Sharing Chocolate

Consider a chocolate bar consisting of 3 x 4 segments. You can break the chocolate bar along any of the divisions between the rows or columns, creating two bars that themselves can be split in the same manner.

A specific question: Can the 3 x 4 chocolate bar be split into 4 pieces with 1, 2, 3, and 6 perfectly rectangular segments? The more general question: Can an m by n chocolate bar (0 <= m, n <= 100) be split into p pieces (1 <= p <= 15) with a_1, a_2, a_3, .., a_{p-1}, and a_p perfectly rectangular segments?

The problem can be solved using recursive backtracking and memoization, and while challenging, it's accessible to all of you given what we've covered in class so far. It's possible to use recursive backtracking alone, but the memoization can be used to remember common sub-problems and bring an exponential running time solution down to a polynomial one. Even 100 x 100 chocolate bars can be recursively divided to discover the desired split in less than a few seconds.

Input

The data needs to be read in from a file named **`sharing-chocolate.in`**. The input file consists of multiple test cases, each describing a chocolate bar. Each description starts with a line containing two integers, which are the dimensions of a chocolate bar. The next line contains a list of (unsorted) integers (with or without duplicates), which are the sizes of the segments. The end of input is the end of file (i.e. an attempt to read two integers would bring an **`ifstream`** into a fail state).

Output

For each test case, print **`yes`** or **`no`**. Print **`yes`** if the desired split is possible, and print **`no`** otherwise. Your output should be published to a file called **`sharing-chocolate-<your-sunet-id>.out`** and place it the same directory housing **`sharing-chocolate.in`**.

Sample Input:

```
3 4
1 2 3 6
5 5
6 4 9 6
2 3
1 5
4 4
4 2 1 1 4 4
```

Sample Output:

```
yes
yes
no
yes
```