Problem Statement

Each digit can be represented using a certain number of matches. Your goal is to create the largest possible number using the matches that you have. For example, if you need 6 matches for zero, 7 matches for one, and 8 matches for two, and you have 21 matches, the largest number you can create is 210 (8 + 7 + 6 = 21 matches).

You are given a int[] **matches** and an int **n**. The i^{th} element (zero-indexed) of **matches** is the number of matches needed to represent the digit i. **n** is the number of matches you have. Return the largest possible number you can create without extra leading zeros.

Definition

Class:	MatchNumbersEasy
Method:	maxNumber
Parameters:	int[], int
Returns:	String
Method signature:	String maxNumber(int[] matches, int n)
(be sure your method is public)	

Notes

- It is not necessary to use all given matches. Some matches may be left unused.

Constraints

- **matches** will contain between 1 and 10 elements, inclusive.
- Each element of **matches** will be between 1 and 50, inclusive.
- **n** will be between 1 and 50, inclusive.
- **n** matches will be enough to construct at least 1 digit.

Examples

0)

```
{6,7,8}
21
Returns: "210"
```

Example from the problem statement.

1)

```
{5,23,24}
30
Returns: "20"
```

24 matches for two and 5 matches for zero. 1 match is left unused.

2)

```
{1,5,3,2}
1
Returns: "0"
```

This is the only number that can be created.

3)

```
{1,1,1,1,1,1,1,1,1,1}
50
Returns: "99999999999999999999999999999999999999999999999999"
```