Lab 07: Finding the Defect

In this lab, you will be presented with a list of collection of `n` coins. All of these coins have identical weights except for one. The goal of the lab is to identify the defective coin by using a scale that can compare weights of two groups of coins.

Representing the Collection of Coins and Scale

A `Scale` class that represents both the collection of coins and the scale will be provided. The class will have the following ADT:

- `__init__(self, L)` - initialize a `Scale` object with `L`, which is the list of weights of coins
- `weigh(self, range1, range2)` - `range1` and `range2` are both tuples of two elements which represent a range of coins. Returns `1` if the sum of the weights of elements in `range2` are more than weights of elements in `range1`. Returns `0` if the weights are equal, and `-1` otherwise.
- `__len__(self)` - returns the length of the collection of coins

```
L= [1,1,1,1,1,2,1,1,1]
s = Scale(L)
print(s.weigh((0,3), (3,9)))
```

```
1
```

As the sum of weights from `[3,9)` is larger than the sum of the weigths from `[0,3)`, the function will output 1. You will use this class to implement a function to determine the index of the defective weight. It is important to note that your function should not depend on the implementation of `Scale`, it should only depend on the defined ADT above.

Goal

Implement a function, `defects`, which takes in a `Scale` object and returns an integer which is the index of the defective coin. It should use the `weigh` method of the `Scale` class to divide the problem into smaller pieces, atleast halve the problem. The final product should have a runtime of `O(log n)`.

For example, the following is an example output:

```
L= [1,1,1,1,1,2,1,1,1]
s = Scale(L)
print(defects(s))
```

```
5
```

A case you may run into when trying to halve the list is how to tackle odd length lists. An easy solution is to check the middle element manually. Then, the remaining even number of elements can be handled by dividing as before.

Summary

For this lab, you should implement the `defects` method that takes in a `Scale` object and returns the index of the defective coin. This method should use the `weigh` method of the `Scale` object to divide the problem and find a solution in logarithmic time. An example `Scale` class will be provided for testing. However, your solution should not depend on the implementation of `Scale`. All it should rely on is the ADT provided for `Scale` in this lab.