Assignment - Kotlin

11/5/2024

45 Points Possible

Attempt 1

10/28/2024

NEXT UP: Review Feedback

Attempt 1 Score: N/A

Add Comment

Unlimited Attempts Allowed

Details

Overview

This assignment is an introduction to programming in Kotlin. There are on-line environments for Kotlin at `http://play.kotlinlang..org` (http://play.kotlinlang.org) and at `http://repl.com` (http://repl.it) . Kotlin is available on most IDEs (for instance, you can already use Kotlin in IntelliJ, just create a new Kotlin project). Alternatively or additionally you can download and install Kotlin from `http://kotlinlang.org` (http://kotlinlang.org) . Documentation and tutorials on Kotlin are available from `http://kotlinlang.org` (http://kotlinlang.org) . More Kotlin resources can be found at the class resource page (https://usu.instructure.com/courses/761606/pages/resources) . Kotlin can be installed on (all?) most operating systems (e.g., Mac, Linux, Windows).

Turnin

All code should be developed in the file called `Lists.kt`. The problem solution will consist of a sequence of (function) definitions. You may turn in your assignment using the Canvas. Just upload the file `Lists.kt`. You may turnin your assignment as many times as you like. If you write code to test the functions, please place the test code in a separate file; `Lists.kt` should have only the function definitions (and comments!).

Getting Started

Download the starter code in `Lists.kt` (https://usu.instructure.com/courses/761606/files/92926536?wrap=1) (https://usu.instructure.com/courses/761606/files/92926536/download?download_frd=1) and the testing code in `Main.kt` (https://usu.instructure.com/courses/761606/files/92928723?wrap=1) (https://usu.instructure.com/courses/761606/files/92928723/download?download_frd=1) .

Grading

The assignment will be marked for good programming style (indentation and appropriate comments), correct type signatures, clean compilation, and correct execution. Please be sure to fully comment all function headers. It is important to the inputs to the function, its output(s), any side effects, and what the function is intended to do. Each problem is worth 5 points and style is worth 5 points.

What to Do

The assignment is to define the following functions.

Function 1 - generateElement

Write a function to construct a function using a passed list that generates each element in the passed list. *Hint, take advantage of closure!* The function should have the following signature (note: the return type is a function that takes no arguments and returns an integer or the null value if all of the elements in the list have been previously generated).

`fun generateElement(List<Int>) : () -> Int?`

Here are some examples of how it would be used.

```
val f = generateElement(listOf(1, 2, 3))
println(f())  // prints 1
println(f())  // prints 2
println(f())  // prints 3
println(f())  // prints null
val g = generateElement(emptyList())
println(g()) // prints null
```

Function 2 - generateRest

Write a function to construct a function using a passed list that generates a list with the "current" element removed. Assume that if the function has been called *n* times, the current element is the *n*th element in the list. The function should returns a list of integers or the null value if all of the lists in the list have been previously generated.

Here are some examples of how it would be used.

```
val f = generateRest(listOf(1, 2, 3))
println(f())  // prints [2, 3]
println(f())  // prints [1, 3]
println(f())  // prints [1, 2]
println(f())  // prints null
val g = generateRest(emptyList())
println(g()) // prints null
```

Function 3 - primeNumbers

Using a lambda expression, *i.e.*, `list.filter{}`, write a function to build a list of the prime counting numbers up to *n*. A prime number is evenly divisible only by 1 and itself, though 1 is not a prime number. You may assume `n` is an `Int` or a `null`, that is, the function must ensure null safety.

```
  primeNumbers(3)    // returns the list [2,3]
  primeNumbers(5)    // returns the list [2,3,5]
  primeNumbers(0)    // returns the list []
  primeNumbers(null) // returns null
```

Function 4 - runLengthEncoding

Write a function to build the run-length encoding of a list. The run-length encoding is a list of tuples where each tuple is a pair consisting of the number of elements found consecutively in the list. The examples below demonstrate how it should work.

```
runLengthEncoding(listOf(7)) // returns [[7,1]]
runLengthEncoding(listOf(7, 7, 4, 7, 7, 7)) // returns [[7,2], [4, 1], [7, 3]]
runLengthEncoding(emptyList()) // returns []
```

ed. Either list may be empty or `null`, that is, the function must ensure null safety. If a parameter is *null* then it returns *null*.

Function 5 - listApply

Write a function to apply a binary function `f` to the elements in a list of lists. Below is a template of an example call.

```
   listApply(f,[[x1, x2, x3] [y1, y2], [w1]]) = [f(x1, f(x2, x3)), f(y1, y2), w1]
```

The second parameter, the list of lists, may be `null`, in which case the function should return `null`. You may assume that every sublist has at least one element. The result of applying a function to a one element sublist is the value of that element.

```
  fun add(x : Int, y : Int) : Int = x + y
  listApply(::add, listOf(listOf(1),
                          listOf(1,2),
                          listOf(1,2,3)))   // returns [1, 3, 6]
  listApply(::add, null)                    // null
```

Function 6 - connectTheLists

A pair of lists can be connected if the *tail* of one list matches the *head* of the other list, and is connected in that order.

For example, list [a, b, c] connects with list [c, d] in the order [a, b, c], [c, d]. List [a, b, c] connects with the list [d, a], in the order [d, a], [a, b, c]. List [a, b, c] does not connect to the list [b, f].

Write code to connect each list in a list of lists into a *chain*, where the chain is a list of lists such that each list is connected in the order in which it connects. If no such chain can be constructed the function should return *null*.

```
// Some potential lists of lists to try to connect
val words0 : List<List<Int>> = emptyList().
val words1 = listOf(listOf(1, 2), listOf(3, 1))
val words2 = listOf(listOf(1, 2), listOf(5, 3))
val words3 = listOf(listOf(1, 5, 0),
                    listOf(2, 7),
                    listOf(0, 2, 3),
                    listOf(3, 2)
                   )

connectTheLists(words0) // returns []
connectTheLists(words1) // returns [[3, 1], [1, 2]]
connectTheLists(words2) // returns null
connectTheLists(words3) // returns [[1, 5, 0], [0, 2, 3], [3, 2], [2, 7]]
```

Copyright © 2024 by Curtis Dyreson. All rights reserved.

Preview Unavailable

Lists.kt

Download

(https://usu.instructure.com/files/93004913/download?download_frd=1&verifier=BRwirqfRc1fTVXjcGms2oewpgqDslZIZhN9e0IGO)

New Attempt