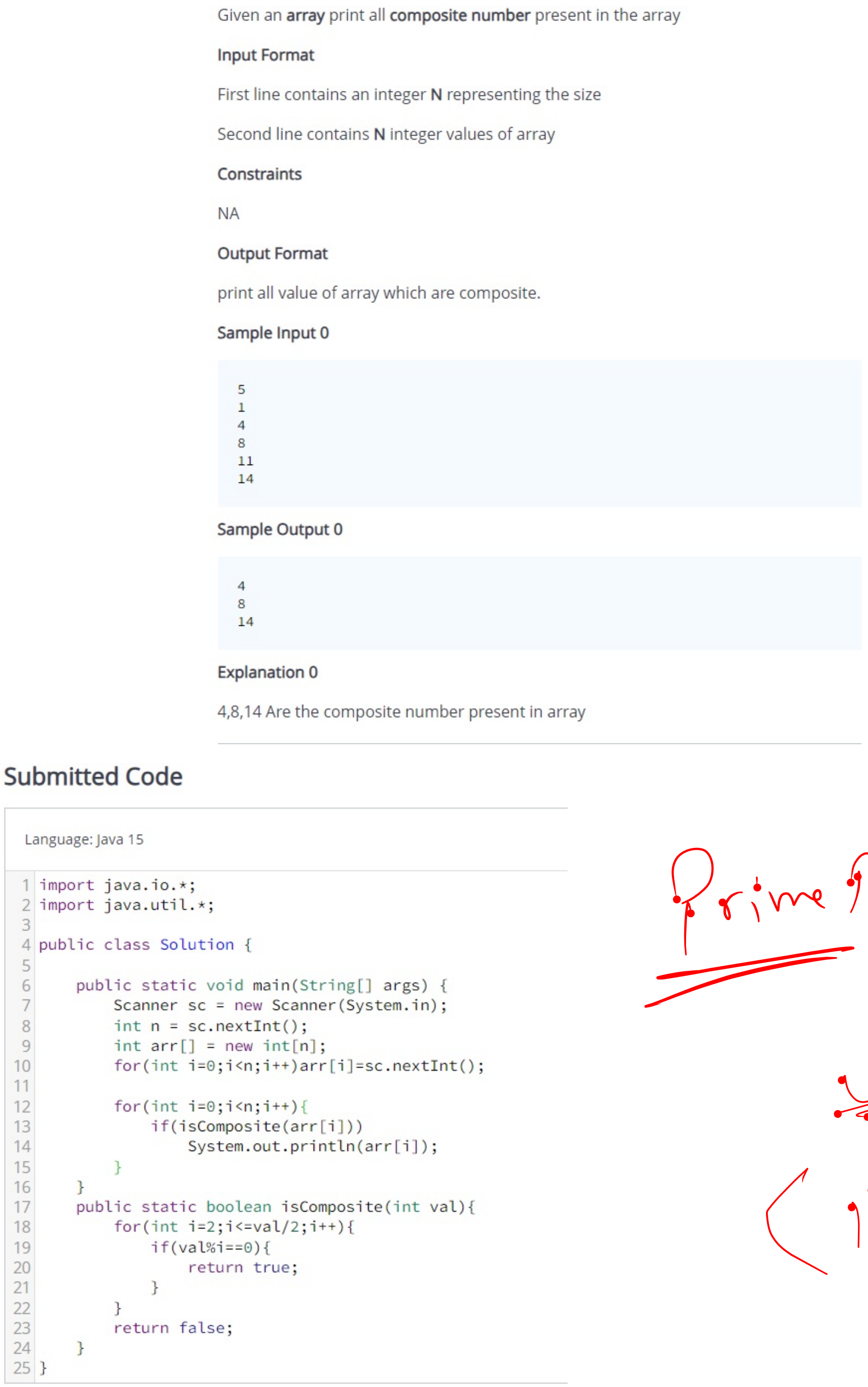


Given an **array** print all **composite number** present in the array

Input Format

First line contains an integer **N** representing the size

Second line contains **N** integer values of array

Constraints

NA

Output Format

print all value of array which are composite.

Sample Input 0

```
5
1
4
8
11
14
```

Sample Output 0

```
4
8
14
```

Explanation 0

4,8,14 Are the composite number present in array

Submitted Code

Language: Java 15

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int arr[] = new int[n];
        for(int i=0;i<n;i++)arr[i]=sc.nextInt();

        for(int i=0;i<n;i++){
            if(isComposite(arr[i]))
                System.out.println(arr[i]);
        }
    }
    public static boolean isComposite(int val){
        for(int i=2;i<=val/2;i++){
            if(val%i==0){
                return true;
            }
        }
        return false;
    }
}
```

N value

[1, 4, 8, 11, 14]

4 % 4
4 % 2
4 % 1

8 % 8
8 % 4
8 % 2
8 % 1

14 % 14
14 % 7
14 % 2
14 % 1

Prime Number

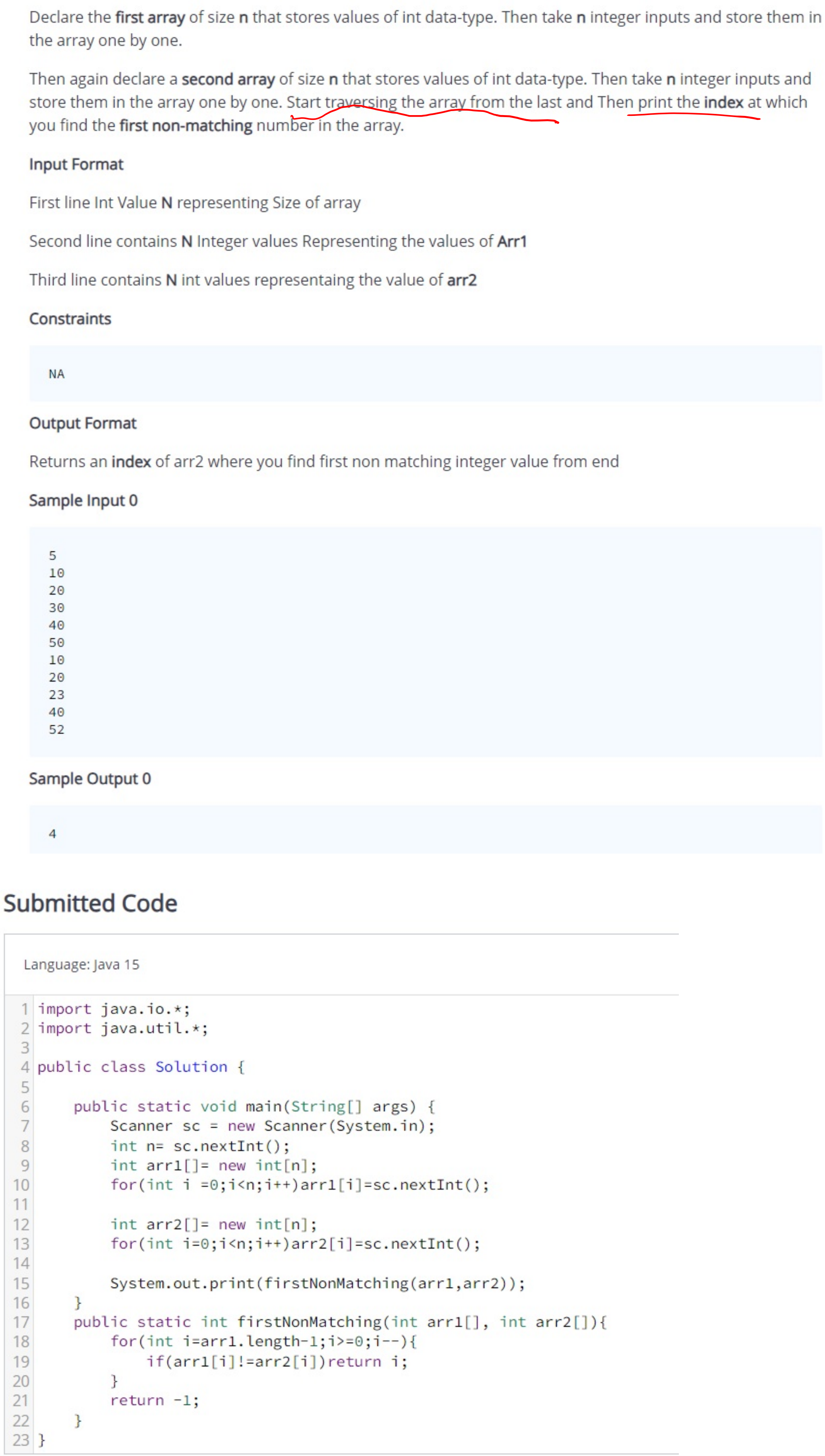
4 8 16

(i = 2 ; i <= n/2 ; i++){

if(n % i == 0) return true

}

return false



Declare the **first array** of size **n** that stores values of int data-type. Then take **n** integer inputs and store them in the array one by one.

Then again declare a **second array** of size **n** that stores values of int data-type. Then take **n** integer inputs and store them in the array one by one. Start traversing the array from the last and Then print the **index** at which you find the **first non-matching** number in the array.

Input Format

First line Int Value **N** representing Size of array

Second line contains **N** Integer values Representing the values of **Arr1**

Third line contains **N** int values representaing the value of **arr2**

Constraints

```
NA
```

Output Format

Returns an **index** of arr2 where you find first non matching integer value from end

Sample Input 0

```
5
10
20
30
40
50
10
20
23
40
52
```

Sample Output 0

```
4
```

Submitted Code

Language: Java 15

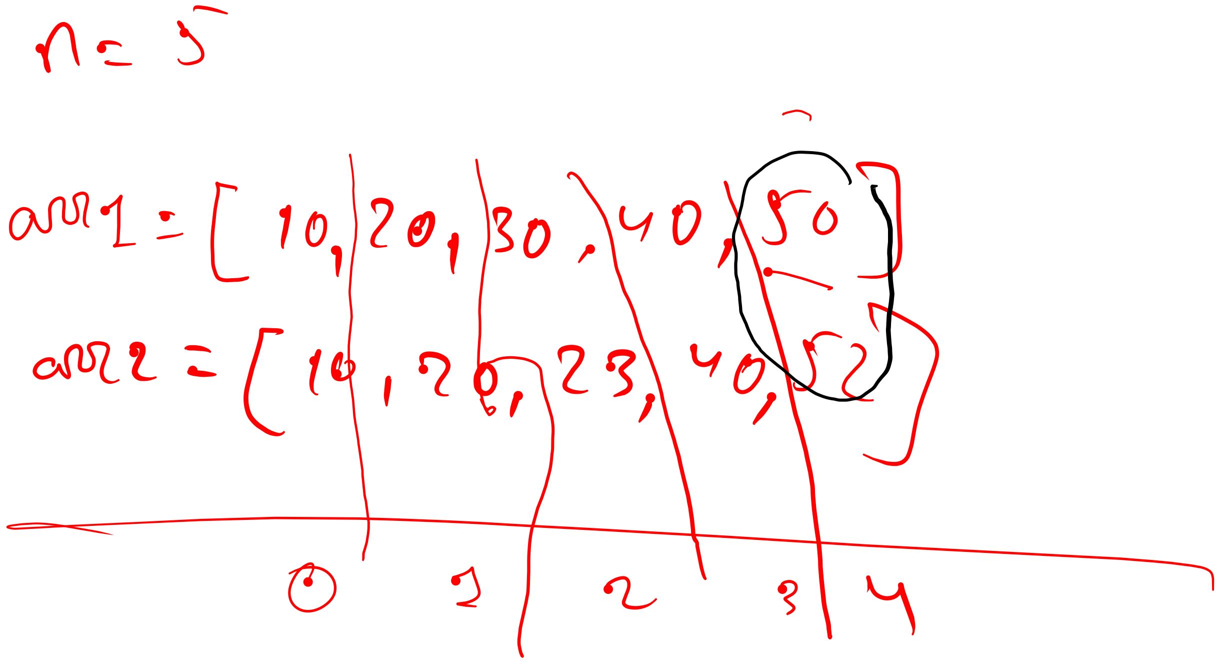
```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n= sc.nextInt();
        int arr1[]= new int[n];
        for(int i =0;i<n;i++)arr1[i]=sc.nextInt();

        int arr2[]= new int[n];
        for(int i=0;i<n;i++)arr2[i]=sc.nextInt();

        System.out.print(firstNonMatching(arr1,arr2));
    }
    public static int firstNonMatching(int arr1[], int arr2[]){
        for(int i=arr1.length-1;i>=0;i--){
            if(arr1[i]!=arr2[i])return i;
        }
        return -1;
    }
}
```



Take **n** as an integer input. Declare the first array of size **n** that stores values of char data-type. Then take **n** character inputs and store them in the array one by one. Traverse the array from the last and **print the index** at which the consonant occurs for the first time.

Input Format

First line consists **N** as size.

Second line consists an array of **N** char Values

Constraints

```
NA
```

Output Format

First **occurance** of Constant from end.

Sample Input 0

```
5
a
b
c
d
q
```

Sample Output 0

```
4
```

Explanation 0

q is first consonant from last

Submitted Code

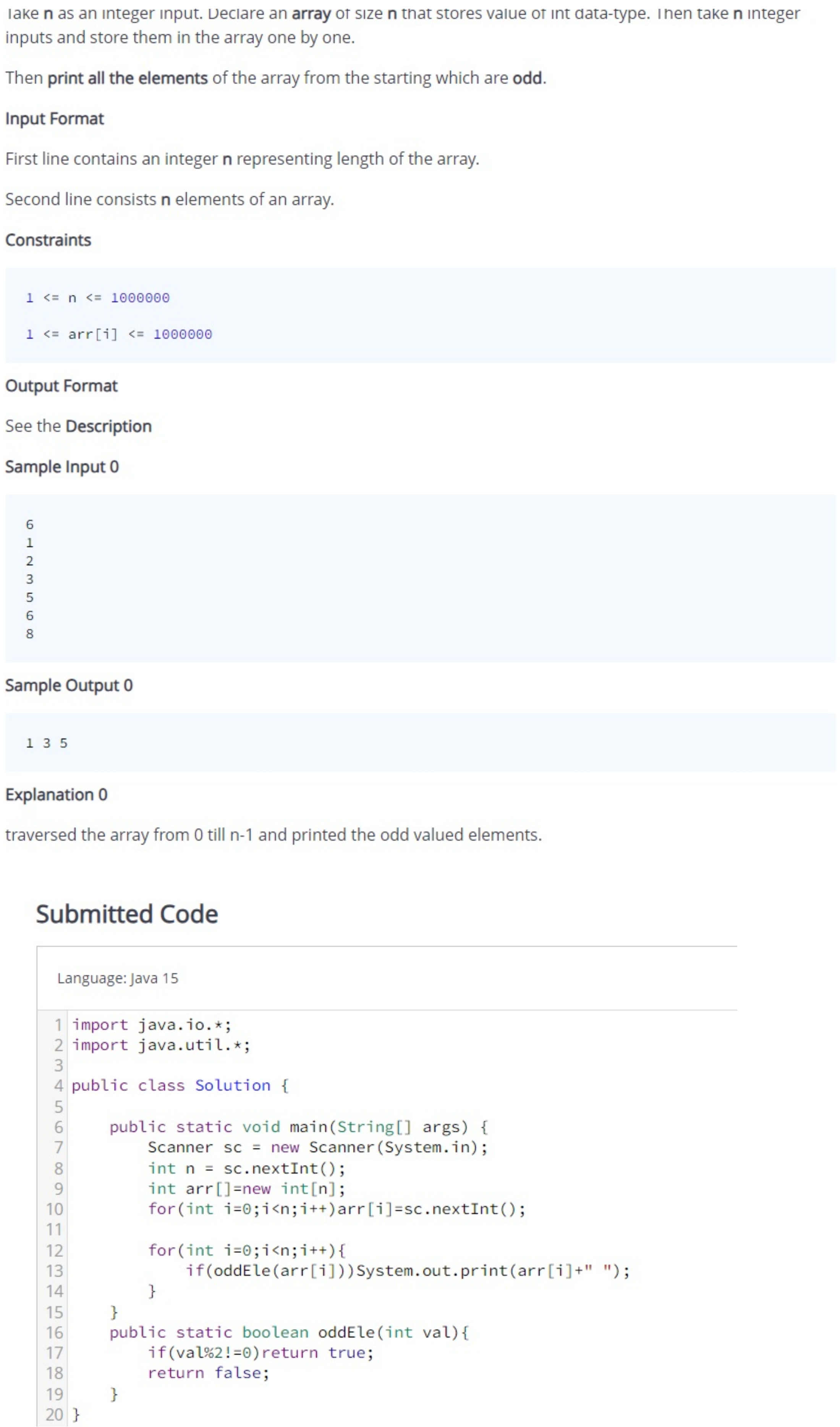
Language: Java 15

```
import java.io.*;
import java.util.*;

public class Solution {

   public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        char [] arr = new char [n];
        for(int i =0;i<n;i++)arr[i]=sc.next().charAt(0);

        System.out.print(lastConsonant(arr));
    }
    public static int lastConsonant(char [] arr){
        for(int i=arr.length-1;i>=0;i--){
            if(arr[i]!='a' || arr[i] !='e'|| arr[i]!='i' || arr[i] !='0'|| arr[i]!='u')return i;
        }
        return -1;
    }
}
```



Take **n** as an integer input. Declare an **array** of size **n** that stores value of int data-type. Then take **n** integer inputs and store them in the array one by one.

Then **print all the elements** of the array from the starting which are **odd**.

Input Format

First line contains an integer **n** representing length of the array.

Second line consists **n** elements of an array.

Constraints

```
1 <= n <= 1000000

1 <= arr[i] <= 1000000
```

Output Format

See the **Description**

Sample Input 0

```
6
1
2
3
5
6
8
```

Sample Output 0

```
1 3 5
```

Explanation 0

traversed the array from 0 till n-1 and printed the odd valued elements.

Submitted Code

Language: Java 15

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int arr[]=new int[n];
        for(int i=0;i<n;i++)arr[i]=sc.nextInt();

        for(int i=0;i<n;i++){
            if(oddEle(arr[i]))System.out.print(arr[i]+" ");
        }
    }
    public static boolean oddEle(int val){
        if(val%2!=0)return true;
        return false;
    }
}
```

a

[1, 2, 3, 4, 5, 6, 8]

1, 3, 5

function

void {

System.out.print()

int, float, double . . . . . . —— ——