Question 1: Up or Down?

Create a function `order?` that consumes a list of `Num`, which has length at least two, and that does not contain any duplicates. It returns "ascending" if the numbers are ordered in ascending order, "descending" if it is in descending order, and "mixed" if neither. For example

(order? (list 1 3 4 7)) => "ascending"
(order? (list 1 2 3 6 5)) => "mixed"

Question 2: We Don’t Need ’Em

Create a function `rid` that consumes a list of at least length 2 and gets rid of the first and last items of a list and keeps the items in the middle. For example

(rid (list 1 2 3 4)) => (list 2 3)

Question 3: Even and Odd

Create a function `more-even?` that consumes a non-empty list of natural numbers and returns `true` if there are more even numbers than odd numbers in the list.

Question 4: Multiply some

Create a function `(multi n L k)` that consumes a `Nat`, a list of `Num`, and a `Num`, and returns a list containing all the values of `L`, where the first `n` values have been multiplied by `k`. For example:

(multi 3 (list 2 3 5 7 11) 10) => (list 20 30 50 7 11)

Question 5: Double Stuff

Create a function `(double-nums L)` that consumes a list of `Any` and returns a copy of the list where all values in `L` that are numbers have been multiplied by 2. The last value in the list is not changed, even if it is a number. For example

(double-nums (list 1 "a" 2 "b" 3)) => (list 2 "a" 4 "b" 3)