CMPS 312 Mobile Application Development
Lab 3-Kotlin OOP and Lambdas

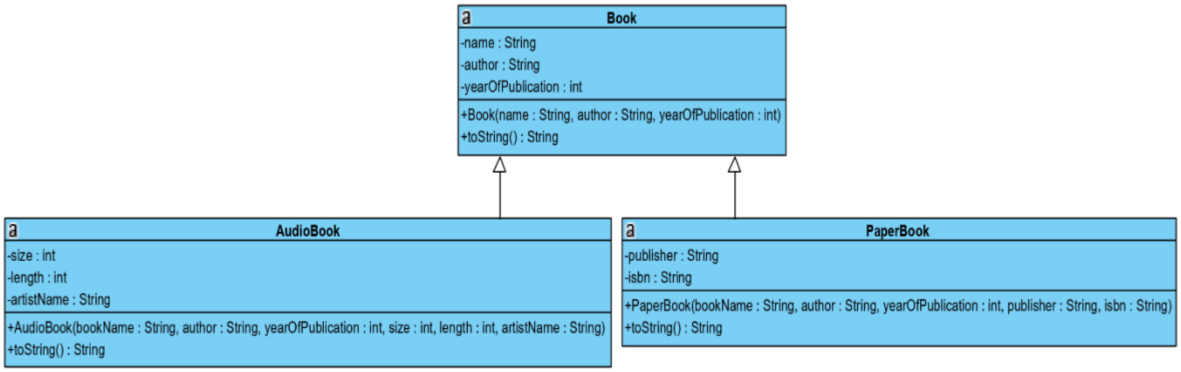
Objective

1. Practice Object Oriented Programming (OOP) using Kotlin
2. Read and parse JSON data
3. Practice processing collections using lambdas

PART A – OOP

Exercise 1

1. Create an application named **`Books`** with no Activity.
2. Create a package called **`model`**.
3. Implement the following class hierarchy inside the model package.



a
Book
-name : String
-author : String
-yearOfPublication : int
+Book(name : String, author : String, yearOfPublication : int)
+toString() : String
a
AudioBook
-size : int
-length : int
-artistName : String
+AudioBook(bookName : String, author : String, yearOfPublication : int, size : int, length : int, artistName : String)
+toString() : String
a
PaperBook
-publisher : String
-isbn : String
+PaperBook(bookName : String, author : String, yearOfPublication : int, publisher : String, isbn : String)
+toString() : String

- The toString() of Book should return **Name, Author, Year of Publication**.
- The toString() of PaperBook should return **Name**, **Author, Year of Publication, Publisher, ISBN**.
- The toString() of AudioBook should return **Name**, **Author, Year of Publication, Size, Length, ArtistName**

The data returned by the toString should be labeled (e.g., ***Name: Ali Baba and the Forty Thieves, Author: Hanna Diyab***).

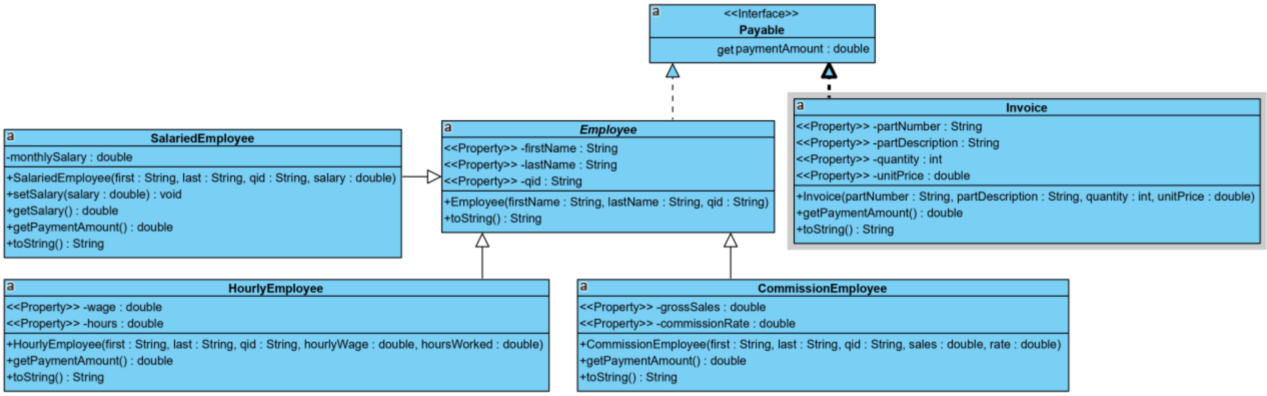
4. Create a main function to test your implementation.
5. In the main function create a List having 2 audio books and 2 paper books.
6. Display the details of each book using the list's forEach method.

Sample Output

```
Book Name : C++
Author Name : John
Year Of Publication : 1/2/2019
Publisher : Oriely
Isbn : 100-11-11


Book Name : Java
Author Name : Mark
Year Of Publication : 1/2/2019
Publisher : NewTimes
Isbn : 100-11-12


Book Name : Android
Author Name : Baaji
Year Of Publication : 1/2/2019
Publisher : Sanford
Isbn : 100-11-13


Book Name : How to get Rich
Author Name : Ali
Year Of Publication : 1/2/2019
Size : 100
Length : 25
Artist Name : Black Panter
```

EXERCISE 2

1. Create an application named “**QU Payroll**”
2. Create a package named **model**
3. Implement the following class hierarchy inside the **model** package

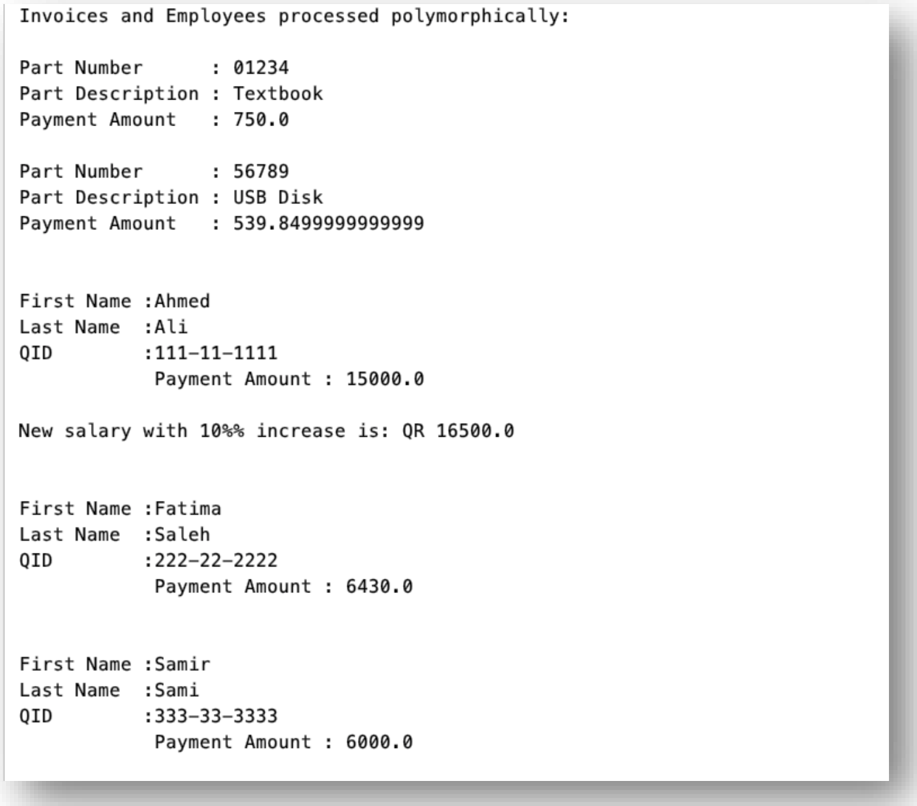


a
<<Interface>>
Payable
get paymentAmount : double
a
SalariedEmployee
-monthlySalary : double
+SalariedEmployee(first : String, last : String, qid : String, salary : double)
+setSalary(salary : double) : void
+getSalary() : double
+getPaymentAmount() : double
+toString() : String
a
Employee
<<Property>> -firstName : String
<<Property>> -lastName : String
<<Property>> -qid : String
+Employee(firstName : String, lastName : String, qid : String)
+toString() : String
a
Invoice
<<Property>> -partNumber : String
<<Property>> -partDescription : String
<<Property>> -quantity : int
<<Property>> -unitPrice : double
+Invoice(partNumber : String, partDescription : String, quantity : int, unitPrice : double)
+getPaymentAmount() : double
+toString() : String
a
HourlyEmployee
<<Property>> -wage : double
<<Property>> -hours : double
+HourlyEmployee(first : String, last : String, qid : String, hourlyWage : double, hoursWorked : double)
+getPaymentAmount() : double
+toString() : String
a
CommissionEmployee
<<Property>> -grossSales : double
<<Property>> -commissionRate : double
+CommissionEmployee(first : String, last : String, qid : String, sales : double, rate : double)
+getPaymentAmount() : double
+toString() : String

- Note that the amount to pay for HourlyEmployee is *wage * hours*. For CommissionEmployee, it is *grossSales * commissionRate*. For Invoice, it is *quantity * unitPrice*.
- Make sure the **salary**, **rate** and **sales** are all non-negative numbers otherwise display a warning message. [hint: for data validation using init or set methods]

Test your implementation using the main method

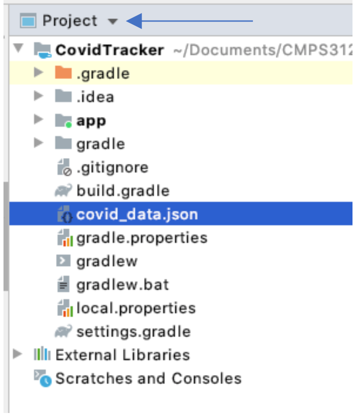
```
fun main() {

    // create payable array List
    val payables = arrayListOf<Payable>()

    // populate array with objects that implement Payable
    payables.add(Invoice("01234", "Textbook", 2, 375.00))
    payables.add(Invoice("56789", "USB Disk", 3, 179.95))
    payables.add(SalariedEmployee("Ahmed", "Ali", "111-11-1111", 15000.00))
    payables.add(HourlyEmployee("Fatima", "Saleh", "222-22-2222", 160.75, 40.0))
    payables.add(CommissionEmployee("Samir", "Sami", "333-33-3333", 100000.0, .06))

    println("Invoices and Employees processed polymorphically:\n");

    // generically process each element in array payableObjects using foreach
    payables.forEach { payable ->
        // output currentPayable and its appropriate payment amount
        println("$payable\n")

        //If SalariedEmployee then increase the salary by 10%
        if (payable is SalariedEmployee) {
            val oldBaseSalary = payable.monthlySalary;
            payable.monthlySalary = oldBaseSalary * 1.1;
            println("New salary with 10%% increase is: QR ${payable.getPaymentAmount()}\n");

        }
    }
}
```



```
Invoices and Employees processed polymorphically:

Part Number      : 01234
Part Description : Textbook
Payment Amount   : 750.0

Part Number      : 56789
Part Description : USB Disk
Payment Amount   : 539.8499999999999


First Name :Ahmed
Last Name  :Ali
QID        :111-11-1111
            Payment Amount : 15000.0

New salary with 10%% increase is: QR 16500.0


First Name :Fatima
Last Name  :Saleh
QID        :222-22-2222
            Payment Amount : 6430.0


First Name :Samir
Last Name  :Sami
QID        :333-33-3333
            Payment Amount : 6000.0
```

PART B - LAMBDAS

1. Create an application with and name it "**CovidTracker**"
2. Copy the **covid_data.json** from your lab repo under **Lab 3 folde**r and paste it in the root directory of your project.

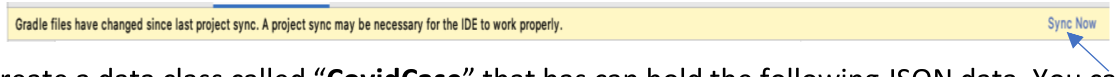


3. Once you copy the file and open it in Android Studio you might get a warning message showing
"File size exceeds configured limit (2.5MB). Code insight features not available…"
To fix the above problem do the following

a. Go to Help > Edit Custom Properties
b. Add: *idea.max.intellisense.filesize=999999*
c. Restart the IDE.

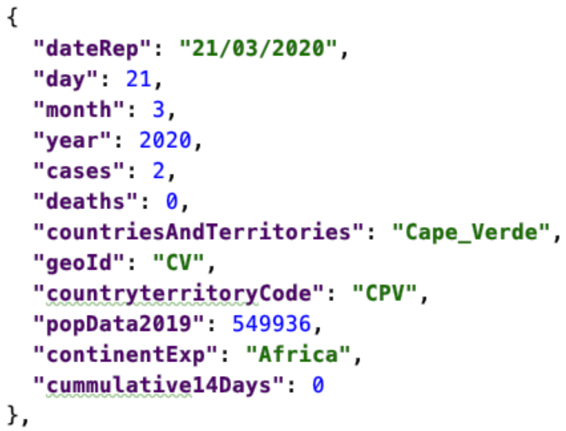
4. Open **build.gradle** and add the following dependency and then press on the Sync Now at the corner of the screen.

```
implementation group: 'com.google.code.gson', name:'gson', version: '2.8.5'
```



Gradle files have changed since last project sync. A project sync may be necessary for the IDE to work properly.

5. Create a data class called "**CovidCase**" that has can hold the following JSON data. You can drive the properties from the below JSON object.



```
{
  "dateRep": "21/03/2020",
  "day": 21,
  "month": 3,
  "year": 2020,
  "cases": 2,
  "deaths": 0,
  "countriesAndTerritories": "Cape_Verde",
  "geoId": "CV",
  "countryterritoryCode": "CPV",
  "popData2019": 549936,
  "continentExp": "Africa",
  "cummulative14Days": 0
},
```

6. Create a new Kotlin file named **CovidAnalysis** and implement and test the following functions that return:

- The **total death** for a **given country**.
- The **total death** for a given **country on a specific month**
- The **total number of cases** in each **continent**
- The **top three countries** with the **highest number of COVID cases.**
- The **top three countries** with the **lowest number of COVID cases.**
- The country with the **lowest death** in **specific continent**.
- All countries and **their total COVID deaths.**