Lab 5 A

Problem 1: The Great Cake Dilemma

During the Bake Festival, two chefs, Chef Whisk and Chef Spoon, always bake cakes and share it to the people in town. They believe all the people who are given cakes should get equal number of slices. But they have a quirky problem: they always bake cakes with a different number of slices. They want to find the largest number of people they can share their cakes with such that the cakes can be evenly shared leaving any leftovers. Your task is to help the chefs figure out the largest number of people that can eat cakes everyday!

Input Format:

- The first line contains an integer N, representing the number of days.
- The next N lines each contain two integers: $s1$ (slices in Chef Whisk's cake) and $s2$ (slices in Chef Spoon's cake).

Constraints:

- $1 \le N \le 100$
- $1 \le s1, s2 \le 10^{18}$

Output Format:

Return a single integer: the largest number of people that was served during the festival

Input 1:

```
3                                                                          Copy
12  15
18  24
24  36
```

Output 2:

```
12                                                    Copy
```

Input 2:

```
2                                                     Copy
7 5
13 27
```

Output 2:

```
1                                                     Copy
```

Explanation:

In the first example, the maximum number of slices that can be evenly divided between Chef Whisk and Chef Spoon for each pair of cakes are:

- For the pair (12, 15), they can serve 3 people, with each person getting 4 slices from the first cake and 5 from the second cake.
- For the pair (18, 24), they can serve 6 people, with each person getting 3 slices from the first cake and 4 from the second cake.
- For the pair (24, 36), they can serve 12 people, with each person getting 2 slices from the first cake and 3 from the second cake.

The largest number of people they can serve during the festival is 12.

In the second example, they can only serve 1 from both pairs, so the output is 1.

Problem 2: Neel's Coin Challenge

One day, Neel met a wise merchant in a bustling marketplace. The merchant handed him a scroll with a number on it and said, "Neel, you must collect coins to match this exact sum. These coins come in various values, and you have an infinite supply of each."

Neel looked at the coins scattered around, each with different denominations. "How many ways can I combine these to reach the sum?" he asked.

"That's your task," the merchant replied with a smile. "Find all the combinations to unlock the treasure."

Neel sat down, carefully calculating every possible way to combine the coins to match the target sum. With each new combination, the lock on the treasure chest glowed brighter.

Could Neel find all the ways and unlock the treasure?

Input Format:

- The first line contains an integer **sum**, representing the target value.
- The second line contains an integer **n**, representing the size of the array.
- The third line contains an integer array **coins[]**, representing the coin denominations available.

Output Format:

Print the total number of distinct combinations that can form the target **sum**.

Constraints:

- $1 \leq$ **sum** ≤ 30
- $1 \leq$ **N** ≤ 10
- $1 \leq$ **coins[i]** ≤ 30

Example 1:

```
Input:
    4
    3
    1 2 3

    Output:
    4
```

Explanation: There are four combinations to make sum 4:

- {1, 1, 1, 1}
- {1, 1, 2}
- {2, 2}

- {1, 3}

Example 2:

```
Input:
    10
    4
    2 5 3 6

    Output:
    5
```

Explanation: There are five combinations to make sum 10:

- {2, 2, 2, 2, 2}
- {2, 2, 3, 3}
- {2, 2, 6}
- {2, 3, 5}
- {5, 5}

Example 3:

```
Input:
    10
    1
    10

    Output:
    1
```

Explanation: The only way to make sum 10 is by using one coin of value 10.

Example 4:

```
Input:
    sum = 5
    coins[] = {4}

    Output:
```

```
0
```

Explanation: There is no way to make sum 5 with the given coins.

Problem 3: Vector Multiplication - A Galactic Conundrum

In a distant galaxy far, far away, the interstellar travelers from planet **Mathlore** are facing a peculiar problem. They are navigating through space and have encountered two asteroids, each represented by vectors in 3D space. These space travelers, known for their love of mathematical challenges, must figure out if these asteroids will collide or pass by harmlessly. To solve this, they need your help in calculating the **cross product** of the asteroids' vectors and determine whether the paths are **orthogonal** (perpendicular) to each other.

The interstellar navigators have just enough energy to run your code once, so they need you to define a struct in C that represents the x, y, and z components of the vectors. Help them compute the cross product of these vectors, and figure out if the asteroids are heading for an intergalactic showdown (orthogonal paths) or just a friendly flyby.

Task 1: Compute the Cross Product

When given two vectors, say A and B, the cross product will give the travelers a new vector that is perpendicular to both. This is essential to avoid *space dust* in their navigational systems.

Display the cross product so the travelers can recalibrate their trajectory.

Task 2: Check for Orthogonality

Once the cross product is calculated, the travelers must determine if the vectors are orthogonal. If orthogonal, it means the asteroids are on a direct collision course! Otherwise, it's smooth sailing ahead.

Input Format:

- The input consists of two vectors, each with three integer components (x, y, z).

Output Format:

- Display the cross product of the two vectors. (format - 3 integers)
- Indicate if the vectors are orthogonal, or if the travelers are safe. (format - YES/NO)

Constraints:

- The vector components must be integers between -1000 and 1000.

Example Input 1:

```
3 -3 1
4 9 2
```

Example Output 1:

```
-15i -2j +39k
NO
```

Example Input 2:

```
1 0 0
0 1 0
```

Example Output 2:

```
k
NO
```

Example Input 2:

```
0 0 0
0 0 0
```

Example Output 2:

```
0
YES
```