdWay

1

1

1

1

1

0

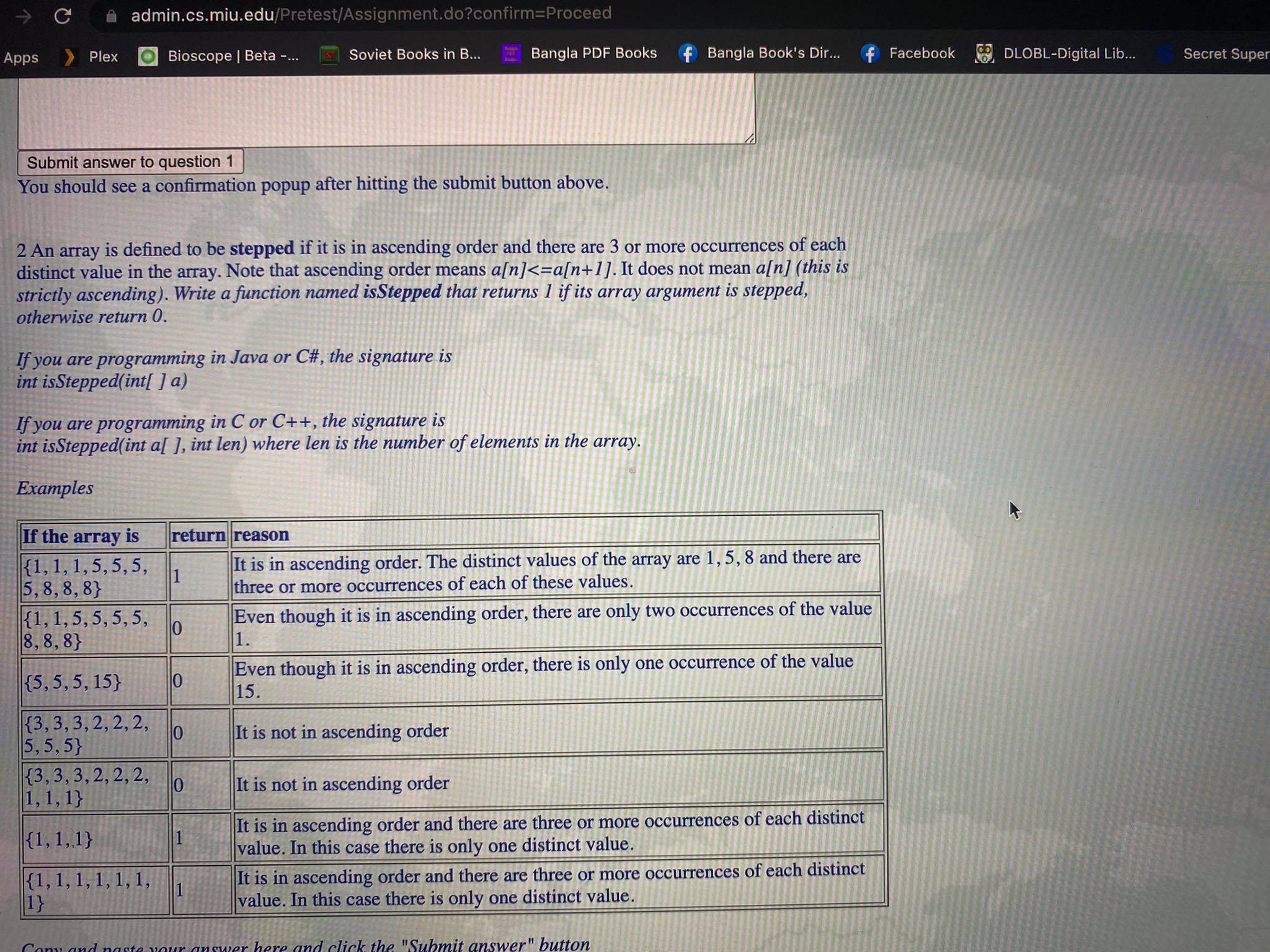
0

0

0

1

Process finished with exit code 0



package com.test;  
  
public class isSteppedArraySecondQuestion {  
  
 public static int isStepped(int[] a){  
 int firstElement =a[0];  
 int counter =1;  
  
 for(int i=1; i<a.length; i++){  
 if(firstElement==a[i]){  
 counter++;  
 firstElement=a[i];  
 }  
 else if(counter>=3 && firstElement<a[i]){  
 counter=1;  
 firstElement=a[i];  
 }  
 else {  
 return 0;  
 }  
 }  
 if(counter<3){  
 return 0;  
 }  
 return 1;  
  
 }  
  
 public static void main(String[] args) {  
  
 System.*out*.println(*isStepped*(new int[]{1,1,1,5,5,5,5,8,8,8}));  
 System.*out*.println(*isStepped*(new int[]{1,1,5,5,5,5,5,8,8,8}));  
 System.*out*.println(*isStepped*(new int[]{5,5,5,5,15}));  
 System.*out*.println(*isStepped*(new int[]{3,3,3,2,2,5,5,5,5,5}));  
 System.*out*.println(*isStepped*(new int[]{3,3,3,2,2,2,1,1,1}));  
 System.*out*.println(*isStepped*(new int[]{1,1,1,1}));  
 System.*out*.println(*isStepped*(new int[]{1,1,1,1,1,1,1}));  
  
  
 }  
}

/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home/bin/java -javaagent:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/lib/idea\_rt.jar=56410:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/bin -Dfile.encoding=UTF-8 -classpath /Users/zayedabdullah/IdeaProjects/moharishiProblems/out/production/moharishiProblems com.test.isSteppedArraySecondQuestion

1

0

0

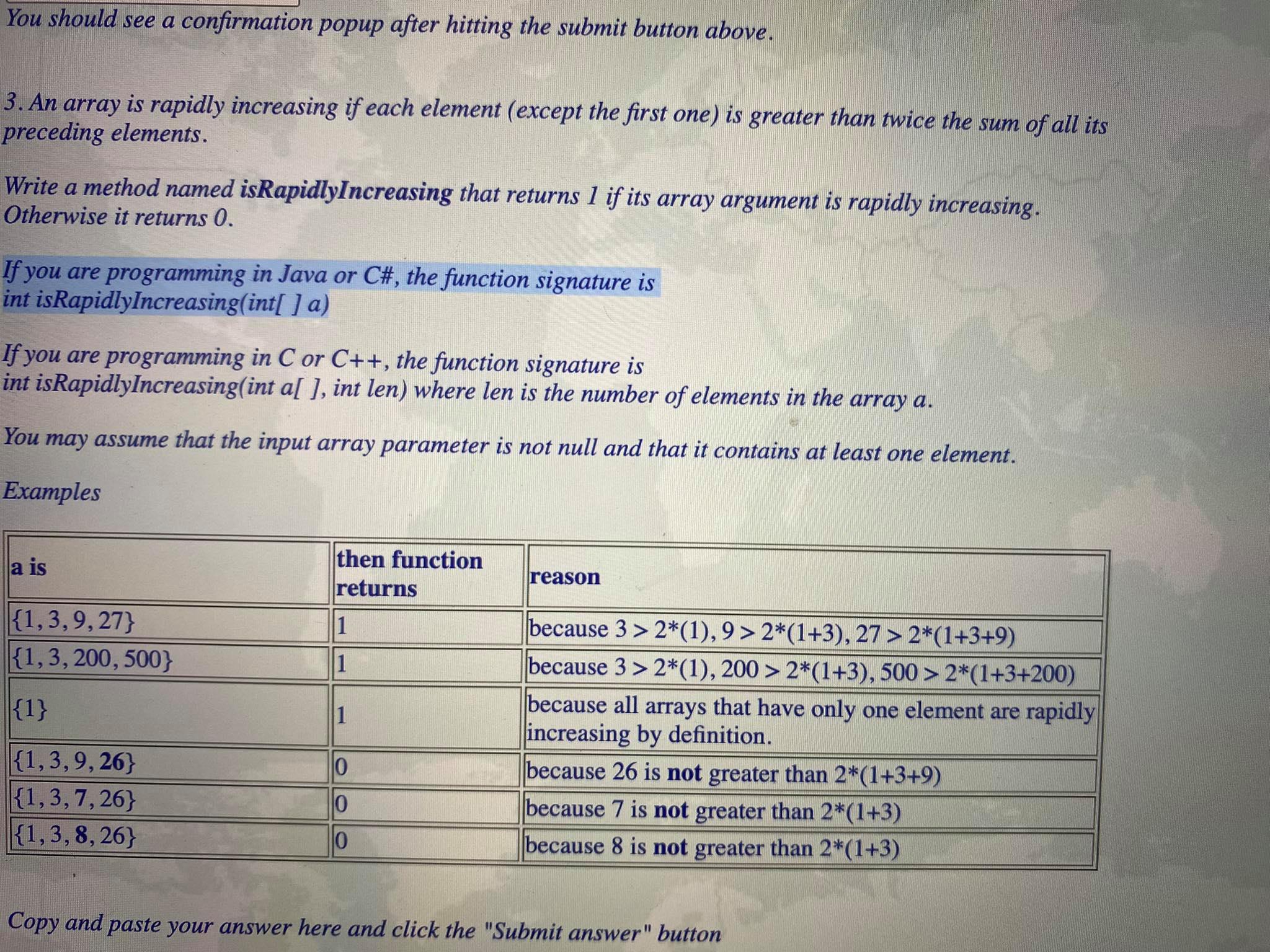
0

0

1

1

Process finished with exit code 0



package com.test;  
  
public class isRapidlyIncreasing {  
  
 public static int isRapidlyIncreasing(int[] a){  
 int firstElement =a[0];  
 int sum=0;  
  
 for(int i=1; i<a.length; i++){  
 if(a[i]>2\*(firstElement+sum)){  
 sum=sum+firstElement;  
 firstElement=a[i];  
 }  
 else {  
 return 0;  
 }  
 }  
 return 1;  
  
 }  
 public static void main(String[] args) {  
  
 System.*out*.println(*isRapidlyIncreasing*( new int[]{1,3,9,27}));  
 System.*out*.println(*isRapidlyIncreasing*( new int[]{1,3,200,500}));  
 System.*out*.println(*isRapidlyIncreasing*( new int[]{1}));  
 System.*out*.println(*isRapidlyIncreasing*( new int[]{1,3,9,26}));  
 System.*out*.println(*isRapidlyIncreasing*( new int[]{1,3,7,26}));  
 System.*out*.println(*isRapidlyIncreasing*( new int[]{1,3,8,26}));  
  
  
 }  
}

/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home/bin/java -javaagent:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/lib/idea\_rt.jar=56559:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/bin -Dfile.encoding=UTF-8 -classpath /Users/zayedabdullah/IdeaProjects/moharishiProblems/out/production/moharishiProblems com.test.isRapidlyIncreasing

1

1

1

0

0

0

Process finished with exit code 0

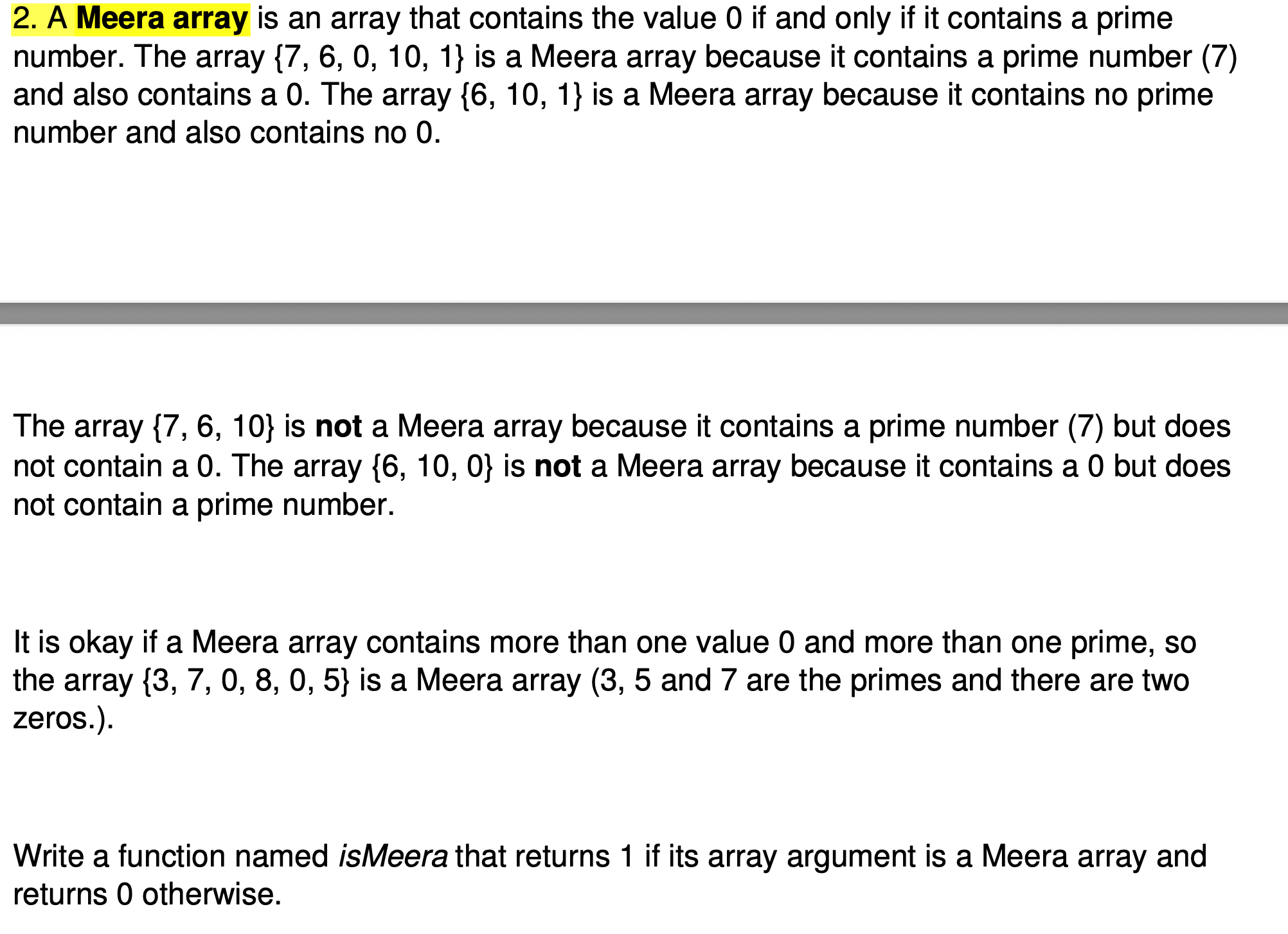
Sieve Of Eratosthenes Algorithm For Prime Numbers In A Range :

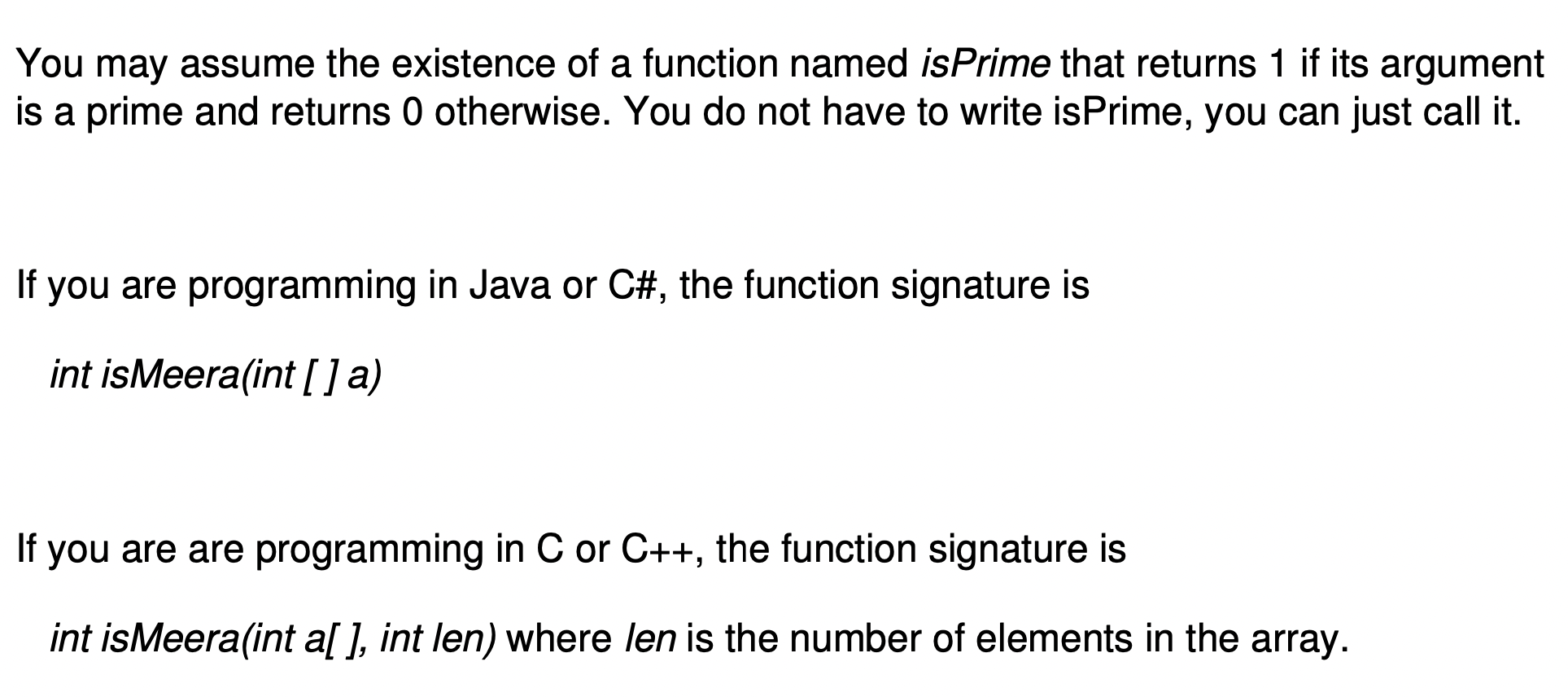
package com.test;  
  
import java.util.Arrays;  
  
public class sieveOfEratosthenes {  
  
 public static boolean[] sieveOfEratosthenes(int n){  
 boolean[] isPrime = new boolean[n+1];// to align numbers with index  
 Arrays.*fill*(isPrime, true);  
 isPrime[0]=false;  
 isPrime[1]=false;  
  
 for(int i=2; i\*i<=n; i++){  
 for(int j= 2\*i; j<=n; j+=i){  
 isPrime[j]=false;  
 }  
 }  
 return isPrime;  
  
 }  
  
 public static void printPrime (boolean[] a){  
  
 for(int i=0; i<a.length; i++){  
 if(a[i]==true){  
 System.*out*.print(i+" ,"+ " ");  
 }  
 }  
  
 }  
  
  
  
  
 public static void main(String[] args) {  
 boolean[] rangedNumber = *sieveOfEratosthenes*(12);  
 *printPrime*(rangedNumber);  
 }  
}

/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home/bin/java -javaagent:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/lib/idea\_rt.jar=58208:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/bin -Dfile.encoding=UTF-8 -classpath /Users/zayedabdullah/IdeaProjects/moharishiProblems/out/production/moharishiProblems com.test.sieveOfEratosthenes

2 , 3 , 5 , 7 , 11 ,

Process finished with exit code 0





package com.test;  
  
public class meeraArrayWithPrimeNumberAndZero {  
  
 public static int isMeera(int[] a){  
 int isZeroThere =0;  
 int isPrimeThere =0;  
  
 for(int i=0; i<a.length; i++){  
 if(a[i]==0){  
 isZeroThere++;  
 }  
 else {  
 for(int j=2; j\*j<a[i]; j++){ // i<root n  
 if(a[i]%j!=0){  
 isPrimeThere++;  
 break;  
 }  
 }  
 }  
 }  
 if((isPrimeThere>0 &&isZeroThere>0) ||( isPrimeThere==0 && isZeroThere==0)){  
 return 1;  
 }  
 return 0;  
  
 }  
  
  
 public static void main(String[] args) {  
  
 System.*out*.println(*isMeera*(new int[]{7,6,0,10,1}));  
 System.*out*.println(*isMeera*(new int[]{6,10,1}));  
 System.*out*.println(*isMeera*(new int[]{7,6,10}));  
 System.*out*.println(*isMeera*(new int[]{6,10,0}));  
 System.*out*.println(*isMeera*(new int[]{3,7,0,8,0,5}));  
  
 }  
}

/Library/Java/JavaVirtualMachines/jdk-17.0.1.jdk/Contents/Home/bin/java -javaagent:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/lib/idea\_rt.jar=64105:/Users/zayedabdullah/Library/Application Support/JetBrains/Toolbox/apps/IDEA-C/ch-0/213.6461.79/IntelliJ IDEA CE.app/Contents/bin -Dfile.encoding=UTF-8 -classpath /Users/zayedabdullah/IdeaProjects/moharishiProblems/out/production/moharishiProblems com.test.meeraArrayWithPrimeNumberAndZero

1

1

0

0

1

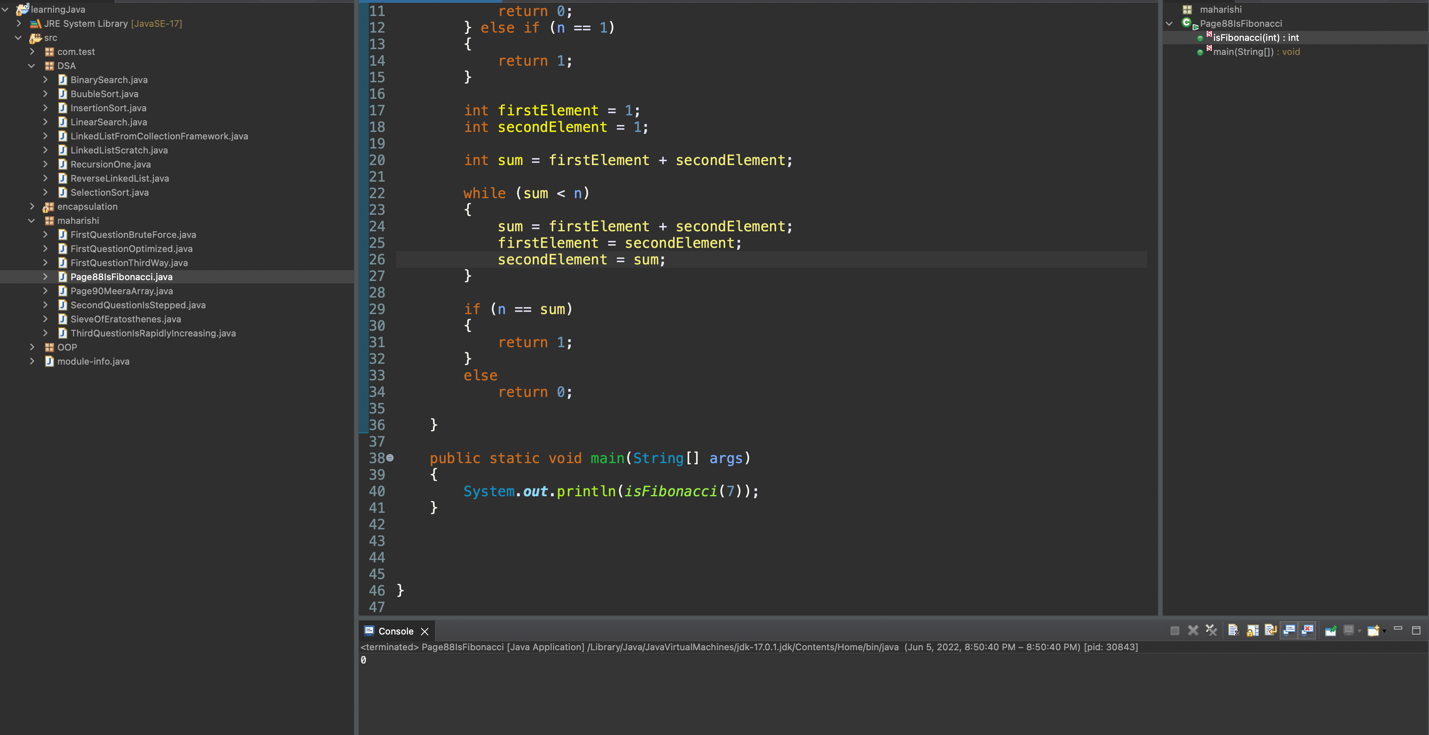
Process finished with exit code 0

1. A **Fibonacci number** is a number in the sequence 1, 1, 2, 3, 5, 8, 13, 21,.... Note that first two Fibonacci numbers are 1 and any Fibonacci number other than the first two is the sum ofthe previous two Fibonacci numbers. For example, 2 = 1 + 1, 3 = 2 + 1, 5 = 3 + 2 and so on.

Write a function named *isFibonacci* that returns 1 if its integer argument is a Fibonacci number, otherwise it returns 0.

The signature of the function is

*int isFibonacci (int n)*

**

package maharishi;

public class Page88IsFibonacci

{

public static int isFibonacci(int n)

{

if (n == 0)

{

return 0;

} else if (n == 1)

{

return 1;

}

int firstElement = 1;

int secondElement = 1;

int sum = firstElement + secondElement;

while (sum < n)

{

sum = firstElement + secondElement;

firstElement = secondElement;

secondElement = sum;

}

if (n == sum)

{

return 1;

}

else

return 0;

}

public static void main(String[] args)

{

System.out.println(isFibonacci(7));

}

}

}

**Question 3.** An array is said to be hollow if it contains 3 or more zeros in the middle that are preceded and followed by the same number of non-zero elements. Write a function named isHollow that accepts an integer array and returns 1 if it is a hollow array, otherwise it returns 0. The function signature is

int isHollow(int[ ] a).

Examples: isHollow({1,2,4,0,0,0,3,4,5}) returns true. isHollow ({1,2,0,0,0,3,4,5}) returns false. : isHollow ({1,2,4,9, 0,0,0,3,4, 5}) returns false. isHollow ({1,2, 0,0, 3,4}) returns false.

package maharishi;

public class Page88isHollow

{

public static int isHollow(int[] a)

{

int zeroCount = 0;

int count = 0;

for (int i = 0; i < a.length; i++)

{

if (a[i] != 0)

{

count++;

} else

{

zeroCount++;

}

}

if (zeroCount >= 3 && count % 2 == 0)

{

if (zeroCount == count / 2)

{

return 1;

}

}

return 0;

}

public static void main(String[] args)

{

System.out.println(isHollow(new int[]

{ 1, 2, 3, 0, 0, 0, 3, 4, 5 }));

}

}

**Question 1.** Write a function fill with signature

int[] fill(int[] arr, int k, int n)

which does the following: It returns an integer array arr2 of length n whose first k elements are the same as the first k elements of arr, and whose remaining elements consist of repeating blocks of the first k elements. You can assume array arr has at least k elements. The function should return null if either k or n is not positive.

Examples: fill({1,2,3,5, 9, 12,-2,-1}, 3, 10) returns {1,2,3,1,2,3,1,2,3,1}. Fill({4, 2, -3, 12}, 1, 5) returns {4, 4, 4, 4, 4}. fill({2, 6, 9, 0, -3}, 0, 4) returns null.

package maharishi;

import java.util.Arrays;

public class Page88intFill

{

public static int[] fill(int[] arr, int k, int n)

{

if (k <= 0 || n <= 0)

{

return null;

}

int[] arr2 = new int[n];

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

{

arr2[i] = arr[i];

}

int i = 0;

while (k < n)

{

arr2[k] = arr2[i];

i++;

k++;

}

if (arr2.length == n)

{

return arr2;

} else

{

return null;

}

}

public static void main(String[] args)

{

System.out.println(Arrays.toString(fill(new int[]

{ 1, 2, 3, 5, 9, 12, -2, -1 }, 3, 10)));

System.out.println(Arrays.toString(fill(new int[]

{ 4, 2, -3, 12 }, 1, 5)));

System.out.println(Arrays.toString(fill(new int[]

{ 2, 6, 9, 0, -3 }, 0, 4)));

}

}

Output

**[1, 2, 3, 1, 2, 3, 1, 2, 3, 1]**

**[4, 4, 4, 4, 4]**

**null**

Reverse an array using swap

package maharishi;

import java.util.Arrays;

public class ReverseAnArray

{

public static int[] reverseArray(int[] arr)

{

int start = 0;

int end = arr.length - 1;

int temp;

while (start < end)

{

temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

start++;

end--;

}

return arr;

}

public static void main(String[] args)

{

System.out.println(Arrays.toString(reverseArray(new int[]

{ 1, 2, 3, 4, 5 })));

}

}

Reverse array using recursion

package maharishi;

import java.util.Arrays;

public class ReverseAnArrayWithRecursion

{

public static void reverseArrayWithRecursion(int[] arr, int start, int end)

{

int temp;

if (start >= end)

{

return;

}

temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

reverseArrayWithRecursion(arr, start + 1, end - 1);

}

public static void main(String[] args)

{

int[] arrayToReverse =

{ 1, 2, 3, 4, 5, 6 };

reverseArrayWithRecursion(arrayToReverse, 0, 5);

System.out.println(Arrays.toString(arrayToReverse));

}

}

1. Write a function named **minDistance** that returns the smallest distance between two factors of a number. For example, consider 13013 = 1\*7\*11\*13. Its factors are 1, 7, 11, 13 and 13013. **minDistance**(13013) would return 2 because the smallest distance between any two factors is 2 (13 ­ 11 = 2). As another example, **minDistance** (8) would return 1 because the factors of 8 are 1, 2, 4, 8 and the smallest distance between any two factors is 1 (2 – 1 = 1).

The function signature is int **minDistance**(int n)

package maharishi;

public class Page86minDistance

{

public static int minDistance(int n)

{

int minDistance = Integer.MAX\_VALUE;

int previousFactor = 1;

for (int i = 2; i < n; i++)

{

if (n % i == 0)

{

if ((i - previousFactor) < minDistance)

{

minDistance = i - previousFactor;

previousFactor = i;

}

}

}

return minDistance;

}

public static void main(String[] args)

{

System.out.println(minDistance(9));

}

}

Get minimum and maximum number of an array at the same time

Linear Method

package maharishi;

public class MinMaxBothLinear

{

// class Pair is used to return two values from getMinMax

static class Pair

{

int min;

int max;

}

static Pair getMinMax(int[] arr, int n)

{

Pair minmax = new Pair();

int i;

// if there is only one item then return this as min max both ;

if (n == 1)

{

minmax.max = arr[0];

minmax.min = arr[0];

return minmax;

}

// if there is more than one elements, then initialize min and max

if (arr[0] > arr[1])

{

minmax.max = arr[0];

minmax.min = arr[1];

} else

{

minmax.max = arr[1];

minmax.min = arr[0];

}

for (i = 2; i < n; i++)

{

if (arr[i] > minmax.max)

{

minmax.max = arr[i];

} else if (arr[i] < minmax.min)

{

minmax.min = arr[i];

}

}

return minmax;

}

public static void main(String[] args)

{

// driver program to test above function

Pair minmax = getMinMax(new int[]

{ 1000, 11, 445, 1, 330, 3000 }, 6);

System.out.printf("\nMinimum element is %d", minmax.min);

System.out.printf("\nMaximum element is %d", minmax.max);

}

}

Binary Search Method / Divide & Conquer Method

package maharishi;

public class MinMaxBothTournamentMethod

{

// binary search i guess

static class Pair

{

int min;

int max;

}

static Pair getMinMax(int[] arr, int low, int high)

{

Pair minmax = new Pair();

Pair mml = new Pair();

Pair mmr = new Pair();

int mid;

// if ther is only one element

if (low == high)

{

minmax.max = arr[low];

minmax.min = arr[low];

return minmax;

}

// if there are two elements

if (high == low + 1)

{

if (arr[low] > arr[high])

{

minmax.max = arr[low];

minmax.min = arr[high];

} else

{

minmax.max = arr[high];

minmax.min = arr[low];

}

}

// if there are more than two elements

mid = (low + high) / 2;

mml = getMinMax(arr, low, mid);

mmr = getMinMax(arr, mid + 1, high);

// compare minimum of two parts

if (mml.min < mmr.min)

{

minmax.min = mml.min;

} else

{

minmax.min = mmr.min;

}

if (mml.max > mmr.max)

{

minmax.max = mml.max;

} else

{

minmax.max = mmr.max;

}

return minmax;

}

public static void main(String[] args)

{

// Driver program to test above function

int[] arr =

{ 1000, 11, 445, 1, 330, 3000 };

int arr\_size = 6;

Pair minmax = getMinMax(arr, 0, arr\_size - 1);

System.out.printf("\nMinimum element is %d", minmax.min);

System.out.printf("\nMaximum element is %d", minmax.max);

}

}

Pair checking / Double Checking Method :

package maharishi;

public class MinMaxBothCompareInPairsMethod

{

// class Pair is used to return two values from getMinMax

static class Pair

{

int min;

int max;

}

static Pair getMinMax(int[] arr, int n)

{

Pair minmax = new Pair();

int i;

// if there is even number of elements, then initialize the first two elements

// as min and max

if (n % 2 == 0)

{

if (arr[0] > arr[1])

{

minmax.max = arr[0];

minmax.min = arr[1];

} else

{

minmax.min = arr[0];

minmax.max = arr[1];

}

i = 2; // set the starting index for loop

}

// if therer is odd number of elements in the array then set the first element

// as minimum and maximum

else

{

minmax.min = arr[0];

minmax.max = arr[0];

i = 1; // set the starting index for the loop

}

// in while loop , pick elements in pair and compare the pair with max and min

// so far

while (i < n - 1)

{

if (arr[i] > arr[i + 1])

{

if (arr[i] > minmax.max)

{

minmax.max = arr[i];

}

if (arr[i + 1] < minmax.min)

{

minmax.min = arr[i + 1];

}

} else

{

if (arr[i + 1] > minmax.max)

{

minmax.max = arr[i + 1];

}

if (arr[i] < minmax.min)

{

minmax.min = arr[i];

}

}

i += 2; // increment the loop by two as two elements are processed in loop

}

return minmax;

}

public static void main(String[] args)

{

// Driver program to test above function

int[] arr =

{ 1000, 11, 445, 1, 330, 3000 };

int arr\_size = 6;

Pair minmax = getMinMax(arr, arr\_size);

System.out.printf("\nMinimum element is %d", minmax.min);

System.out.printf("\nMaximum element is %d", minmax.max);

}

}

1. A **Meera number** is a number such that the number of nontrivial factors is a factor of the number. For example, 6 is a Meera number because 6 has two nontrivial factors : 2 and 3. (A nontrivial factor is a factor other than 1 and the number). Thus 6 has two nontrivial factors. Now, 2 is a factor of 6. Thus the number of nontrivial factors is a factor of 6. Hence 6 is a Meera number. Another Meera number is 30 because 30 has 2, 3, 5, 6, 10, 15 as nontrivial factors. Thus 30 has 6 nontrivial factors. Note that 6 is a factor of 30. So 30 is a Meera Number. However 21 is **not** a Meera number. The nontrivial factors of 21 are 3 and 7. Thus the number of nontrivial factors is 2. Note that 2 is not a factor of 21. Therefore, 21 is not a Meera number.

Write a function named *isMeera* that returns 1 if its integer argument is a Meera number, otherwise it returns 0.

The signature of the function is int isMeera(int n)

package maharishi;

public class Page84MeeraNumber

{

static int isMeera(int n)

{

int count = 0;

for (int i = 2; i < n; i++)

{

if (n % i == 0)

{

count++;

}

}

if (count > 1 && n % count == 0)

{

return 1;

}

else

return 0;

}

public static void main(String[] args)

{

System.out.println(isMeera(6));

}

}

An array is defined to be ***odd-heavy*** if it contains at least one odd element and every odd element is greater than every even element. So {11, 4, 9, 2, 8} is odd-heavy because the two odd elements (11 and 9) are greater than all the even elements. And {11, 4, 9, 2, 3, 10} is not odd-heavy because the even element 10 is greater than the odd element 9. Write a function called *isOddHeavy* that accepts an integer array and returns 1 if the array is odd-heavy; otherwise it returns 0. Some other examples: {1} is odd-heavy, {2} is not odd-heavy, {1, 1, 1, 1} is odd-heavy, {2, 4, 6, 8, 11} is odd-heavy, {-2, -4, -6, -8, -11} is not odd-heavy.

If you are programming in Java or C#, the function signature is

*int isOddHeavy(int[ ] a)*

If you are programming in C or C++, the function signature is

*int isOddHeavy(int a[ ], int len)*where *len* is the number of elements in the array.

public class Page85OddHeavy

{

static int isOddHeavy(int[] a)

{

boolean hasOdd = false;

for (int i = 0; i < a.length; i++)

{

if (a[i] % 2 == 1)

{

hasOdd = true;

for (int j = 0; j < a.length; j++)

{

if (a[j] % 2 == 0)

{

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

{

return 0;

}

}

}

}

}

return hasOdd ? 1 : 0;

}

public static void main(String[] args)

{

System.out.println(isOddHeavy(new int[]

{ 11, 4, 9, 2, 8 }));

System.out.println(isOddHeavy(new int[]

{ 11, 4, 9, 2, 3, 10 }));

System.out.println(isOddHeavy(new int[]

{ 1, 1, 1, 1, 1, 1 }));

System.out.println(isOddHeavy(new int[]

{ 2, 4, 6, 8, 11 }));

System.out.println(isOddHeavy(new int[]

{ -2, -4, -6, -8, -11 }));

System.out.println(isOddHeavy(new int[]

{ 1 }));

System.out.println(isOddHeavy(new int[]

{ 2 }));

}

}

Output :

1

0

1

1

0

1

0