10/10/2024

// Declaramos las variables antes

var sueldo : Double = 300.54  
var saldo : Double = 300.54

ingresar\_sueldo()  
  
 /\* Declaramos las variables antes y así podemos usarlo con *todo el proyecto*\*/  
 var sueldo : Double = 300.54  
 var saldo : Double = 300.54  
 val fecha = "01/10/2024"  
 var dia = fecha.subSequence(0,2).toString().*toInt*()  
 if(dia == 1) ingresar\_sueldo()  
  
}  
// Aqui es donde van nuestra funcion  
fun mostrar\_saldo() { // al estar en gris es porque no lo estamos usando  
 *println*("Tienes $saldo")  
}  
  
fun ingresar\_sueldo() {  
 saldo += sueldo  
 *println*("Se ha ingresado tu sueldo de $sueldo \n")  
 mostrar\_saldo() // Le llamamos para que nos muestre lo que hemos cobrado y el ingreso  
}

// Lo que sale por consola

Se ha ingresado tu sueldo de 300.54

Tienes 601.08

/\*16/10/2024\*/

// Calculo factorial  
println(calculo\_factorial(10))  
  
// Calculo cuadrados  
val numeros = *listOf*(1, 2, 3, 4, 5)  
val cuadradosNumeros = cuadrados(numeros)  
println(cuadradosNumeros)  
  
// Calculo del área de circunferencia y cilindro  
val radio = 50.2  
val altura = 14.5  
  
*println*("El area del circulo con radio $radio es: ${areaCirculo(radio)}")  
*println*("El volumen del cilindro con radio $radio y altura $altura es: ${volumenCilindro(radio, altura)}")

fun retirar\_dinero(cantidad : Int) {  
 if (verificar\_cantidad(cantidad)) {  
 saldo -= cantidad  
 *println*("Se ha retirado la cantidad de $cantidad")  
 mostrar\_saldo()  
 }  
}  
  
fun verificar\_cantidad(cantidad\_retirada : Int) : Boolean { // Para que me devuelva si es falso o verdadero le ponemos el Boolean  
 if (cantidad\_retirada > saldo) return false  
 else return true  
}  
  
/\*Ej1: Escribir una función que reciba un numero y se calcule su factorial, mediante la recursividad.\*/  
fun calculo\_factorial(num : Int) : Int {  
 if (num <= 1) {  
 return 1  
 }  
 return num \* (calculo\_factorial(num -1))  
}  
/\*Ej2: Escribir una función que reciba una muestra de números en una lista y devuelva otra lista con sus cuadrados.\*/  
fun cuadrados(listaNumeros: List<Int>): List<Int> {  
 return listaNumeros.*map* **{  
 it** \* **it  
 }**}  
/\*Ej3: Escribir una función que calcule el área de un círculo y otra que calcule el volumen de un cilindro usando la primera función\*/  
fun areaCirculo(radio: Double): Double {  
 return *PI* \* radio \* radio  
}  
  
fun volumenCilindro(radio: Double, altura: Double): Double {  
 val areaBase = areaCirculo(radio)  
 return areaBase \* altura  
}

/\*17/10/2024\*/  
val simpleArray = *arrayOf*(1, 2, 3) // Declaramos el array  
  
val nullArray: Array<Int?> = *arrayOfNulls*(3) // Le metemos el tamaño  
  
val initArray = Array<Int>(3) **{** 0 **}**initArray.*forEach* **{** *println*(**it**) **}** // Así recorremos todos los elementos del array. Cuando ponemos it es la variable de cada uno de nuestros elementos  
  
// Arrays multidimensional  
val twoArray = Array(2) **{**Array<Int>(2) **{** 2 **} }** // Nos mostrará por consola [[0,0], [0,0]]  
val threeArray = Array(3) **{**Array<Int>(3) **{** 3 **} }** // [[[0,0,0], [0,0,0], [0,0,0]], [[0,0,0], [0,0,0], [0,0,0]], [[0,0,0], [0,0,0], [0,0,0]]]  
  
/\*Ejercicio 1: Crea una Array multidemesional de rango 2, introduce valores aleatorios y recorrela mostrando sus valores\*/  
val arrayRangoDos = Array(2) **{**Array<Int>(2) **{**Random.nextInt(0,10)**} }***println*("Array de rango 2")  
for (fila in arrayRangoDos) {  
 *println*(fila.*joinToString*("\t")) // Mostrar los valores de cada fila separados por tabulaciones  
}  
  
/\*Ejercicio 2: Crea una Array multidemesional de rango 3, introduce valores aleatorios y recorrela mostrando sus valores\*/  
val arrayRangoTres = Array(3) **{**Array<Int>(3) **{**Random.nextInt(0,10)**} }***println*("Array de rango 3")  
for (fila in arrayRangoTres) {  
 *println*(fila.*joinToString*("\t")) // Mostrar los valores de cada fila separados por tabulaciones  
}

/\*23/10/2024\*/

/\*MAIN\*/  
// Crear variables  
val Sandra: Person = Person("Sandra", "A123456789")  
val anonimo: Person = Person()  
  
// Imprimir valores iniciales  
*println*(Sandra.alive) // true  
*println*(Sandra.getName()) // Sandra  
*println*(Sandra.getPasaporte()) // A123456789  
Sandra.die()  
*println*(Sandra.alive)  
  
*println*(anonimo.alive) // true  
*println*(anonimo.getName())  
*println*(anonimo.getPasaporte())  
  
// Usar los setters para cambiar los valores del anonimo  
anonimo.setName("Angel")  
anonimo.setPasaporte("B98754321")  
  
// Imprimir valores después de cambiar  
*println*(anonimo.alive)  
*println*(anonimo.getName())  
*println*(anonimo.getPasaporte())

/\*CLASE\*/

package com.example.poo  
  
class Person(private var nombre: String = "Anonimo", private var pasaporte: String? = null) {  
 var alive: Boolean = true  
  
 // Método para "morir"  
 fun die() {  
 alive = false  
 }  
  
 // Getters  
 fun getName(): String {  
 return this.nombre  
 }  
  
 fun getPasaporte(): String? {  
 return this.pasaporte  
 }  
  
 // Setters  
 fun setName(nombre: String) {  
 this.nombre = nombre  
 }  
  
 fun setPasaporte(pasaporte: String?) {  
 this.pasaporte = pasaporte  
 }  
}

/\*07/11/2024\*/  
Main  
// Referencia al TextView y Button  
val miTexto = findViewById<TextView>(R.id.miTexto)  
val botonCambiarTexto = findViewById<Button>(R.id.*botonCambiarTexto*)  
  
// Evento de clic para cambiar el texto  
botonCambiarTexto.setOnClickListener **{** miTexto.*text* = "Holaaaa" // Cambia el texto del TextView  
**}**

XML

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/main"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Texto original"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.113" />  
  
 <!--Boton para cambiar el texto-->  
 <Button  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Cambiar texto"  
 app:layout\_constraintBottom\_toBottomOf="@+id/botonCambiarTexto"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 android:layout\_marginTop="20dp"/>  
  
 <Button  
 android:id="@+id/button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Button"  
 app:layout\_constraintBottom\_toBottomOf="@+id/botonCambiarTexto"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 android:layout\_marginTop="20dp"/>  
  
</androidx.constraintlayout.widget.ConstraintLayout>