Summation with Anonymous Functions

In this lesson, we will see a practical example of anonymous functions and see how they allow you to write concise code.

We'll cover the following

- sumOfInts
- sumOfCubes
- sumOfFactorials

Let’s revisit our summation program using the anonymous function. Before we get started, here is the code we have written so far.

```
// Higher-order sum function
def sum(f: Int => Int, a: Int, b: Int): Int ={
  if(a>b) 0
  else f(a) + sum(f, a+1, b)
}

// Helper function
def id(x: Int): Int = x

def cube(x: Int): Int = x*x*x

def factorial(x: Int): Int = if (x==0) 1 else x * factorial(x-1)

// Summation Functions
def sumOfInts(a: Int, b: Int) = sum(id, a, b)
def sumofCubes(a: Int, b: Int) = sum(cube,a,b)
def sumOfFactorials(a: Int, b:Int) = sum(factorial,a,b)
```

To write our summation functions using anonymous functions, we will start by creating our `sum` function the same as before.

```
def sum(f: Int => Int, a: Int, b: Int): Int ={
  if(a>b) 0
  else f(a) + sum(f, a+1, b)
}
```

The code for the `sum` function hasn’t changed.

For this version of the program, we aren’t required to make helper functions and we can simply start defining our summation functions `sumOfInts`, `sumOfCubes`, and `sumOfFactorials`. Where ever we inserted the helper functions in the previous program, we will now insert an anonymous function, i.e., the first argument of `sum`.

`sumOfInts`

This code requires the following environment variables to execute:

LANG	C.UTF-8

```
def sum(f: Int => Int, a: Int, b: Int): Int ={
  if(a>b) 0
  else f(a) + sum(f, a+1, b)
}

def sumOfInts(a: Int, b: Int) = sum(x => x, a, b )

// Driver Code
println(sumOfInts(1,5))
```

`sumOfCubes`

This code requires the following environment variables to execute:

LANG	C.UTF-8

```
def sum(f: Int => Int, a: Int, b: Int): Int ={
  if(a>b) 0
  else f(a) + sum(f, a+1, b)
}

def sumofCubes(a: Int, b:Int) = sum(x => x*x*x, a, b)

// Driver Code
println(sumofCubes(1,5))
```

`sumOfFactorials`

This code requires the following environment variables to execute:

LANG	C.UTF-8

```
def sum(f: Int => Int, a: Int, b: Int): Int ={
  if(a>b) 0
}

def sumOfFactorials(a: Int, b:Int) = {
  sum({def f(x: Int):Int = if(x==0) 1 else x * f(x-1); f},a,b)
}


// Driver Code
println(sumOfFactorials(1,5))
```

As the helper function that was created for `sumOfFactorials` was a recursive function, we needed to create an anonymous function by giving it a temporary name `f` to allow the function to call itself in the function body.

This appears to make the code difficult to read. Hence, it is better to not call recursive functions anonymously.

This code requires the following environment variables to execute:

LANG	C.UTF-8

```
def sum(f: Int => Int, a: Int, b: Int): Int ={
  if(a>b) 0
  else f(a) + sum(f, a+1, b)
}

def factorial(x: Int): Int ={
  if(x==0) 1
  else x * factorial(x-1)
}

def sumOfFactorials(a: Int, b: Int) = sum(factorial,a,b)

println(sumOfFactorials(1,5))
```

In the next lesson, you will optimize a program using anonymous functions.