Test This (#1)

Task. Write a reasonable set of unit tests for the specification below. You *do not* have to implement the specified function, just the tests.

Specification: Write a function `findInterval` that takes a test point, `query_point`, and a sorted list of interval endpoints, `intervals`, and returns the index of the largest endpoint that is ≤ `query_point`. If `query_point` lies below all the endpoints, return one less than the minimum index (-1 for 0-indexed languages, 0 for 1-indexed languages like R).

To clarify, both the query point and each of the interval endpoints are real numbers. Specifically, the query point is a number q and the interval endpoints are numbers

$$a_0 < a_1 < a_2 < \cdots < a_{n-1}.$$

The task is to find which interval specified by a consecutive pair of the a_i's contains the point q. If $a_k \leq q < a_{k+1}$, the function should return k. If $q > a_{n-1}$, the function should return $n-1$, and if $q < a_0$, the function should return -1. (In R, where arrays are indexed starting with 1, we would call these a_1 through a_n and return n and 0 in the last two cases, respectively.)

For example,

```
findInterval(1.2, [1.0, 2.0, 3.0, 4.0])  =>  0 (0-indexed) or 1 (1-indexed)
findInterval(0.2, [1.0, 2.0, 3.0, 4.0])  => -1 (0-indexed) or 0 (1-indexed)
findInterval(3.5, [1.0, 2.0, 3.0, 4.0])  =>  2 (0-indexed) or 3 (1-indexed)
findInterval(5.0, [1.0, 2.0, 3.0, 4.0])  =>  3 (0-indexed) or 4 (1-indexed)
```

(Don't use these examples in your tests.)

Requirements.

☐ Write unit tests for the function specified above.

☐ You need not implement the described function, however.

☐ Use whatever language and testing framework you prefer.