**Taller Estructuras de Datos en Kotlin**

Yvette Daniela Campo Osorio

Cristhian David Henao Hoyos



Analisis y Desarrollo de Sistemas de Información

SENA

Armenia, Quindio

2023

**Introducción a las estructuras de datos en Kotlin**

Las estructuras de datos son aquellas que permiten organizar la información de manera ordenada, eficientemente y diseñar la solución correcta para un determinado problema. Utilizar estructuras de datos en Kotlin tiene grandes ventajas como lo son:

* Interoperabilidad con código Java.
* Curva de aprendizaje fácil.
* Tiempo de programación reducido.
* Desarrollo multiplataforma.
* Flexibilidad

Kotlin y Java cuentan con varias diferencias como: 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.

# Argumentos

## Arreglos en Kotlin

Un arreglo es un conjunto de múltiples datos o objetos guardados en una variable. Para crear arreglos en kotlin se usa la clase genérica Array<T> para representar los arreglos. Para acceder a los elementos de un arreglo podemos hacer uso del el método get(); el método get() toma un elemento Int como parámetro y muestra el elemento. Los elementos de un arreglo se modifican de la siguiente manera: arreglo[0] = valor o val elemento = arreglo[0]

Un arreglo se puede recorrer con un ciclo for:

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

for (nombre in nombres) {

println(nombre)}

Las siguientes son funciones de mucha utilidad para trabajar con arreglos en kotlin:

* 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]

## Listas en Kotlin

Una lista es una secuencia de elementos o nodos del mismo tipo donde cada elemento sabe cuál es el siguiente nodo, si no hay ningún nodo apunta al NULL. Las listas en Kotlin se crean generando una variable 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(). Para acceder a los elementos de una lista se usa list[índice] en índice se solicita el elemento q esta desplazado las posiciones respecto a el primer elemento. Para modificar los elementos de una lista se hace de la siguiente manera:

fun main() {

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

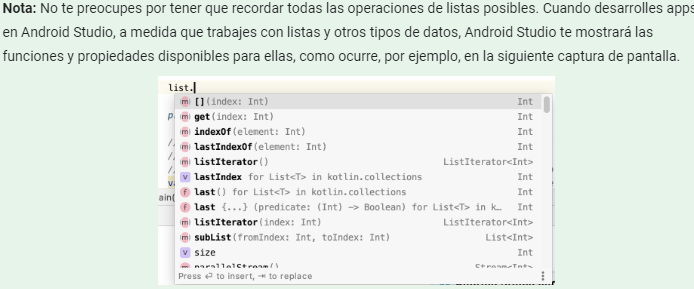
println("List: $numbers")

}

Para recorriendo una lista se hace de la siguiente manera:

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

A continuación un pequeño listado de funciones útiles para usar con listas



**Conjuntos en Kotlin**

Un conjunto es una colección de elementos sin orden, esto no puede tener ningún duplicado. La creación de conjuntos en kotlin se hace de la siguiente manera:

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

Ejemplo para acceder a los elementos de un conjunto

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)

}

**Mapas en Kotlin**

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. Es posible hacer diferentes tipos de mapas como lo son los Mapas De Solo Lectura que solo te proveen 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:

# Lista de Referencia o Bibliografía

https://blog.soyhenry.com/que-es-una-estructura-de-datos-en-programacion/#:~:text=¿Para%20qué%20sirven%20las%20estructuras,correcta%20para%20un%20determinado%20problema.

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

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

<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>.

<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/>

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