Estructuras de datos en kotlin

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

- Las estructuras de datos son una forma de organizar información dentro del computador para que puedan ser utilizados de manera ágil
  b. Ventajas de utilizar estructuras de datos en Kotlin
- Estas permiten acceder y manipular los datos de manera ágil.
- También permiten el almacenamiento de diferentes tipos de datos de manera organizada y estructurada .
- Permite la reutilización de estas.

c. Diferencias entre las estructuras de datos en Kotlin y C#

- En su sintaxis en kotlin se usa la palabra reservada “arrayOf” y en kotlin “new” junto con el tipo de datos y la longitud de este.
- Kotlin admite tipos de datos nulos y no nulos y C# por el contrario no.

2. Arreglos en Kotlin
   a. ¿Qué es un arreglo?

- Es una estructura de datos que almacena datos del mismo tipo en secuencia en la memoria,
  b. Creación de arreglos en Kotlin

```
val miArreglo = arrayOf("samuel","pablo","camilo")
```

- c. Accediendo a los elementos de un arreglo

```
print(miArreglo[0])
```

```
[Running] cd "e:\Sena\Cristian 5T\Kotlin\Estr
samuel
[Done] exited with code=0 in 23.325 seconds
```

- d. Modificando los elementos de un arreglo

```
miArreglo[2]="cristian"

println(miArreglo[2])
```

```
[Running] cd "e:\Sena\Cristian 5T\Kotlin\Estr
cristian

[Done] exited with code=0 in 22.035 seconds
```

- e. Recorriendo un arreglo

```
miArreglo.forEach{ recorrer ->
    println(recorrer)
}
```

```
[Running] cd "e:\Sena\Cristian 5T\Kotlin\Estr
samuel
pablo
cristian

[Done] exited with code=0 in 16.722 seconds
```

- f. Funciones útiles para trabajar con arreglos en Kotlin

```
val tamaño = miArreglo.size
println(tamaño)
//esta sirve para mostrar el tamaño de un arreglo
```

```
[Running] cd "e:\Sena\Cristian 5T\Kotlin\Estr
3

[Done] exited with code=0 in 15.554 seconds
```

```
miArreglo.set(1,"daniel")
println(miArreglo[1])
//este sirve para cambiar una posicion e un arreglo
```

```
[Running] cd "e:\Sena\Cristian 5T\Kotlin\Est
daniel

[Done] exited with code=0 in 20.899 seconds
```

- Existen muchas mas dependiendo el tipo de dato que contenga el arreglo

3. Listas en Kotlin
   a. ¿Qué es una lista?

- Estas son una estructura de datos que permite añadir o eliminar cualquier elemento que contenga.
  b. Creación de listas en Kotlin

```
var lista: List<Int> = listOf(1,2,3,4,5,6)
```

- c. Accediendo a los elementos de una lista

```
println(lista[4])
```

```
[Running] cd "e:\Sena\Cristian 5T\Kotlin\Est
5

[Done] exited with code=0 in 11.028 seconds
```

- d. Modificando los elementos de una lista

```
//se debe cambiar el tipo de lista a la siguiente manera
var lista: MutableList<Int> = mutableListOf(1,2,3,4,5,6)

lista[4]= 44

println(lista)
```

```
[Running] cd "e:\Sena\Cristian 5T\Kotlin\Estr
[1, 2, 3, 4, 44, 6]

[Done] exited with code=0 in 13.135 seconds
```

- e. Recorriendo una lista

```
lista.forEach{recorrer ->
    println(recorrer)
}
```

```
[Running] cd "e:\Sena\Cristian 5T\Kotlin\Estru
1
2
3
4
5
6

[Done] exited with code=0 in 11.398 seconds
```

- f. Funciones útiles para trabajar con listas en Kotlin

```
//Esta sirve para agregar varios valores al final de la lista
lista.addAll(listOf(7,8,9,10))

println(lista)
```

```
[Running] cd "e:\Sena\Cristian 5T\Kotlin\Est
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

[Done] exited with code=0 in 18.026 seconds
```

```
// Elimina el primer elemento de la lista que coincida con el elemento dado.
lista.remove(10)
println(lista)
```

```
[Running] cd "e:\Sena\Cristian 5T\Kotlin\Est
[1, 2, 3, 4, 5, 6, 7, 8, 9]

[Done] exited with code=0 in 14.083 seconds
```

4. Conjuntos en Kotlin
   a. ¿Qué es un conjunto?

- Es una colección de elementos que no admite elementos duplicados y no tienen un orden especifico

  b. Creación de conjuntos en Kotlin

```
//conjunto mutable
val conjunto = mutableSetOf("Daniel","Samuel","Pablo")
```

- c. Accediendo a los elementos de un conjunto

```
// accediendo al primer elemento del conjunto
val primerElemento = conjunto.first()

println(primerElemento)
```

```
Daniel

[Done] exited with code=0 in 9.536 seconds
```

- d. Modificando los elementos de un conjunto

```
// Agregar elementos al conjunto mutable
conjunto.add("Cristofert")//agregando un elemento
conjunto.addAll(listOf("Camilo", "Yuliam"))//agregando varios elementos
```

- e. Recorriendo un conjunto

```
for(recorrer in conjunto){
    println(recorrer)
}
```

```
Daniel
Samuel
Pablo
Cristofert
Camilo
Yuliam

[Done] exited with code=0 in 12.485 seconds
```

- f. Funciones útiles para trabajar con conjuntos en Kotlin

```
//esta sirve para saber la cantidad de elementos que hay en un conjunto
var cantidadcon = conjunto.size;
println(cantidadcon)
```

```
6

[Done] exited with code=0 in 13.386 seconds
```

-

5. Mapas en Kotlin
   a. ¿Qué es un mapa?

- Esta permite almacenar y recuperar valores asociados a claves unicas
  b. Creación de mapas en Kotlin

```
//creacion de un mapa mutable
val mapaMutable = mutableMapOf(
"Daniel" to 18,
"Cristofert" to 19,
"Luisa" to 20,
"Cristian" to 17
```

- c. Accediendo a los elementos de un mapa

```
println(edades["Daniel"])
```

- d. Modificando los elementos de un mapa

```
edades["Cristian"]= 21
println(edades["Cristian"])
```

```
21

[Done] exited with code=0 in 15.874 seconds
```

- e. Recorriendo un mapa

```
for ((name, age) in edades) {
    println("Nombre: $name - Edad: $age")
}
```

```
Nombre: Daniel - Edad: 18
Nombre: Cristofert - Edad: 19
Nombre: Luisa - Edad: 20
Nombre: Cristian - Edad: 17
```

- f. Funciones útiles para trabajar con mapas en Kotlin

```
//este sirve para obtener un valor a partir de su clave
val valor = edades.get("Cristofert")

println(valor)
```

```
19

[Done] exited with code=0 in 13.114 seconds
```

-

7. Pares en Kotlin
   a. ¿Qué es un par?

- Un par almacena dos valores relacionados entre si
  b. Creación de pares en Kotlin

```kotlin
//creando un par mediante la funcion pair()
val info: Pair<String, Int> = Pair("Samuel", 18)
```

- c. Accediendo a los elementos de un par

```kotlin
//accedineod a los valores del par
println(info.first)
println(info.second)
```

- d. Modificando los elementos de un par

```kotlin
//creando una copia del para y asi modificarla
var info2 = info.copy(second = 20)
println(info2)
```

- e. Recorriendo un par

```kotlin
for (valor in info) {
    println(valor)
}
```

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

```kotlin
    //esta sirve ara saber si algun dato del par es igual a otro
println(info.equals(info2))
```

```
false

[Done] exited with code=0 in 10.067 seconds
```

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

E**jecicio array**: Cree un array con los planetas e imprimalos

```kotlin
val miArreglo = arrayOf("Mercurio","Venus","Tierra","Júpiter","Saturno","Urano","Neptuno","Plutón")

miArreglo.forEach{ recorrer ->
    println(recorrer)
}
```

```
Mercurio
Venus
Tierra
J?piter
Saturno
Urano
Neptuno
Plut?n
```

Ejercicio listas: Crer una lista con el nombre de 6 personas y modificar algun nombre

```kotlin
var lista: MutableList<String> = mutableListOf("Samuel", "Pablo", "Daniel", "Critian", "Alvaro", "Willer")

//se cambio el nombre de Alvaro por Oscar
lista[4]= "Oscar"

lista.forEach{recorrer ->
    println(recorrer)
}
```

```
Samuel
Pablo
Daniel
Critian
Oscar
Willer
```

Ejercicio conjuntos: Cree un conjunto, utilice una condición para acceder a un elemento y luego imprima el conjunto.

```kotlin
val conjunto = mutableSetOf("Daniel","Samuel","Pablo","Cristofert","Juan")

if(conjunto.contains("Camilo")){
    println("El conjunto contiene Camilo")
}else{
    println("El conjunto no contiene Camilo")
}

for(recorrer in conjunto){
    println(recorrer)
}
```

```
El conjunto no contiene Camilo
Daniel
Samuel
Pablo
Cristofert
Juan
```

Ejercicio mapa: Cree un mapa, asígnele a cada clave un valor y luego imprima cada clave con su respectivo valor.

```kotlin
val edades = mutableMapOf(
"Daniel" to 18,
"Cristofert" to 19,
"Luisa" to 20,
"Cristian" to 17,
"Sara" to 16
)

for ((name, age) in edades) {
    println("Nombre: $name - Edad: $age")
}
```

```
Nombre: Daniel - Edad: 18
Nombre: Cristofert - Edad: 19
Nombre: Luisa - Edad: 20
Nombre: Cristian - Edad: 17
Nombre: Sara - Edad: 16
```

Ejercicio par: Cree un par, accede a los 2 elementos del conjunto e imprímelos.

```kotlin
val info: Pair<String, Int> = Pair("Juan Felipe", 18)

println(info)
```

```
(Juan Felipe, 18)
```