Módulo Profesional: Big Data Aplicado

Scala

Strings

```scala
val str: String  ="Hola, yo estoy aprendiendo Scala"

//1. charAt --> accede al carácter en el índice 2 de la cadena
println(str.charAt(2))

//2. substring --> devuelve los carácteres de la posición 7 hasta 11
println(str.substring(7,11))

//3. split --> se utiliza para dividir la cadena en partes.
//toList convierte un array resultante del split en una lista de Scala
println(str.split(" ").toList)
```

Strings

```
//4. startsWith --> verifica si el string comienza por el valor indicado. Devuelve un booleano
println(str.startsWith("Hola"))

//5.replace --> reemplaza carácter por otro
println(str.replace(" ","-"))

//6. toLowerCase --> convierte todos los caracteres de la cadena en minúsculas
println(str.toLowerCase())

//7. length --> devuelve la longitud
println(str.length)
```

Strings

```
//8. +: operador para añadir al inicio de la cadena. :+ operador para añadir al final de la cadena
val aNumberString="2"
val aNumber = aNumberString.toInt

println('a' +: aNumberString :+ 'z')

//9. imprime al revés la cadena
println(str.reverse)

//10. take (2) --> devuelve los dos primeros caracteres de la cadena
println(str.take(2))
```

Strings

```scala
//11. Scala-specific: String interpolators ---s-interpolators
//Esto permite insertar valores de variables directamente en el texto utilizando el símbolo $.
val name = "David"
val edad = 12
val presenta = s"Hola, mi nombre es $name y tengo $edad años"
val otropresenta =s"Hola, mi nombre es $name y cumpliré ${edad + 1} años"
println(otropresenta)
```

Strings

```
//12. F-interpolators
//El prefijo f permite especificar el formato para los valores interpolados dentro de la cadena.
//$nombre%s --> %s: Indica que el valor de la variable nombre debe tratarse como una cadena de texto (string)
//$velocidad%2.2f -->
//   %f: Indica que el valor de velocidad es un número en punto flotante (float).
//   2.2:
//      El primer 2 especifica que el número tendrá un ancho mínimo de 2 caracteres
//          (rellenará espacios si el número es más corto).
//      El segundo 2 especifica que tendrá 2 dígitos después del punto decimal.

val velocidad = 1.2f
val nombre = "Roberto"
println (f"$nombre%s puede comer $velocidad%2.2f bocadillos por minuto")
```

Strings

```
//13. raw-interpolator --> los caracteres especiales se imprimen literalmente, sin efectos especiales
println("Esto es una \n nuevalinea")
println(raw"Esto es una \n nuevalinea")
val linea = "Esto es una \n nuevalinea"
println(raw"$linea")
```

Ejercicio

```
/* Ejercicio: Calcula Descuento:
    Realiza una función que reciba los siguientes parámetros:
        - precio del producto
        - porcentaje de descuento
    La función debe calcular el precio final después de aplicar el descuento.
    Si el descuento es mayor o igual a 50, debe devolver una advertencia:
        "Precio final: [preciofinal]. ¡Descuento muy alto|"
    Si el descuento es menor a 50, simplemente devuelve:
        "Precio final: [preciofinal]"

Una vez implementada la función realiza las siguientes llamadas:
    - Calcula Descuento precio de 100
    - Calcula Descuento precio de 200 y descuento de 20
    - Calcula Descuento precio de 300 y descuento de 50
*/
```