Filipe Lutz Mariano 25956

**Mobile Apps 2**

**Assignment Ten**

**Submission Date 03/01/2025**

Percent 10%

**Include all workings and references**

***Section A.***

1. What is the difference between Internationalization and Localization? (2 Marks)

**Internationalization:**
The process of designing and preparing your application so it can support multiple languages, regions, and cultures. It focuses on enabling the software to be adaptable without hardcoding cultural or regional specifics.
Example: Externalizing text strings into resource files.

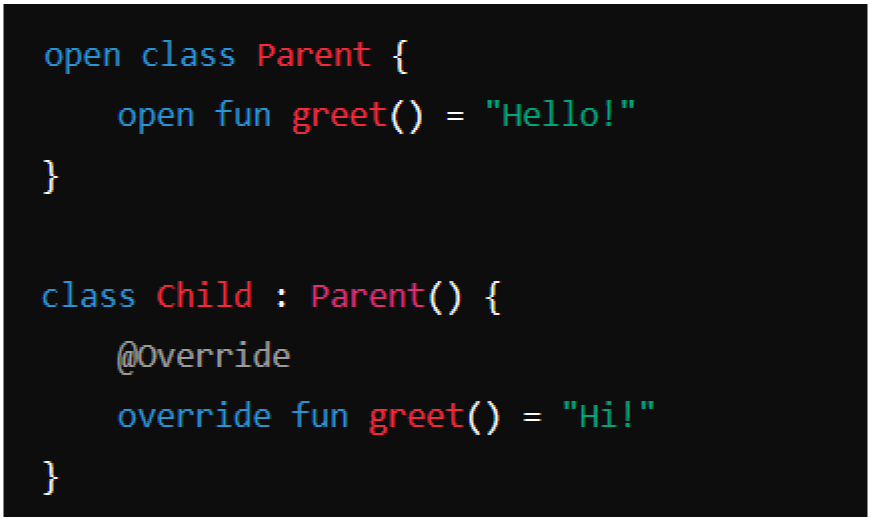
**Localization:**
The process of adapting an application for a specific language, region, or culture by translating text, formatting dates, numbers, or currencies, and applying other regional customizations.
Example: Translating text to French or adapting a date format from MM/DD/YYYY to DD/MM/YYYY.

**Key Difference:**

- Internationalization is about **preparing** the software for adaptation.
- Localization is about **implementing** the adaptation for a specific locale.

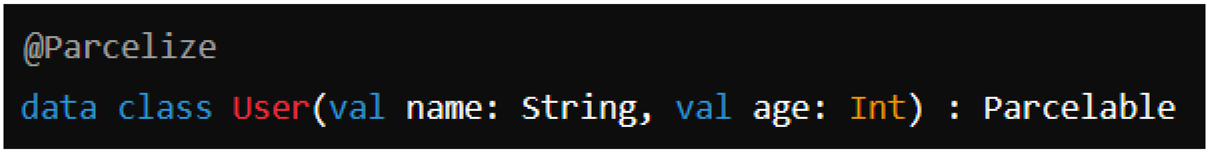
2. Besides unit tests, give 3 examples of the use of annotations in Kotlin. (3 Marks)

1. **@Override**
   Used to indicate that a method overrides a method in a superclass. This helps the compiler ensure that the method signature matches the overridden method.
   Example:



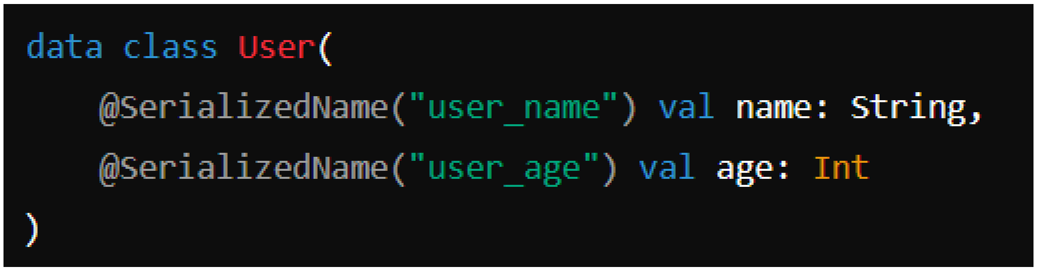
```
open class Parent {
    open fun greet() = "Hello!"
}

class Child : Parent() {
    @Override
    override fun greet() = "Hi!"
}
```

2. **@Parcelize**:
   Used in conjunction with the kotlin-parcelize plugin to automatically generate the Parcelable implementation for a data class.
   Example:



```
@Parcelize
data class User(val name: String, val age: Int) : Parcelable
```

3. **@SerializedName**:
   Used in JSON serialization libraries like Gson to map JSON keys to Kotlin properties.
   Example:



```
data class User(
    @SerializedName("user_name") val name: String,
    @SerializedName("user_age") val age: Int
)
```

***Section B.***

**Use simple code in both answers**

1. Using the sample code provided, perform the following:
   a. Complete the function **MethodB** in Class ***TheThing*** so it passes the related Unit test
   b. Write a test case that will test if the function **multiplyXY** to the following
      i. The multiplication is carried out
      ii. It returns a positive number

(3 *marks*)

2. Write code that will ensure that a timestamp is printed for every test case

(2 *marks*)

**The code is also on GitHub Repository: UnitTestExample**

## References:

1. Internationalization and Localization
2. Improve code inspection with annotations
3. Annotations
4. Kotlin annotations
5. Test code using JUnit in JVM – tutorial
6. Unit Testing in Android using JUnit
7. TimeStamp