Solution Review: Sum of Lists

In the following lesson, we will go over the solution for the challenge: Sum of Lists.

We'll cover the following

Task

In this challenge, you had to create a recursive function, `sum`, which sums together all the integers in a list.

Solution

A skeleton of the function was already provided for you. Let’s look it over.

```
int sum(List<int> numberList, int index) {

}
```

`sum` takes two parameters. The first is a list of type `List<int>` and the second is the `index` of the last item. `sum` returns a value of type `int`.

General Logic

We will start the summation from the last item in the list and move our way backwards. This is why we need to keep track of the index of the last item in the list.

Base Case

Since we are required to create a recursive function, the first thing we need to do is figure out the base case. The smallest possible list is an empty list. The smallest index is **0**, this would mean that when `index` < **0**, our base case should be called which will simply return **0**.

```
if(index < 0 ){
    return 0;
  }
```

Recursive Case

The second case is if the List is not empty. In this case, we separate the last element of the List from the rest of the elements.

We need to sum the last element with the recursive call to `sum`. This is passed `index-1` and the remaining list excluding the last item as the new list will be one item smaller.

```
else {
    return numberList[index] + sum(numberList, index-1);
  }
```

If our list has **5** elements, we need to pass **4** to sum. The recursive case will be called as `index` < **0**. `sum` will now look at the list starting from the second to last index, i.e., **3**. This will continue until we have reached the first item of the list.

You can find the complete solution below:

> You were required to write the code written from **line 2** to **line 6**.

```
int sum(List<int> numberList, int index) {
  if(index < 0 ){
    return 0;
  } else {
    return numberList[index] + sum(numberList, index-1);
  }
}

main() {
  // Driver Code
  var result = sum([1,2,3,4,5], 4);
  print(result);
}
```

Another Solution

You can also use the ternary operator to solve this challenge.

```
int sum(List<int> numberList, int index) {
  return numberList[index] + (index > 0 ? sum(numberList, index - 1) : 0);
}
int main() {
  // Driver Code
  var result = sum([1,2,3,4,5], 4);
  print(result);
}
```

Let’s move on to higher-order functions in the next lesson.