**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.**

Luisa F. Restrepo Gaviria

Tecnología en análisis y desarrollo de sistemas de información, servicio nacional de aprendizaje

2469285: Taller Estructuras de Datos en Kotlin

Inst. Cristian David Henao

20 de febrero de 2023

realizar un informe donde se evidencien los siguientes puntos:

1. Introducción a las estructuras de datos en Kotlin

**a. ¿Qué son las estructuras de datos y para qué se utilizan?**

R/ Las estructuras de datos son una de las formas comunes para representar información, tal cual como usamos una variable de tipo Array para representar un nu3finito de aquel elemento, también podemos representar en una estructura de datos una lista pero de tipo enlazada, es decir, la estructura puede ser creada por uno mismo o prevista por una librería y se utilizan para resolver algún problema de manera más sencilla gracias a las reglas qué tiene y nunca cambian, es decir que la mayoría de cosas son verdaderas.

**b. Ventajas de utilizar estructuras de datos en Kotlin**

R/ 1. Es orientado a objetos lo más habitual en el mundo del desarrollo de aplicaciones móviles, esto era un paradigma de POO, gracias a kotlin se ha venido quitando esos esquemas para demostrar que la POO se puede trabajar de la mano a la programación funcional.

2. Es integrado con Android Studio, esto da el hecho de que todo pueda funcionar desde el primer momento para Android es una gran ventaja, ya que sin saber nada de kotlin se puede tener un proyecto montado y listo para trabajar en tiempo inesperado sin ningún problema.

**c. Diferencias entre las estructuras de datos en Kotlin y Java**

R/ las direfencias entre la estructura de datos de kotlin y la de java es que se trata de un lenguaje que, al igual que Java, está dedicado a objetos, pero con la diferencia de que posee características para ser completamente funcional, es decir, el escrito de código es bastante limpio y conciso, no requiere de tantas líneas de código como si lo requiere Java, por lo que es más práctico y requiere menos tiempo y por último es un lenguaje que está bien respaldado por empresas como Google y JetBrains. Por lo cual es un lenguaje que tiene un buen futuro hablando de kotlin.

2. Arreglos en Kotlin

**a. ¿Qué es un arreglo?**

R/Un arreglo es una estructura de datos que almacenan un conjunto de elementos de un mismo tipo en una variable única, estos alimentos se pueden almacenar en posiciones consecutivas y en cada elemento se puede acceder con un índice numérico.

**b. Creación de arreglos en Kotlin**

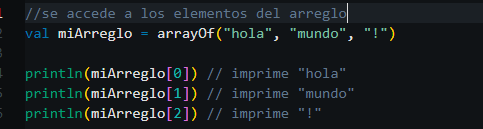
R/ Para crear los arreglos en kotlin se puede usar el constructor array() o también se puede usar la función arrayOf().

Texto

Descripción generada automáticamente

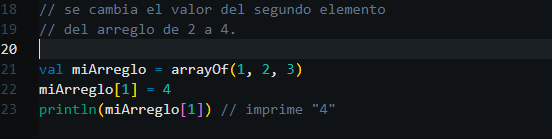
**c. Accediendo a los elementos de un arreglo**

R/ Para poder acceder a los elementos de un arreglo se debe utilizar el operador de corchetes[], seguido del índice del elemento en donde este empieza en cero para el primer elemento del arreglo.



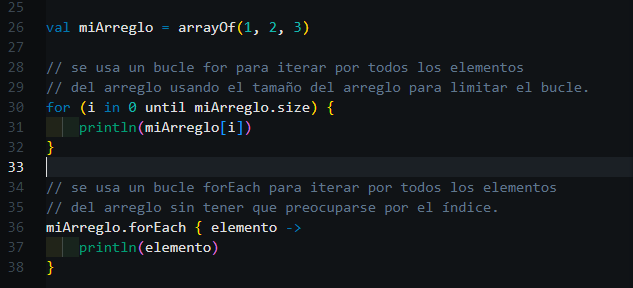
**d. Modificando los elementos de un arreglo**

R/ Para modificar los elementos de un arreglo se puede hacer uso del operador de corchetes[] y por ende asignar un nuevo valor.



**e. Recorriendo un arreglo**

R/ Para recorrer un arreglo es posible usar un bucle for o también un bucle forEach.



**f. Funciones útiles para trabajar con arreglos en Kotlin**

R/ Existen muchas funciones útiles para trabajar en kotlin algunas son:

* First(): devuelve el primer elemento del arreglo
* Last(): devuelve el ultimo elemento del arreglo
* Reverse(): invierte el orden de los elementos
* Sort(): ordena los elementos del arreglo en orden ascendente
* Size(): devuelve el numero de elementos en el arreglo

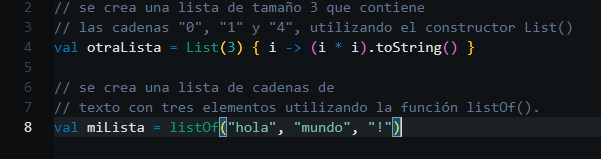
3. Listas en Kotlin

**a. ¿Qué es una lista?**

R/Una lista es una estructura de datos que almacena un conjunto de elementos con una secuencia de ordenada, estas pueden disminuir y crecer de tamaño dinámicamente, el cual se hacen más flexibles para ciertas situaciones.

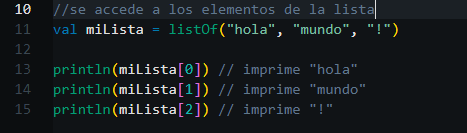
**b. Creación de listas en Kotlin**

R/ Para poder crear listas en kotlin, se puede usar el constructor list() o la función listOf().



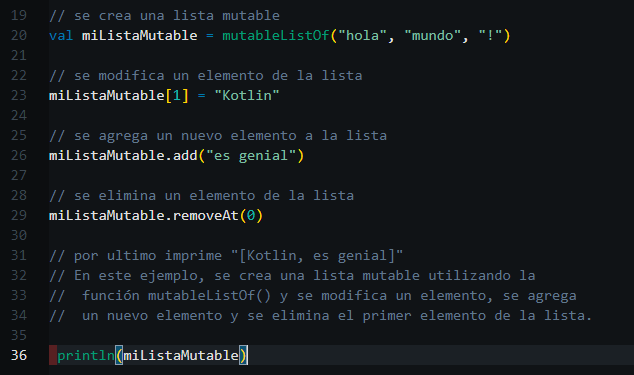
**c. Accediendo a los elementos de una lista**

R/ Para acceder a los elementos de una lista, se utiliza el operador de corchetes [] seguido del índice del elemento. El índice comienza en cero para el primer elemento de la lista.



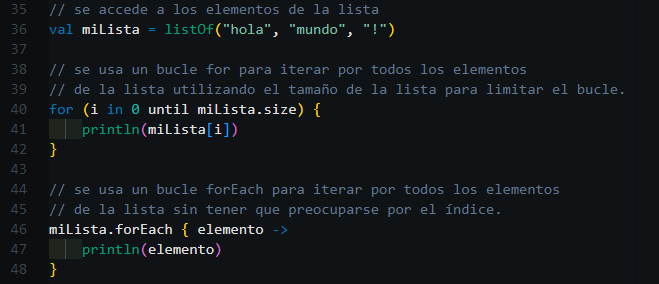
d. Modificando los elementos de una lista

R/ En diferencia de los arreglos, las listas son inmutables por defecto en Kotlin, lo que significa que no se pueden modificar directamente los elementos de la lista. Sin embargo, se puede crear una lista mutable utilizando la función mutableListOf() o el constructor MutableList(), que permite agregar, eliminar o modificar elementos en la lista.



**e. Recorriendo una lista**

R/ Para recorrer una lista, se puede usar un bucle for o un bucle forEach.



**f. Funciones útiles para trabajar con listas en Kotlin**

R/ Existen muchas funciones útiles para trabajar en kotlin algunas son:

size: devuelve el número de elementos en la lista.

first(): devuelve el primer elemento de la lista.

last(): devuelve el último elemento de la lista.

contains(elemento): devuelve true si el elemento está presente en la lista.

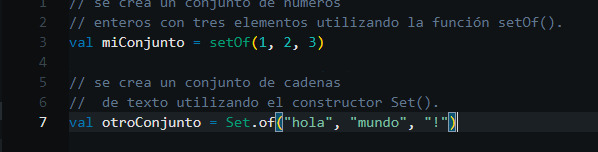
4. Conjuntos en Kotlin

**a. ¿Qué es un conjunto?**

R/ Un conjunto es una colección de elementos sin orden y sin duplicados. Los elementos en un conjunto son únicos y no se pueden repetir.

**b. Creación de conjuntos en Kotlin**

R/ Para crear un conjunto en Kotlin, se puede usar la función setOf() o el constructor Set().



**c. Accediendo a los elementos de un conjunto**

R/ Para acceder a los elementos de un conjunto, se puede utilizar el operador de corchetes [] seguido del elemento que se desea acceder. Sin embargo, como los conjuntos no tienen un orden definido, no es posible acceder a los elementos por índice.

Texto

Descripción generada automáticamente

**d. Modificando los elementos de un conjunto**

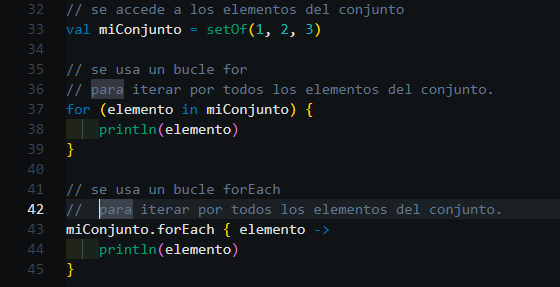
R/ En diferencia a los arreglos y las listas, los conjuntos son inmutables por defecto en Kotlin, lo que significa que no se pueden modificar directamente los elementos del conjunto. Sin embargo, se puede crear un conjunto mutable utilizando la función mutableSetOf() o el constructor MutableSet(), que permite agregar, eliminar o modificar elementos en el conjunto.

Texto

Descripción generada automáticamente

**e. Recorriendo un conjunto**

R/ Para recorrer un conjunto, se puede usar un bucle for o un bucle forEach.



**f. Funciones útiles para trabajar con conjuntos en Kotlin**

R/ Existen muchas funciones útiles para trabajar en kotlin algunas son:

size: devuelve el número de elementos en el conjunto.

contains(elemento): devuelve true si el elemento está presente en el conjunto.

union(conjunto): devuelve un nuevo conjunto que contiene todos los elementos de ambos conjuntos.

5. Mapas en Kotlin

**a. ¿Qué es un mapa?**

R/ Un mapa es una colección de pares clave-valor, donde cada clave es única y se utiliza para acceder al valor asociado. En un mapa, los elementos no tienen un orden definido.

**b. Creación de mapas en Kotlin**

R/ Para crear un mapa en Kotlin, se puede utilizar la función mapOf() o el constructor Map().

Texto

Descripción generada automáticamente

**c. Accediendo a los elementos de un mapa**

R/ Para acceder a los elementos de un mapa, se puede utilizar el operador de corchetes [] seguido de la clave que se desea acceder.

Texto

Descripción generada automáticamente

**d. Modificando los elementos de un mapa**

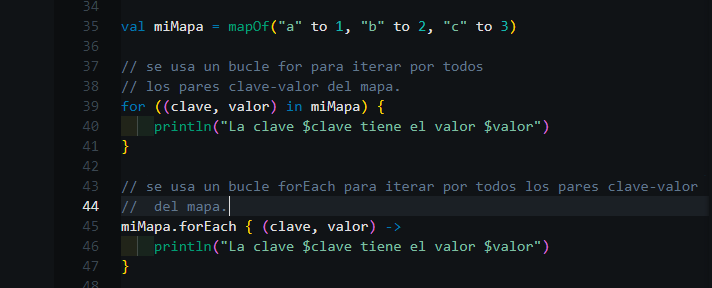
R/ Al igual que los conjuntos, los mapas también son inmutables por defecto en Kotlin. Sin embargo, se puede crear un mapa mutable utilizando la función mutableMapOf() o el constructor MutableMap(), que permite agregar, eliminar o modificar pares clave-valor en el mapa.

Texto

Descripción generada automáticamente

**e. Recorriendo un mapa**

R/ Para recorrer un mapa, se puede usar un bucle for o un bucle forEach. En ambos casos, se puede acceder tanto a las claves como a los valores.



**f. Funciones útiles para trabajar con mapas en Kotlin**

R/ Existen muchas funciones útiles para trabajar en kotlin algunas son:

size: devuelve el número de pares clave-valor en el mapa.

isEmpty: devuelve true si el mapa está vacío, false de lo contrario.

containsKey: devuelve true si el mapa contiene una clave dada, false de lo contrario.

containsValue: devuelve true si el mapa contiene un valor dado, false de lo contrario.

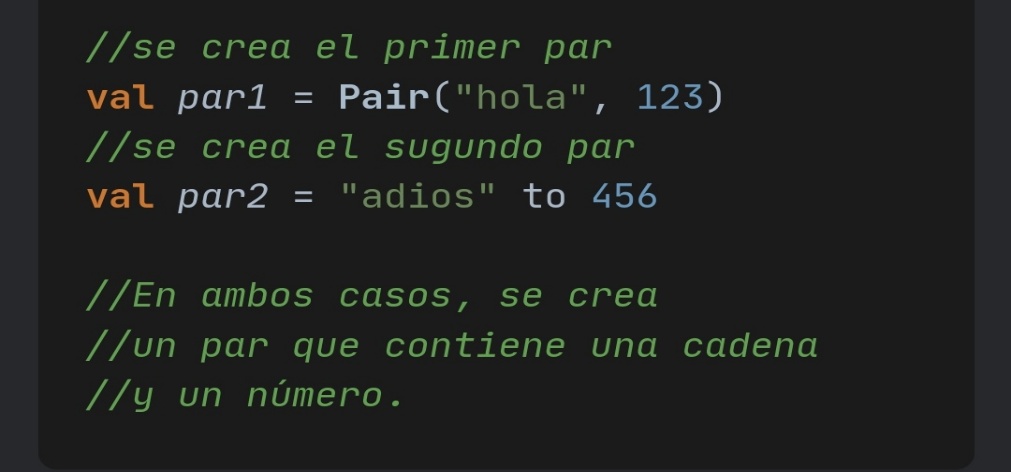
get: devuelve el valor asociado con una clave dada, o null si la clave no está en el mapa.

6. Pares en Kotlin

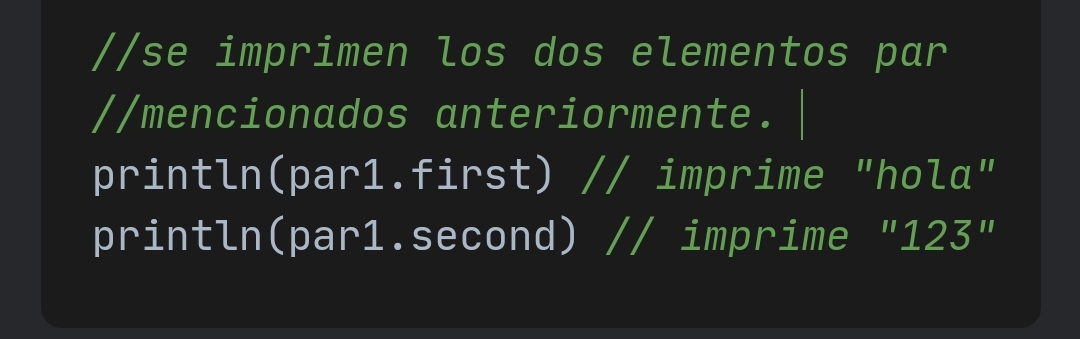
**a. ¿Qué es un par?**

R/ En Kotlin, un par es una estructura de datos que almacena dos valores relacionados. Es una forma simple de agrupar dos valores y tratarlos como una sola entidad.

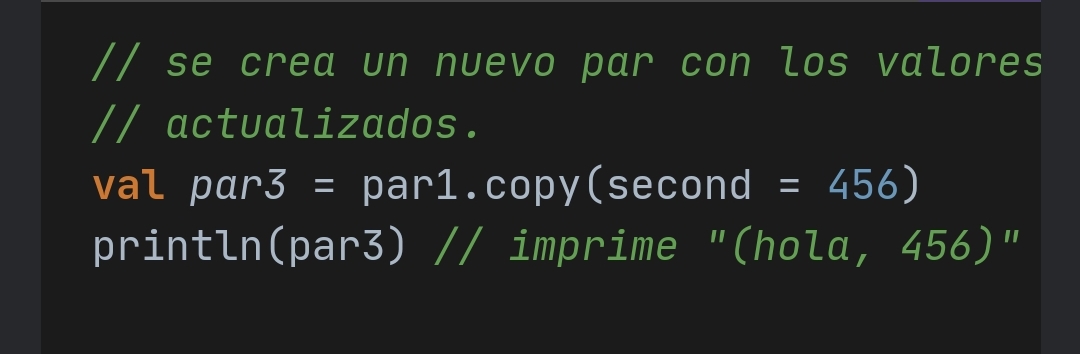
**b. Creación de pares en Kotlin**

R/ En Kotlin, un par se puede crear utilizando la función Pair o el operador especial to.

**c. Accediendo a los elementos de un par**

R/ Para acceder a los elementos de un par, se pueden utilizar las propiedades first y second.

**d. Modificando los elementos de un par**

R/ Como los pares son inmutables, no se pueden modificar directamente. En su lugar, se puede crear un nuevo par con los valores actualizados.

**e. Recorriendo un par**

R/ Como un par solo contiene dos valores, no es necesario recorrerlo.

**f. Funciones útiles para trabajar con pares en Kotlin**

R/ Existen muchas funciones útiles para trabajar en kotlin algunas son:

Component1 y component2: son funciones que devuelven el primer y segundo valor de un par, respectivamente.

toList: convierte un par en una lista de dos elementos.

toMap: convierte un par en un mapa con una única entrada.

Son especialmente útiles en el contexto de la desestructuración, la cual es una forma de extraer los valores individuales de un par o cualquier otra estructura de datos y asignarlos a variables separadas.

pair**:** Esta función permite crear un nuevo par.

to**:** Esta función es una forma más legible de crear un nuevo par.

7. Prácticas de estructuras de datos en Kotlin

**a. Ejercicios prácticos para aplicar los conceptos aprendidos**

R/ 1. Dado un arreglo de 12 posiciones y luego imprímalos.

2. Dada una lista de números, imprime el resultado de la suma de todos.

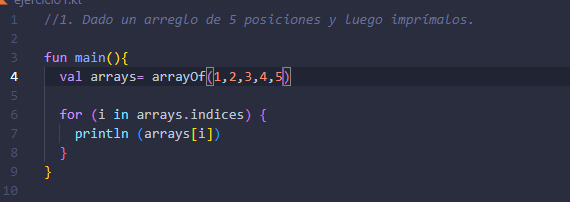
3. Dado un conjunto de palabras, utilizar una condición para tener acceso a un elemento y luego imprime esa lista.

4. Dado un mapa de nombres asígneles un valor e imprima todos los nombres con dicho valor.

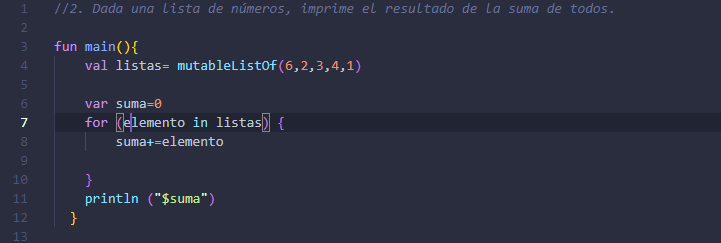
5. Dado un par de elementos , acceder a un solo elemento e imprime el resultado .

**b. Solución a los ejercicios prácticos**

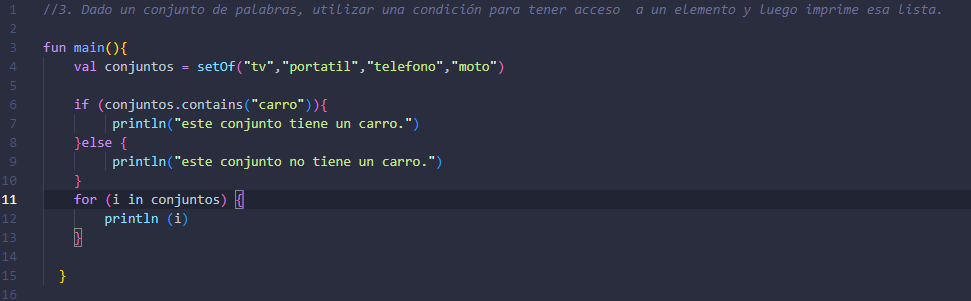
R/1.

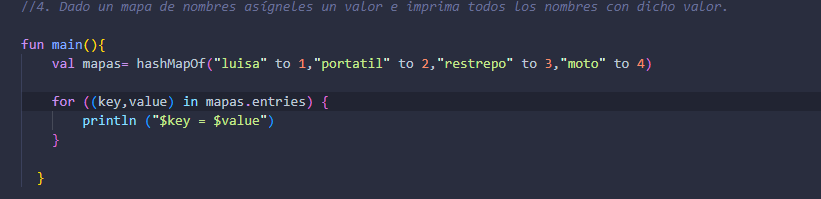


2.



3.



4.

5.