



Actividad. Herencia de Clases Abstractas e Interfaces

☰ Tags	Done
➤ Materias	📱 <u>PROGRAMACIÓN MÓVIL</u>
📅 Fecha de entrega	@March 20, 2024
📄 Boleta	2022602116
👤 Nombre del estudiante	Martinez Martinez Adrian

Instrucciones

A partir del proyecto del sistema de reserva de Viajes (Class Travel) realiza los siguientes puntos:

- Crear la clase International para viajes internacionales, contemplar que ahora el usuario proporciona el nombre del País y la Ciudad.

```
package org.example.Class
```

```
class International(override val country: String, override val city: String) {
```

```
    // [4] Redefinido para obtener el precio de un viaje con impuestos
    override fun getPrice(numDays: Int): Int {
        val basePrice = getBasePrice(city)
        val taxPercentage = getTaxPercentage(country)
        if (basePrice == 0 || taxPercentage == 0) {
            println("Lo sentimos, la ciudad o el país seleccionados no son válidos")
            return 0
        }
    }
}
```

```

        val totalPrice = basePrice * numDays * (1 + taxPercentage)
        return totalPrice.toInt()
    }

    // [2] Eliminado el quotePrice

    private val cityCountryMap = mapOf(
        "Munich" to "Alemania",
        "Berlín" to "Alemania",
        "Francfort" to "Alemania",
        "Santiago" to "Chile",
        "Valparaíso" to "Chile",
        "Vancouver" to "Canadá",
        "Ottawa" to "Canadá",
        "Montreal" to "Canadá"
    )

    // [3] Precios
    private fun getBasePrice(city: String): Int {
        return when (city) {
            "Munich" -> 980
            "Berlín" -> 820
            "Francfort" -> 850
            "Santiago" -> 643
            "Valparaíso" -> 721
            "Vancouver" -> 620
            "Ottawa" -> 680
            "Montreal" -> 600
            else -> 0
        }
    }

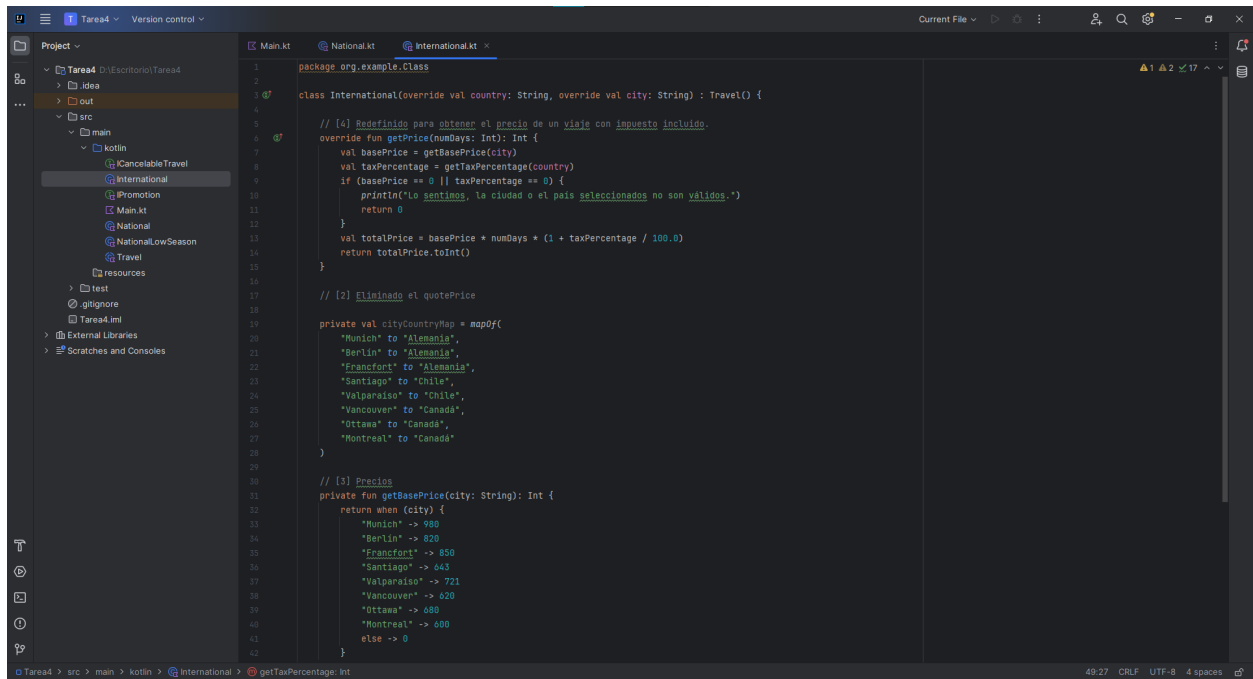
    private fun getTaxPercentage(country: String): Int {
        return when (country) {
            "Alemania" -> 16
            "Chile" -> 5

```

```

        "Canadá" -> 10
        else -> 0
    }
}
}

```



1. Existe un miembro en la Clase Travel que a pesar de ser abstracta, podría ser idéntico tanto en National como en International. En caso de afirmación, agrega el cuerpo en la clase abstracta y elimínala de sus hijos.

```
package org.example.Class
```

```

abstract class Travel {
    abstract val country: String
    abstract val city: String
    protected val serviceType = "Viaje"
    protected var reserved = false
    protected var paidAmount = 0

    fun reserve(numDays: Int){

```

```

        if(reserved){
            println("""¡Ya reservaste tu viaje!
                    País: $country
                    Ciudad: $city
                    Precio: $paidAmount""".trimMargin())

            return
        }
        val amount = getPrice(numDays)
        if(amount==0){
            return
        }
        reserved = true
        paidAmount = amount
        println("""¡Viaje reservado exitosamente!
                País: $country
                Ciudad: $city
                Precio: $paidAmount""".trimMargin())
    }

    protected abstract fun getPrice(numDays: Int): Int

    // [2] Implementacion de quotePrice en la clase abstracta
    fun quotePrice(numDays: Int) {
        val price = getPrice(numDays)
        if(price==0){
            println("Lo sentimos, no hacemos viajes a esta ciudad")
        } else{
            println("Tu viaje a $city, $country cuesta \$$price")
        }
    }
}

```

1. Debemos establecer los impuestos por país, y las ciudades a donde viajar:
Alemania tiene el 16% del precio total como impuesto y las ciudades disponibles son:

Munich, \$980 por día

Berlín, \$820 por día

Francfort, \$850 por día

Chile cobra únicamente el 5% como impuesto y sus ciudades son:

Santiago, \$643 por día

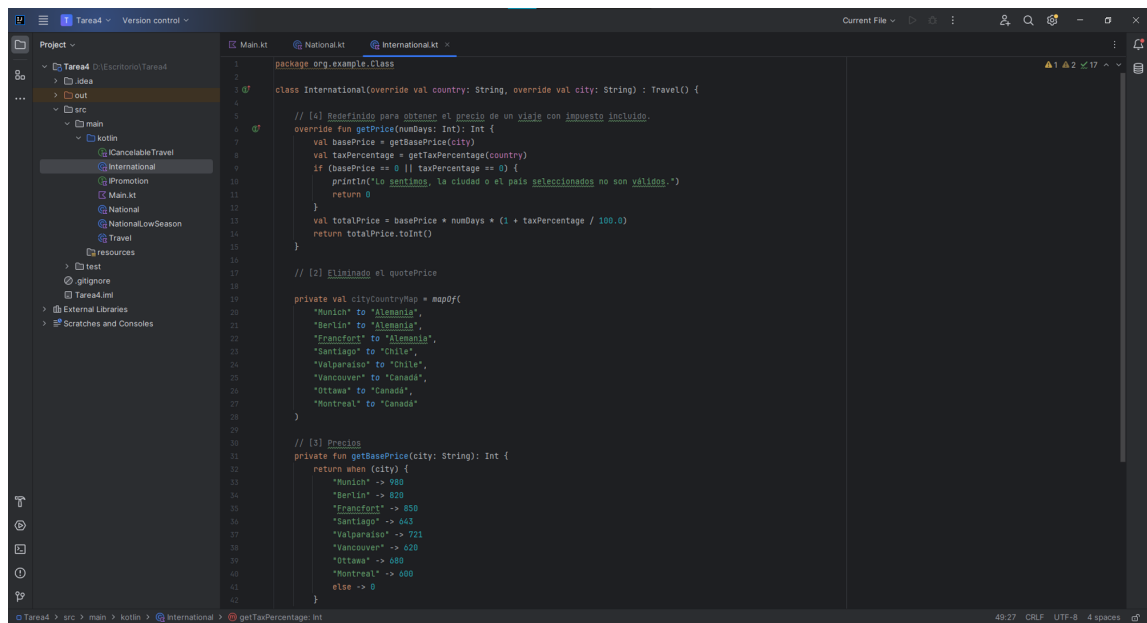
Valparaíso, \$721 por día

Canadá cobra el 10% de impuesto y las ciudades a visitar son:

Vancouver, \$620 por día

Ottawa, \$680 por día

Montreal, \$600 por día



- Redefinir la función getPrice() para que se obtenga el precio de un viaje con impuesto incluido.

```
package org.example.Class
```

```
abstract class Travel {
    abstract val country: String
    abstract val city: String
}
```

```

protected val serviceType = "Viaje"
protected var reserved = false
protected var paidAmount = 0

fun reserve(numDays: Int){
    if(reserved){
        println("""¡Ya reservaste tu viaje!
                País: $country
                Ciudad: $city
                Precio: $paidAmount""".trimMargin())

        return
    }
    val amount = getPrice(numDays)
    if(amount==0){
        return
    }
    reserved = true
    paidAmount = amount
    println("""¡Viaje reservado exitosamente!
            País: $country
            Ciudad: $city
            Precio: $paidAmount""".trimMargin())
}

protected abstract fun getPrice(numDays: Int): Int

// [2] Implementacion de quotePrice en la clase abstracta
fun quotePrice(numDays: Int) {
    val price = getPrice(numDays)
    if(price==0){
        println("Lo sentimos, no hacemos viajes a esta ciudad")
    } else{
        println("Tu viaje a $city, $country cuesta \$$price")
    }
}

```

```

    }
}

```

The screenshot shows the IntelliJ IDEA interface with the project 'Tarea4' open. The file explorer on the left shows the project structure, including the 'Travel' interface. The main editor displays the 'National.kt' file, which implements the 'Travel' interface. The code defines an abstract class 'Travel' with properties 'country', 'city', 'serviceType', 'reserved', and 'paidAmount'. It includes methods 'reserve(numDays: Int)' and 'quotePrice(numDays: Int)'. The 'reserve' method prints the reservation details and updates the 'reserved' and 'paidAmount' properties. The 'quotePrice' method prints the price for the reservation.

```

1 package org.example.class
2
3 abstract class Travel {
4     abstract val country: String
5     abstract val city: String
6     protected val serviceType = "viaje"
7     protected var reserved = false
8     protected var paidAmount = 0
9
10    fun reserve(numDays: Int){
11        if(reserved){
12            println("Ya reservaste tu viaje!")
13            Pais: $country
14            Ciudad: $city
15            Precio: $paidAmount***.trimMargin()
16        }
17        return
18    }
19    val amount = getPrice(numDays)
20    if(amount==0){
21        return
22    }
23    reserved = true
24    paidAmount = amount
25    println("Viaje reservado exitosamente!")
26    Pais: $country
27    Ciudad: $city
28    Precio: $paidAmount***.trimMargin()
29 }
30
31 protected abstract fun getPrice(numDays: Int): Int
32
33 // [2] Implementacion de quotePrice en la clase abstracta
34 fun quotePrice(numDays: Int) {
35     val price = getPrice(numDays)
36     if(price==0){
37         println("Lo sentimos, no hacemos viajes a esta ciudad")
38     } else{
39         println("Tu viaje a $city, $country cuesta \$$price")
40     }
41 }

```

- Crear una interfaz que permita cancelar viajes. Implementarlo en la clase NationalLowSeason.

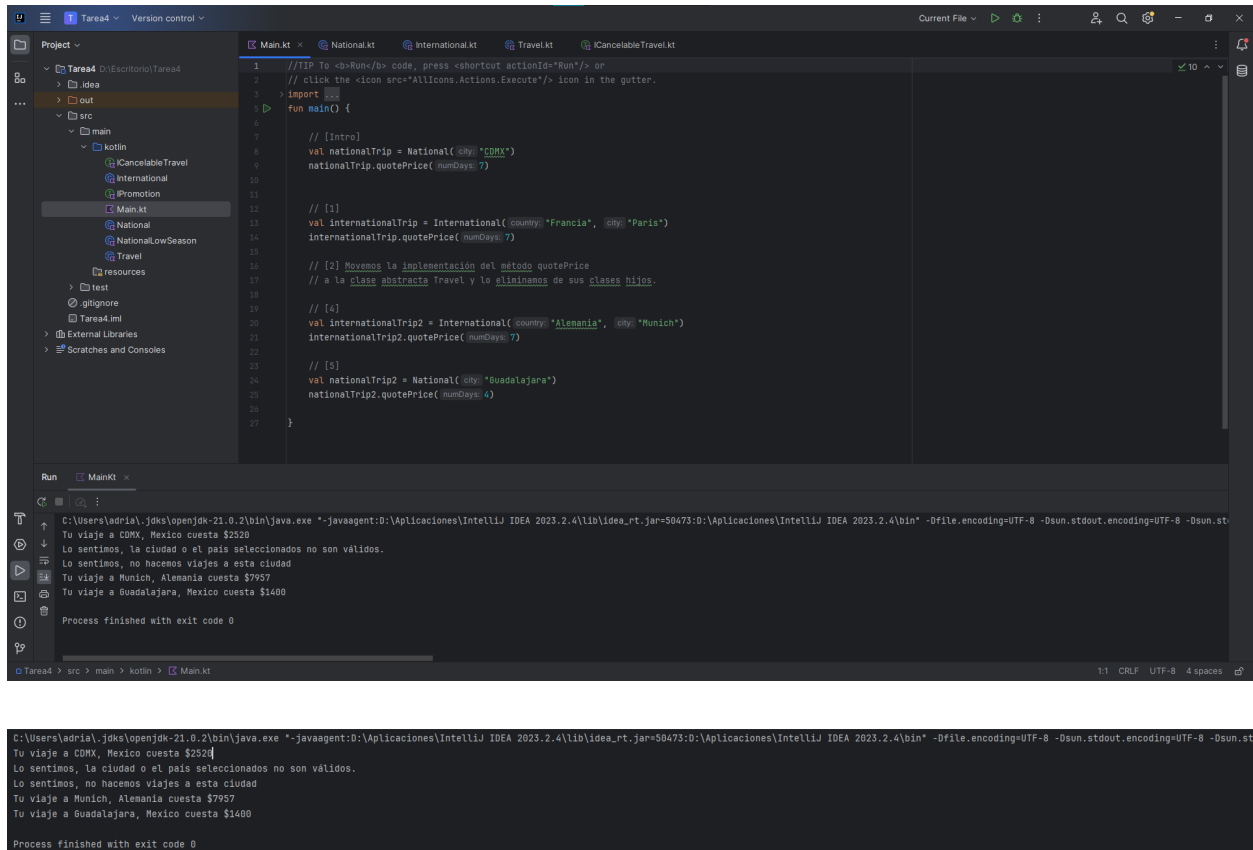
The screenshot shows the IntelliJ IDEA interface with the project 'Tarea4' open. The file explorer on the left shows the project structure, including the 'CancelableTravel' interface. The main editor displays the 'CancelableTravel.kt' file, which defines the 'CancelableTravel' interface with a 'cancelTravel()' method.

```

1 package org.example.interfaces
2
3 interface CancelableTravel {
4     fun cancelTravel()
5 }

```

Salidas finales del main :



The screenshot displays an IDE window with a project named 'Tarea4'. The left sidebar shows the project structure, including a 'main' directory with a 'kotlin' subdirectory containing 'Main.kt'. The main editor shows the code in 'Main.kt', which includes imports for 'National.kt', 'International.kt', 'Travel.kt', and 'ICancelableTravel.kt'. The code defines a 'main' function that creates instances of 'National', 'International', and 'NationalTrip2' and calls their 'quotePrice' methods. The output window at the bottom shows the execution results, including the cost of travel for different destinations and the number of days.

```
1 //TIP To <b>Run</b> code, press <shortcut actionId="Run"/> or
2 // click the <icon src="AllIcons.Actions.Execute"/> icon in the gutter.
3 import kotlin.*
4
5 fun main() {
6
7     // [Intro]
8     val nationalTrip = National(country = "CDMX")
9     nationalTrip.quotePrice(numDays = 7)
10
11
12     // [1]
13     val internationalTrip = International(country = "Francis", city = "Paris")
14     internationalTrip.quotePrice(numDays = 7)
15
16     // [2] Movemos la implementación del método quotePrice
17     // a la clase abstracta Travel y lo eliminamos de sus clases hijas.
18
19     // [4]
20     val internationalTrip2 = International(country = "Alemania", city = "Munich")
21     internationalTrip2.quotePrice(numDays = 7)
22
23     // [5]
24     val nationalTrip2 = National(city = "Guadalajara")
25     nationalTrip2.quotePrice(numDays = 4)
26 }
27
```

Run C:\Users\adria\jdk\openjdk-21.0.2\bin\java.exe --javaagent:D:\Aplicaciones\IntelliJ IDEA 2023.2.4\lib\idea_rt.jar=50473:D:\Aplicaciones\IntelliJ IDEA 2023.2.4\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.st

Tu viaje a CDMX, Mexico cuesta \$2520
Lo sentimos, la ciudad o el país seleccionados no son válidos.
Lo sentimos, no hacemos viajes a esta ciudad
Tu viaje a Munich, Alemania cuesta \$7957
Tu viaje a Guadalajara, Mexico cuesta \$1400
Process finished with exit code 0

C:\Users\adria\jdk\openjdk-21.0.2\bin\java.exe --javaagent:D:\Aplicaciones\IntelliJ IDEA 2023.2.4\lib\idea_rt.jar=50473:D:\Aplicaciones\IntelliJ IDEA 2023.2.4\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.st

Tu viaje a CDMX, Mexico cuesta \$2520
Lo sentimos, la ciudad o el país seleccionados no son válidos.
Lo sentimos, no hacemos viajes a esta ciudad
Tu viaje a Munich, Alemania cuesta \$7957
Tu viaje a Guadalajara, Mexico cuesta \$1400
Process finished with exit code 0