**PRACTICA 3: KOTLIN**

**1. Dada la declaración siguiente:**

**val miVariable:String?=null**

**Indica si las afirmaciones que aparecen a continuación son ciertas o falsas:**

1. **La instrucción println(miVariable.length) dará como resultado null.**

Falso

1. **La instrucción println(miVariable?.length) dará como resultado 0.**

Falso

1. **La instrucción println(miVariable?.length ?: -1) dará como resultado -1.**

Verdadero

1. **La instrucción println(miVariable!!.length) fuerza una excepción de tipo NullPointerException.**

Verdadero

**2. ¿Cuáles de las afirmaciones siguientes son ciertas o falsas respecto a las expresiones Lambda en Kotlin?**

**a. Se expresan entre llaves { }.**

Verdadero

**b. Utilizan la palabra clave fun sin especificar el nombre.**

Falso

**c. No especifican el tipo de retorno, ya que este es inferido por el compilador.**

Verdadero

**d. Los argumentos se expresan entre paréntesis.**

Falso

**e. Una expresión Lambda puede asignarse a una variable y ejecutarla.**

Verdadero

**3. ¿Qué afirmaciones son ciertas respecto a las definiciones siguientes?**

**open class B (var x:Int, var y:Int)**

**class A (x:Int, y:Int, var z:Int):B(x, y), C, D {...}**

**a. La clase A desciende de las clases A, B y C.**

Falso

**b. La clase A desciende de la clase B.**

Verdadero

**c. La clase A implementa las interfaces C y D.**

Verdadero

**d. La clase A define tres propiedades mutables.**

Falso

**e. La clase A define una propiedad mutable, pero tiene acceso a x e y a través de B.**

Verdadero

**4. Se pide crear un programa en Kotlin que implemente las funcionalidades siguientes:**

**a. Define una interfaz Superficie que contenga el método mostrarArea.**

**b. Define una clase abierta Figura que contendrá una propiedad de tipo String con el color de la figura, y un método mostrarColor que muestre el color de la figura.**

**c. Define una clase Rectangulo, que sea también una figura y que defina una base y una altura. Además, esta clase implementará la interfaz Superficie de manera que pueda mostrarse su área.**

**d. Crea una función principal que cree un cuadrado de base 3 y altura 2 de color azul, y que muestre su área y la propiedad de color mediante los métodos de las clases.**

**Código:**

interface Superficie {

fun mostrarArea(): Double

}

open class Figura(var color: String) {

fun mostrarColor() = println(color)

}

class Rectangulo(color: String, var base: Double, var altura: Double): Figura(color), Superficie {

override fun mostrarArea(): Double {

return base\*altura

}

}

fun main() {

val cuadrado = Rectangulo("Verde", 3.0, 2.0)

var resultado = cuadrado.mostrarArea()

println(resultado)

cuadrado.mostrarColor()

}

**5. Siguiendo con el ejercicio anterior, crea ahora las clases y funcionalidades correspondientes para:**

**a. Crear un rectángulo de base 3 y altura 5, de color azul.**

class Rectangulo(color: String, var base: Double, var altura: Double): Figura(color), Superficie {

override fun mostrarArea(): Double {

return base\*altura

}

}

fun main() {

val rectangulo = Rectangulo("Azul", 3.0, 5.0)

rectangulo.mostrarColor()

println("El rectangulo tiene un area de ${rectangulo.mostrarArea()}")

}

**b. Crear un cuadrado de lado 4, de color verde.**

class Cuadrado(color: String, var lado: Double): Figura(color), Superficie {

override fun mostrarArea(): Double {

return lado\*lado

}

}

fun main() {

val cuadrado = Cuadrado("Verde", 4.0)

cuadrado.mostrarColor();

println("El cuadrado tiene un area de ${cuadrado.mostrarArea()}")

**c. Crear un triángulo base 2 y altura 5, de color naranja.**

lass Triangulo(color: String, var base: Double, var altura: Double): Figura(color), Superficie {

override fun mostrarArea(): Double {

return (base\*altura)/2

}

}

fun main() {

val triangulo = Triangulo("Naranja", 2.0, 5.0)

triangulo.mostrarColor()

println("El triangulo tiene un area de ${triangulo.mostrarArea()}")

**d. Crear un círculo de radio 7, de color rojo.**

class Circulo(color: String, var radio: Double): Figura(color), Superficie {

override fun mostrarArea(): Double {

return 3.14\*(radio\*radio)

}

}

fun main() {

val circulo = Circulo("Rojo", 7.0)

circulo.mostrarColor();

println("El circulo tiene un area de ${circulo.mostrarArea()}"

**e. Crear una elipse de radios 5 y 6, de color amarillo.**

class Elipse(color: String, var radio1: Double, var radio2: Double): Figura(color), Superficie {

override fun mostrarArea(): Double {

return 3.14\*radio1\*radio2

}

}

fun main() {

val elipse = Elipse("Amarillo", 5.0, 6.0)

elipse.mostrarColor();

println("La elipse tiene un area de ${elipse.mostrarArea()}")

1. **Y que se muestre el área de cada objeto y su color, del mismo modo que hemos hecho en el caso anterior.**

Hecho