Waiter

You are a waiter at a party. There is a pile of numbered plates. Create an empty $answers$ array. At each iteration, i, remove each plate from the top of the stack in order. Determine if the number on the plate is evenly divisible by the i^th prime number. If it is, stack it in pile B_i. Otherwise, stack it in stack A_i. Store the values in B_i from top to bottom in $answers$. In the next iteration, do the same with the values in stack A_i. Once the required number of iterations is complete, store the remaining values in A_i in $answers$, again from top to bottom. Return the $answers$ array.

Example

$A = [2, 3, 4, 5, 6, 7]$
$q = 3$

An abbreviated list of primes is $[2, 3, 5, 7, 11, 13]$. Stack the plates in reverse order.

$A_0 = [2, 3, 4, 5, 6, 7]$
$answers = []$

Begin iterations. On the first iteration, check if items are divisible by 2.
$A_1 = [7, 5, 3]$
$B_1 = [6, 4, 2]$

Move B_1 elements to $answers$.

$answers = [2, 4, 6]$

On the second iteration, test if A_1 elements are divisible by 3.
$A_2 = [7, 5]$
$B_2 = [3]$

Move B_2 elmements to $answers$.

$answers = [2, 4, 6, 3]$

And on the third iteration, test if A_2 elements are divisible by 5.
$A_3 = [7]$
$B_3 = [5]$

Move B_2 elmements to $answers$.

$answers = [2, 4, 6, 3, 5]$

All iterations are complete, so move the remaining elements in A_3, from top to bottom, to $answers$.

$answers = [2, 4, 6, 3, 5, 7]$. Return this list.

Function Description

Complete the *waiter* function in the editor below.

waiter has the following parameters:

- *int number[n]:* the numbers on the plates
- *int q:* the number of iterations

Returns

- *int[n]:* the numbers on the plates after processing

Input Format

The first line contains two space separated integers, n and q.
The next line contains n space separated integers representing the initial pile of plates, i.e., A.

Constraints

$1 \leq n \leq 5 \times 10^4$
$2 \leq number[i] \leq 10^4$
$1 \leq q \leq 1200$

Sample Input 0

```
5 1
3 4 7 6 5
```

Sample Output 0

```
4
6
3
7
5
```

Explanation 0

Initially:

A = [3, 4, 7, 6, 5]<-TOP

After 1 iteration (divide by 2, the 1st prime number):

A_1 = [5, 7, 3]<-TOP

B_1 = [6, 4]<-TOP

Move B_1 elements to $answers$.

$answers = [4, 6]$

All iterations are complete, so move A_1 elements to $answers$.

$answers = [4, 6, 3, 7, 5]$.

Sample Input 1

```
5 2
3 3 4 4 9
```

Sample Output 1

```
4
4
9
3
3
```

Explanation 1

Initially:

A = [3, 3, 4, 4, 9]<-TOP

After 1^{st} iteration (divide by 2):

A_1 = [9, 3, 3]<-TOP

B_1 = [4, 4]<-TOP

Move B_1 to $answers$.

$answers = [4, 4]$

After 2^{nd} iteration (divide by 3):

A_2 = []<- TOP

B_2 = [3, 3, 9]<-TOP

Move B_2 elements to $answers$.

$answers = [4, 4, 9, 3, 3]$

There are no values remaining in A_2.

Contest ends in 2 months

Submissions: 91
Max Score: 10
Difficulty: Medium

Rate This Challenge:
☆☆☆☆☆

More

Java 8

```java
import java.io.*;
import java.math.*;
import java.security.*;
import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.function.*;
import java.util.regex.*;
import java.util.stream.*;
import static java.util.stream.Collectors.joining;
import static java.util.stream.Collectors.toList;

class Result {

    /*
     * Complete the 'waiter' function below.
     *
     * The function is expected to return an INTEGER_ARRAY.
     * The function accepts following parameters:
     *  1. INTEGER_ARRAY number
     *  2. INTEGER q
     */
    // Write your code here
         public static List<Integer> waiter(List<Integer> number, int q) {
        int prime = 2;
        List<Integer> answer = new ArrayList<>();
        List<Integer> A = new ArrayList<>();
        for (int i = 0; i < q; i++) {
            for (int j = 0; j < number.size(); j++) {
                if (number.get(j) % prime == 0) {
                    answer.add(number.get(j));
                    number.remove(j);
                    j--;
                } else {
                    A.add(0, number.get(j));
                }
            }
            number.clear();
            number.addAll(A);
            A.clear();
            prime++;
            for (int j = 2; j < prime; j++) {
                if (prime % j == 0) {
                    prime++;
                    j = 2;
                }
            }
        }
        Collections.reverse(number);
        answer.addAll(number);
        return answer;
    }

    }



public class Solution {
    public static void main(String[] args) throws IOException {
        BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));
        BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT_PATH")));

        String[] firstMultipleInput = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

        int n = Integer.parseInt(firstMultipleInput[0]);

        int q = Integer.parseInt(firstMultipleInput[1]);

        List<Integer> number = Stream.of(bufferedReader.readLine().replaceAll("\\s+$", "").split(" "))
            .map(Integer::parseInt)
            .collect(toList());

        List<Integer> result = Result.waiter(number, q);

        bufferedWriter.write(
            result.stream()
                .map(Object::toString)
                .collect(joining("\n"))
            + "\n"
        );

        bufferedReader.close();
        bufferedWriter.close();
    }
}

```

Line: 56 Col: 1

Upload Code as File □ Test against custom input Run Code Submit Code

Testcase 0 ✓ Testcase 1 ✓

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
5 1
3 4 7 6 5
```

Your Output (stdout)

```
4
6
3
7
5
```

Expected Output

```
4
6
3
7
5
```

Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy |