JS Recoupling

Write a function `divide` that takes two arguments: *i*) a word (`string`) and *ii*) the number of (non-overlapping) groups $n \in \mathbb{N}_0$ (`number`) into which the word must be divided. If the word passed to the function `divide` cannot be divided into n groups that have the same length, an exception must be raised with the message `invalid division`. Otherwise, the function must return a list (`array`) containing the n groups (`string`) into which the given word can be divided. All groups need to have the same length (same number of letters).

Write another function `recouple` that takes two arguments: *i*) a sequence (`array`) of $m \in \mathbb{N}_0$ words (`string`) and *ii*) the number of (non-overlapping) groups $n \in \mathbb{N}_0$ (`number`) into which the words must be divided. If at least one of the words passed to the function `recouple` cannot be divided into n groups that have the same length, an exception must be raised with the message `invalid division`. Otherwise, the function must return a sequence containing the n new words (`string`) obtained when each of the m given words is divided into n groups that have the same length, and if each of the m corresponding groups is merged into a new word. The type of the returned sequence (`array`) must correspond to the type of the sequence passed as a first argument to the function.

Example

```
> divide("accost", 3)
["ac", "co", "st"]
> divide("COMMUNED", 4)
["CO", "MM", "UN", "ED"]
> divide("programming", 5)
Error: invalid division

> recouple(["ACcoST", "COmmIT", "LAunCH", "DEedED"], 3)
["ACCOLADE", "communed", "STITCHED"]
> recouple(["ACCOLADE", "communed", "STITCHED"], 4)
["ACcoST", "COmmIT", "LAunCH", "DEedED"]
> recouple(["programming", "computer", "games"], 5)
Error: invalid division
```