Taller Estructuras de Datos en Kotlin

Juan David Lopez Montoya

El objetivo de este taller es que los aprendices sean capaces de comprender y utilizar las principales estructuras de datos en Kotlin, incluyendo arreglos, listas, conjuntos, mapas y pares. El aprendiz deberá realizar un informe donde se evidencian los siguientes puntos:

1. Introducción a las estructuras de datos en Kotlin
   1. ¿Qué son las estructuras de datos y para qué se utilizan?

R/: son aquellas que permiten organizar la información de manera ordenada y eficientemente

* 1. Ventajas de utilizar estructuras de datos en Kotlin

R/:Interoperabilidad con código Java. ...

1. Curva de aprendizaje fácil. ...
2. Tiempo de programación reducido. ...
3. Desarrollo multiplataforma. ...
4. Flexibilidad
   1. Diferencias entre las estructuras de datos en Kotlin y Java

R/:Ambos tienen casi la misma velocidad de codificación, pero Java requiere escribir más código. Sin embargo, se dedica menos tiempo a pensar la solución con Java. **Kotlin tiene construcciones más concisas, por lo que requiere escribir menos código**. Sin embargo, se tarda más tiempo en encontrar la solución a una tarea

1. Arreglos en Kotlin
   1. ¿Qué es un arreglo?

R/:Un arreglo es un conjunto de múltiples datos o objetos guardados en una variable

* 1. Creación de arreglos en Kotlin

R/:Kotlin usa la clase genérica Array<T> para representar arreglos

* 1. Accediendo a los elementos de un arreglo

R/:Puedes hacer lo mismo si **usas el método get()** . La sintaxis de subíndice y el método get() toman un elemento Int como parámetro y muestran el elemento en ese índice

* 1. Modificando los elementos de un arreglo

R/:arreglo[0] = valor o val elemento = arreglo[0]

* 1. Recorriendo un arreglo

R/:

val nombres = arrayOf("Luis", "María José", "Fernando")

for (nombre in nombres){

println(nombre)

}.

* 1. Funciones útiles para trabajar con arreglos en Kotlin

R/:

val readOnly: List<String> = listOf("Lunes", "Martes", "Miércoles", "Jueves", "Viernes", "Sábado", "Domingo")

readOnly.size //Muestra el tamaño de la lista

readOnly.get(3) //Devuelve el valor de la posición 3

readOnly.first() //Devuelve el primer valor

readOnly.last() //Devuelve el último valor

println(readOnly) //[Lunes, Martes, Miércoles, Jueves, Viernes, Sábado, Domingo]

1. Listas en Kotlin
   1. ¿Qué es una lista?

R/:Una lista es un secuencia de elementos o nodos del mismo tipo donde cada elemento sabe cual es el siguiente nodo, si no hay ningún nodo apunta al NULL

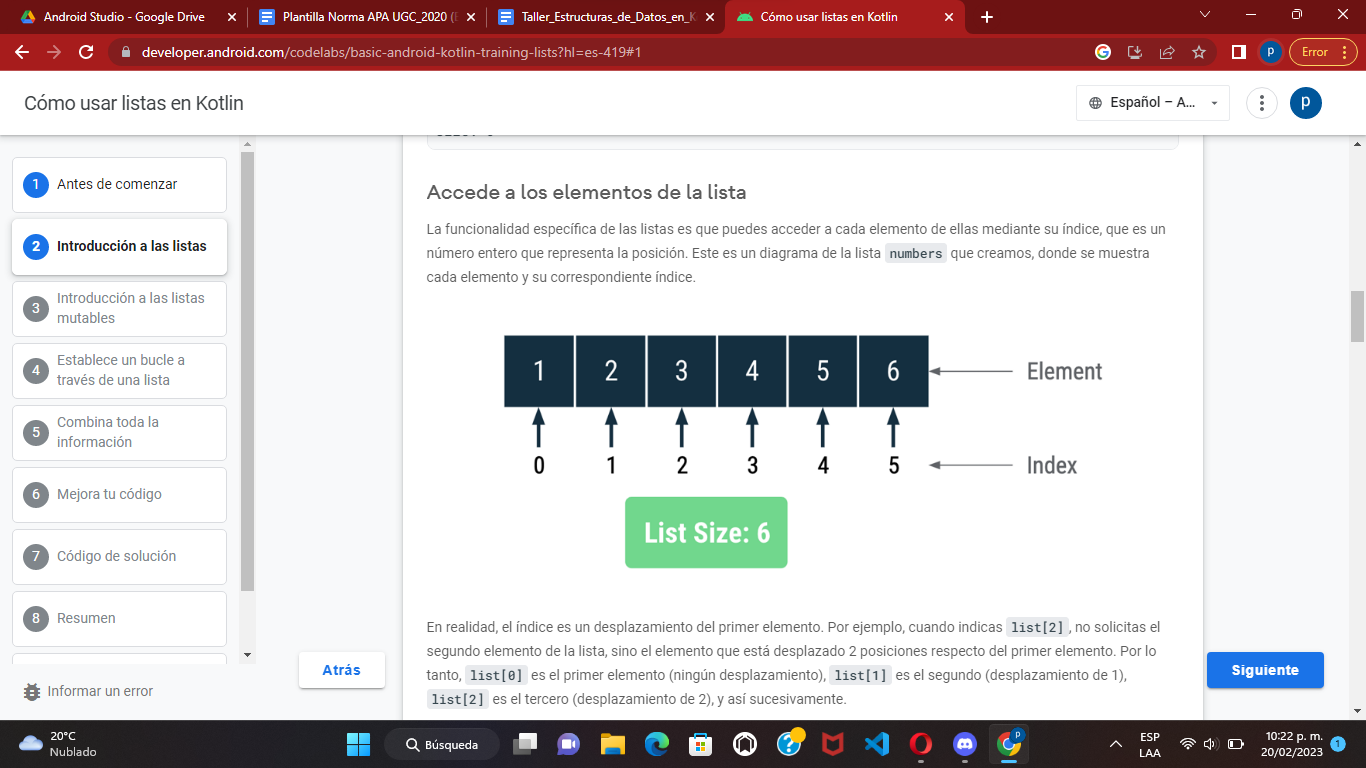
* 1. Creación de listas en Kotlin

R/:

En main() , crea una variable llamada numbers de tipo List<Int> porque contendrá una **lista** de solo lectura de números enteros. Crea una nueva clase List con la función de la biblioteca estándar de **Kotlin** listOf() y pasa los elementos de la **lista** como argumentos separados por comas

* 1. Accediendo a los elementos de una lista

R/:



* 1. Modificando los elementos de una lista

R/:

fun main() {

val numbers = listOf(1, 2, 3, 4, 5, 6)

println("List: $numbers")

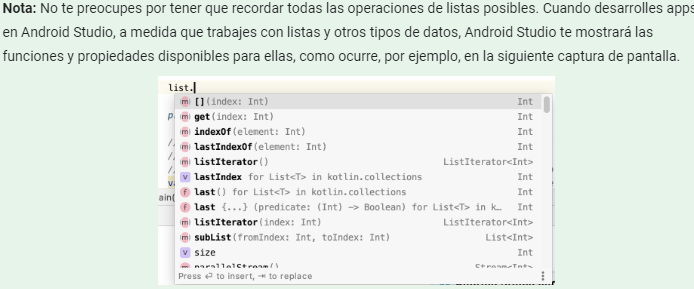
}

* 1. Recorriendo una lista

R/:List: [1, 2, 3, 4, 5, 6]

<https://developer.android.com/codelabs/basic-android-kotlin-training-lists?hl=es-419#1>

* 1. Funciones útiles para trabajar con listas en Kotlin

R/:

1. Conjuntos en Kotlin
   1. ¿Qué es un conjunto?

R/:Un conjunto es una colección de elementos sin orden, esto no puede tener ningún duplicado

* 1. Creación de conjuntos en Kotlin

R/:

val conjunto: Set<Int> = setOf(1, 3, 4)

<http://www.androidcurso.com/index.php/99-kotlin/925-colecciones-en-kotlin-list-set-y-map#:~:text=Un%20conjunto%20(Set)%20es%20una,un%20objeto%20de%20tipo%20Set>.

* 1. Accediendo a los elementos de un conjunto

R/:

fun main() {

val positiveNumbers = setOf(1, 2, 3, 4)

println(positiveNumbers)

val cities = setOf("Cali", "Armenia", "Cartagena", null)

println(cities)

val diamondCardsHand = setOf('A', 1, 2, 3, 'J', 'Q', 'K')

println(diamondCardsHand)

}

<https://www.develou.com/sets-en-kotlin/>

* 1. Modificando los elementos de un conjunto

R/:

* 1. Recorriendo un conjunto

R/:

* 1. Funciones útiles para trabajar con conjuntos en Kotlin

1. Mapas en Kotlin
   1. ¿Qué es un mapa?

R/:Un mapa es una colección que almacena sus elementos (entradas) en forma de pares clave-valor. Esto quiere decir que a cada clave le corresponde un solo valor y será única como si se tratase de un identificador

* 1. Creación de mapas en Kotlin

**Mapas De Solo Lectura**

solo te provee acceso de solo lectura, por lo que solo podrás usar comportamientos de consulta.La interfaz que representa a los mapas en Kotlin es Map<K,V>. Donde los parámetros de tipo K y V representan a los tipos para claves (propiedad keys) y valores (propiedad values).

## **Mapas Mutables**

## Los mapas mutables te otorgan el poder de usar comandos de operaciones sobre los elementos cómo agregar, actualizar y remover entradas. El diseño de esta figura la encuentras en la interfaz MutableMap<K,V>, la cual extiende de Map<K,V>.

mapOf() crea un mapa inmutable compuesto por una lista de pares, donde el primer valor es la clave, y el segundo es el valor. Devuelve un objeto de tipo Map.

mutableMapOf() permite crear un mapa mutable sin indicar la implementación:

* 1. Accediendo a los elementos de un mapa

* 1. Modificando los elementos de un mapa
  2. Recorriendo un mapa
  3. Funciones útiles para trabajar con mapas en Kotlin

<http://www.androidcurso.com/index.php/99-kotlin/925-colecciones-en-kotlin-list-set-y-map#:~:text=mapOf()%20crea%20un%20mapa,un%20objeto%20de%20tipo%20Map.&text=Podemos%20obtener%20el%20valor%20de,un%20atajo%20para%20get()>.

<https://www.develou.com/mapas-en-kotlin/>

1. Pares en Kotlin
   1. ¿Qué es un par?

R/:sí estamos hablando de un par significa que es un número divisible por dos, pero si es un class pair significa que simplemente son una representación genérica (cualquier tipo de datos o clases) de dos valores (pares). Los data class Pair son una estructura que permite guardar dos valores

* 1. Creación de pares en Kotlin
  2. Accediendo a los elementos de un par
  3. Modificando los elementos de un par
  4. Recorriendo un par
  5. Funciones útiles para trabajar con pares en Kotlin

1. Prácticas de estructuras de datos en Kotlin
   1. Ejercicios prácticos para aplicar los conceptos aprendidos
   2. Solución a los ejercicios prácticos

Recursos adicionales: Documentación oficial de Kotlin: https://kotlinlang.org/docs/reference/

Entrega. Se deberá realizar la entrega de un informe con la solución de los puntos anteriores, el aprendiz acompañará la investigación con ejemplos prácticos de cada estructura y deberá publicar el código fuente en un repositorio en GitHub.