**Problem statement**

You have beengiven a task to print the required pattern and he asked for your help with the same.

You must print a matrix corresponding to the given number pattern.

**Example:**

Input: ‘N’ = 4

Output:

4444444

4333334

4322234

4321234

4322234

4333334

4444444

**Constraints :**

1 <= N <= 10^2

Time Limit: 1 sec

**Sample Input 1:**

3

**Sample Output 1:**

33333

32223

32123

32223

33333

**Sample Input 2 :**

5

**Sample Output 2 :**

555555555

544444445

543333345

543222325

543212345

543222325

543333345

544444445

555555555

**Sample Input 3 :**

4

**Sample Output 3 :**

4444444

4333334

4322234

4321234

4322234

4333334

4444444

Code:

#include <stdio.h>

void generate\_pattern(int N) {

int size = 2 \* N - 1; // The size of the pattern

// Loop through each row

for (int i = 0; i < size; i++) {

// Loop through each column in the row

for (int j = 0; j < size; j++) {

// Calculate the minimum distance to any edge (top, left, bottom, right)

int dist = (i < j) ? i : j;

dist = (dist < size - i - 1) ? dist : size - i - 1;

dist = (dist < size - j - 1) ? dist : size - j - 1;

// Print the value based on the distance to the closest edge

printf("%d", N - dist);

}

// Print a new line after each row

printf("\n");

}

}

int main() {

int N;

// Take input for N

scanf("%d", &N);

// Generate and print the pattern for the given N

generate\_pattern(N);

return 0;

}

Test case-1:

Input: 10

Output:

10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10

10 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 10

10 9 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9 10

10 9 8 7 7 7 7 7 7 7 7 7 7 7 7 7 8 9 10

10 9 8 7 6 6 6 6 6 6 6 6 6 6 6 7 8 9 10

10 9 8 7 6 5 5 5 5 5 5 5 5 5 6 7 8 9 10

10 9 8 7 6 5 4 4 4 4 4 4 4 5 6 7 8 9 10

10 9 8 7 6 5 4 3 3 3 3 3 4 5 6 7 8 9 10

10 9 8 7 6 5 4 3 2 2 2 3 4 5 6 7 8 9 10

10 9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9 10

10 9 8 7 6 5 4 3 2 2 2 3 4 5 6 7 8 9 10

10 9 8 7 6 5 4 3 3 3 3 3 4 5 6 7 8 9 10

10 9 8 7 6 5 4 4 4 4 4 4 4 5 6 7 8 9 10

10 9 8 7 6 5 5 5 5 5 5 5 5 5 6 7 8 9 10

10 9 8 7 6 6 6 6 6 6 6 6 6 6 6 7 8 9 10

10 9 8 7 7 7 7 7 7 7 7 7 7 7 7 7 8 9 10

10 9 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9 10

10 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 10

10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10

Test case-2:  
input : 16

Output:

16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16

16 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 16

16 15 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 15 16

16 15 14 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 14 15 16

16 15 14 13 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 13 14 15 16

16 15 14 13 12 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 12 13 14 15 16

16 15 14 13 12 11 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 7 7 7 7 7 7 7 7 7 7 7 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 6 6 6 6 6 6 6 6 6 6 6 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 6 5 5 5 5 5 5 5 5 5 6 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 6 5 4 4 4 4 4 4 4 5 6 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 6 5 4 3 3 3 3 3 4 5 6 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 2 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 2 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 6 5 4 3 3 3 3 3 4 5 6 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 6 5 4 4 4 4 4 4 4 5 6 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 6 5 5 5 5 5 5 5 5 5 6 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 6 6 6 6 6 6 6 6 6 6 6 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 7 7 7 7 7 7 7 7 7 7 7 7 7 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 10 11 12 13 14 15 16

16 15 14 13 12 11 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 11 12 13 14 15 16

16 15 14 13 12 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 12 13 14 15 16

16 15 14 13 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 13 14 15 16

16 15 14 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 14 15 16

16 15 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 15 16

16 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 16

16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16

Test case -3:

Input: 19

Output:

19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19

19 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 19

19 18 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 18 19

19 18 17 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 17 18 19

19 18 17 16 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 16 17 18 19

19 18 17 16 15 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 15 16 17 18 19

19 18 17 16 15 14 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 7 7 7 7 7 7 7 7 7 7 7 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 6 6 6 6 6 6 6 6 6 6 6 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 5 5 5 5 5 5 5 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 4 4 4 4 4 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 3 3 3 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 2 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 2 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 3 3 3 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 4 4 4 4 4 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 5 5 5 5 5 5 5 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 6 6 6 6 6 6 6 6 6 6 6 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 7 7 7 7 7 7 7 7 7 7 7 7 7 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 13 14 15 16 17 18 19

19 18 17 16 15 14 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 14 15 16 17 18 19

19 18 17 16 15 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 15 16 17 18 19

19 18 17 16 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 16 17 18 19

19 18 17 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 17 18 19

19 18 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 18 19

19 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 19

19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19  
  
Test case-4:

Input : 1

Output:

1

Test Case-5:

Input: 11  
Output:

11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11

11 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 11

11 10 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 10 11

11 10 9 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9 10 11

11 10 9 8 7 7 7 7 7 7 7 7 7 7 7 7 7 8 9 10 11

11 10 9 8 7 6 6 6 6 6 6 6 6 6 6 6 7 8 9 10 11

11 10 9 8 7 6 5 5 5 5 5 5 5 5 5 6 7 8 9 10 11

11 10 9 8 7 6 5 4 4 4 4 4 4 4 5 6 7 8 9 10 11

11 10 9 8 7 6 5 4 3 3 3 3 3 4 5 6 7 8 9 10 11

11 10 9 8 7 6 5 4 3 2 2 2 3 4 5 6 7 8 9 10 11

11 10 9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9 10 11

11 10 9 8 7 6 5 4 3 2 2 2 3 4 5 6 7 8 9 10 11

11 10 9 8 7 6 5 4 3 3 3 3 3 4 5 6 7 8 9 10 11

11 10 9 8 7 6 5 4 4 4 4 4 4 4 5 6 7 8 9 10 11

11 10 9 8 7 6 5 5 5 5 5 5 5 5 5 6 7 8 9 10 11

11 10 9 8 7 6 6 6 6 6 6 6 6 6 6 6 7 8 9 10 11

11 10 9 8 7 7 7 7 7 7 7 7 7 7 7 7 7 8 9 10 11

11 10 9 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 9 10 11

11 10 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 10 11

11 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 11

11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11

Split the array to make co-prime products

Description:

You are given a 0-indexed integer array nums of length n.

A split at an index i where 0 <= i <= n - 2 is called valid if the product of the first i + 1 elements and the product of the remaining elements are co-prime.

For example, if nums = [2, 3, 3], then a split at the index i = 0 is valid because 2 and 9 are co-prime, while a split at the index i = 1 is not valid because 6 and 3 are not co-prime. A split at the index i = 2 is not valid because i == n - 1.

Return the smallest index i at which the array can be split validly or -1 if there is no such split.

Two values val1 and val2 are co-prime if gcd(val1, val2) == 1 where gcd(val1, val2) is the greatest common divisor of val1 and val2.

Input format:

1. The first line contains an integer n, that represents number of elements in the array.

2. The second line contains n space separated integers nums[i] representing the elements of the array.

Constraints:

n == nums.length

1 <= n <= 104

1 <= nums[i] <= 106

Code:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_N 1000

void factorize(int n, int \*factors, int \*count) {

\*count = 0;

for (int i = 2; n > 1 && i < MAX\_N; i += 1 + (i % 2)) {

if (n % i == 0) {

factors[(\*count)++] = i;

while (n % i == 0) {

n /= i;

}

}

}

if (n > 1) {

factors[(\*count)++] = n;

}

}

int findValidSplit(int \*nums, int numsSize) {

int right[MAX\_N] = {0}, left[MAX\_N] = {0}; // Maps to track prime factors

int factors[20], count;

for (int i = 0; i < numsSize; i++) {

factorize(nums[i], factors, &count);

for (int j = 0; j < count; j++) {

right[factors[j]]++;

}

}

int common = 0;

for (int i = 0; i < numsSize - 1; i++) {

factorize(nums[i], factors, &count);

for (int j = 0; j < count; j++) {

int f = factors[j];

left[f]++;

if (left[f] == 1) common++; // Increment if it's a new prime factor on the left

if (left[f] == right[f]) common--; // Decrement if counts match

}

if (common == 0) {

return i; // Valid split found

}

}

return -1; // No valid split found

}

int main() {

int numsSize;

printf("Enter the number of elements: ");

scanf("%d", &numsSize);

if (numsSize < 2) {

printf("Invalid input! The array must have at least 2 elements.\n");

return 0;

}

int nums[numsSize];

printf("Enter the elements: ");

for (int i = 0; i < numsSize; i++) {

scanf("%d", &nums[i]);

}

int result = findValidSplit(nums, numsSize);

printf("Smallest valid split index: %d\n", result);

return 0;

}

Testcase-1:

Input:

6

4 7 8 15 3 5

Output:

2

Testcase-2:

Input:

6

4 7 15 8 3 5

Output:

-1

Testcase-3:

Input:

4

5 6 7 8

Output:

0

Testcase-4:

Input:

5

3 5 7 11 13

Output:

0

Testcase-5:

Input:

6

4 8 12 16 20 24

Output:

-1

Testcase-6:

Input:

6

10 15 21 35 49 77

Output:

-1

Testcase-7:

Input:

6

12 18 30 42 56 72

Output:

-1

Testcase-8:

Input:

20

6 10 15 21 35 49 77 2 3 5 7 11 13 17 19 23 29 31 37 41

Output:

11

Testcase-9:

Input:

25

4 6 9 10 14 15 21 22 25 26 33 35 38 39 44 46 49 55 57 58 63 65 77 91 97

Output:

23

Testcase-10:

Input:

30

6 10 15 21 35 49 77 22 26 33 39 51 65 14 9 25 2 3 5 7 11 13 17 19 23 29 31 37 41 43

Output:

22