

**Taller Estructuras de Datos en Kotlin**

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 evidencien 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// las estructuras de datos te permiten almacenar varios valores, generalmente del mismo tipo de datos, de manera organizada.

* 1. Ventajas de utilizar estructuras de datos en Kotlin

R//Una de las ventajas que tiene utilizar estructuras de datos es que facilita la organización de todos los datos y que su sintaxis es fácil de aprender y de hacer.

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

R//Kotlin es más conciso que Java, se calcula que, con este lenguaje, se reduce un 40% de líneas de código en comparación con Java. Interoperabilidad: Kotlin es altamente interoperable con Java.

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

R// Un arreglo es una estructura con valores de datos, que están almacenados de forma contigua en memoria. Todos los elementos son referenciados por un mismo nombre y tienen el mismo tipo de dato.

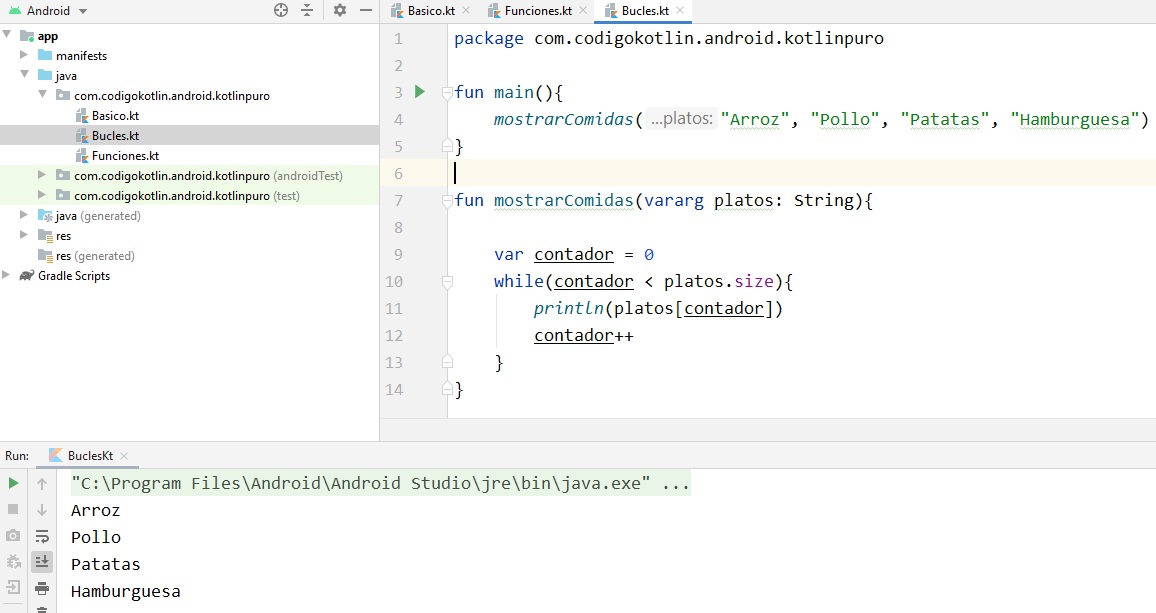
* 1. Creación de arreglos en Kotlin

R// Kotlin usa la clase genérica Array<T> para representar arreglos. Crear instancias con un tipo parametrizado usa los siguientes métodos: arrayOf (vararg elements: T): recibe un argumento variable con elementos de tipo T y retorna el arreglo que los contiene.

* 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. Recorriendo un arreglo

R// 

* 1. Funciones útiles para trabajar con arreglos en Kotlin

R// Definir arreglo en Kotlin usando arrayOf

Declarar arreglo con Array

Recorrer arreglos

Longitud del arreglo

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

R// Una lista es una colección de elementos con un orden específico. Hay dos tipos de listas en Kotlin: Lista de solo lectura: List no se puede modificar después de su creación. Lista mutable: MutableList se puede modificar después de su creación, lo que significa que puedes agregar, quitar o actualizar sus elementos.

* 1. Creación de listas en Kotlin

R// Crea una lista

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

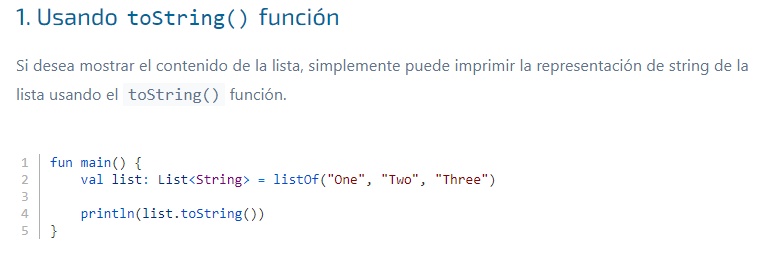
R// Para acceder al estado de la lista puedes usar los siguientes miembros básicos:

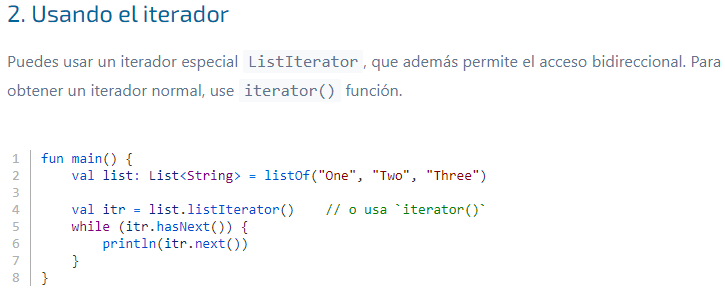
* size para obtener la cantidad de elementos de la lista
* lista[index] para obtener el elemento ubicado en index. Esta es la construcción para el operador de acceso posicional get(index)
* indexOf(element) para obtener el índice de la primera ocurrencia de element
* lastIndexOf(element) para obtener el índice de la última ocurrencia del element
* subList(fromIndex, toIndex) para obtener una porción de la lista en el rango
  1. Modificando los elementos de una lista

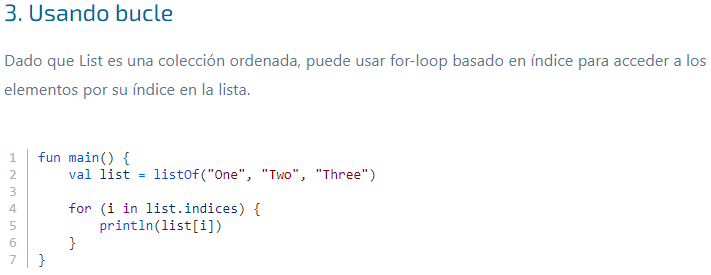
R// [index]=element, para reemplazar un ítem en el índice. Esta construcción es equivalente al operador set(index, element)

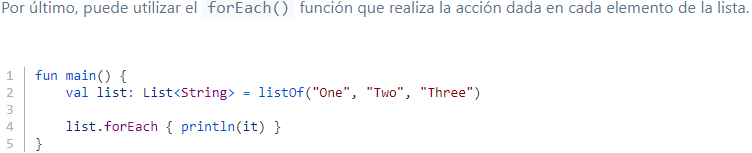
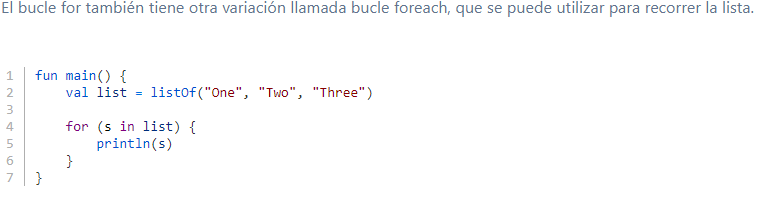
* 1. Recorriendo una lista

R//





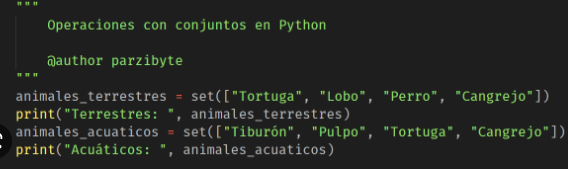




1. Conjuntos en Kotlin
   1. ¿Qué es un conjunto?

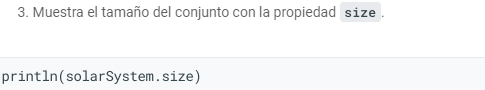
R// Un conjunto (Set) es una colección sin orden de elementos únicos, esto es, no puede tener ningún duplicado. Set extiende de la interfaz Collection, por lo que es inmutable, y no agrega nuevos métodos.

* 1. Creación de conjuntos en Kotlin

R// Para crear conjuntos inmutables utiliza: setOf() crea un conjunto inmutable y devuelve un objeto de tipo Set. 

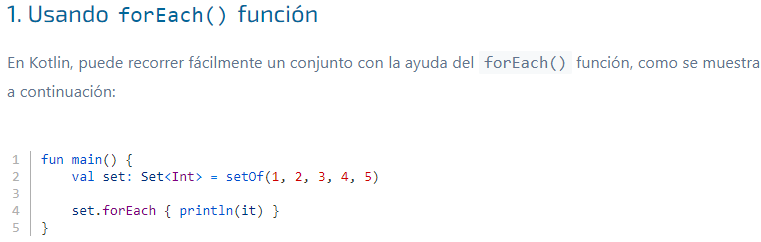
* 1. Accediendo a los elementos de un conjunto

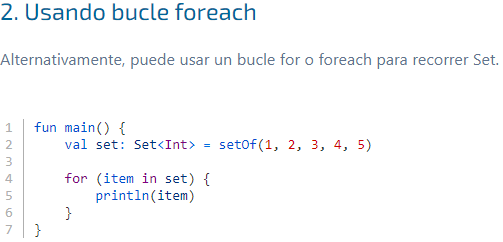
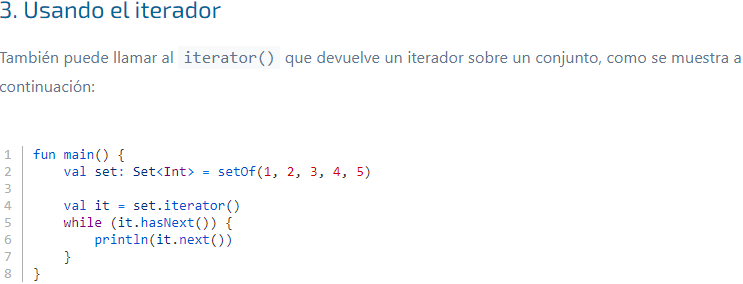
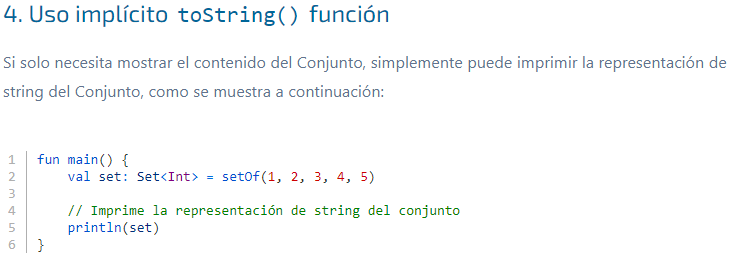
R//



* 1. Modificando los elementos de un conjunto

R// Usa la función remove() para quitar "Pluto".

* 1. Recorriendo un conjunto

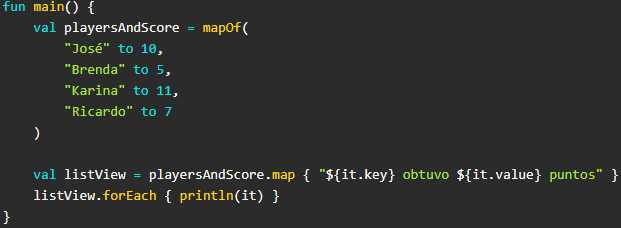
1. Mapas en Kotlin
   1. ¿Qué es un mapa?

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

R//Para crear un Map en Kotlin debemos definir una variable e indicar de que tipo de datos son la clave del mapa y el valor que almacena.

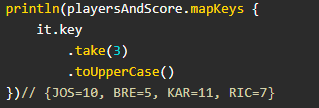
Mediante la función mapOf retornamos un Map indicando cada entrada en nuestro Map mediante un objeto Pair:



* 1. Accediendo a los elementos de un mapa

R// La librería estándar te provee funciones para transformar a las claves y valores por separado.

Usa mapKeys() para obtener un nuevo mapa cuyas claves serán las que especifiques en función de transformación. Los valores serán los mismos del mapa original:



* 1. Modificando los elementos de un mapa

R// 

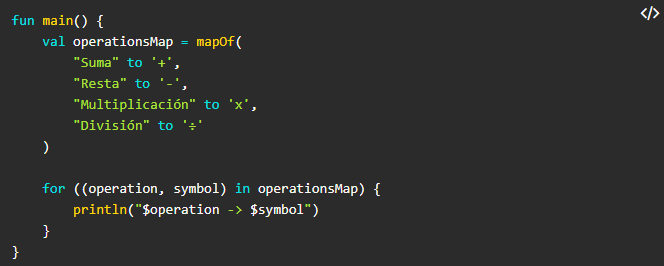


La función de extensión to() usa el modificador infix para permitirte crear pares de forma legible:



* 1. Recorriendo un mapa

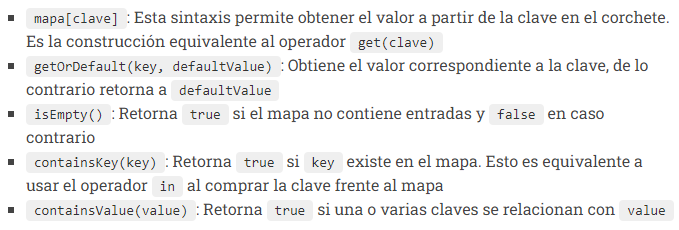
R//



* 1. Funciones útiles para trabajar con mapas en Kotlin

R// 



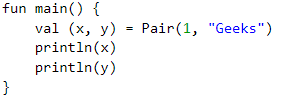


1. Pares en Kotlin
   1. ¿Qué es un par?

R// Los data class Pair son una estructura que permite guardar dos valores. Esto en Android es muy útil ya que podemos emplear los Pair Para guardar una pareja de valores, por ejemplo un usuario y contraseña o cualquier pareja de valores que tengan alguna relación.

* 1. Creación de pares en Kotlin

R//



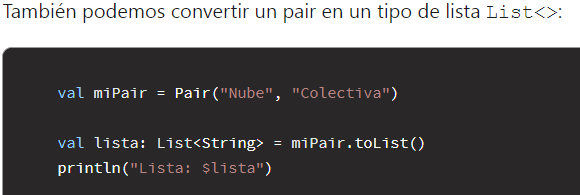
* 1. Accediendo a los elementos de un par

R//Devuelve la representación de cadena del par, incluidos su primer y segundo valor.



* 1. Recorriendo un par

R//



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

Recursos adicionales:

Taller Estructuras de Datos en Kotlin 1

Documentación oficial de Kotlin: [**https://kotlinlang.org/docs/reference/**](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 practicos de cada estructura y deberá publicar el código fuente en un repositorio en GitHub.

Taller Estructuras de Datos en Kotlin 2