**JavaBasics Assignment 701**

**Objective:**

* Understand and practise with Classes, Objects, Inheritance, Encapsulation.
* Understand and practise with Polymorphism, Abstraction.
* Understanding the main difference between method overloading and overriding, between abstract class and interface in java

**Business needs:**

* Create a Java Console application bases on Java Classes/Objects, OOP, Exception Handing, IO, Java Collection to manage Human Resource (HR Management System). This assignment will cover all part of Java Basic. Class Diagram as below here:

**Working requirements:**

* Working environment: Eclipse IDE.
* Delivery: Source code, deployment and testing, reviewing evident packaged in a compress archive.

**Product architecture:**

* N/A

**Technologies:**

The product implements one or more technology:

* Control of Flows
* OOP
* Exception Handing
* Java Collection

**Stored Data:**

* Data stored in collection classes.

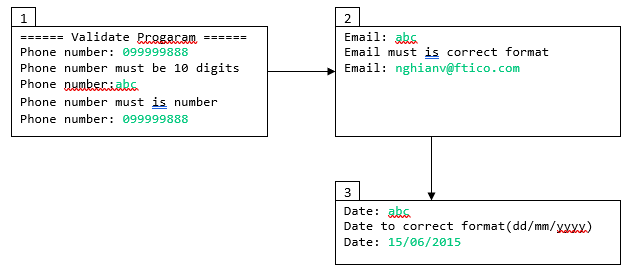
**Project Descriptions:**

**1. Exercise 1**

Thiết kế một chương trình cho phép nhập và kiểm tra định dạng:

* Số điện thoại
  + Kiểm tra số điện thoại có phải số không?
  + Kiểm tra số điện thoại phải có 10 chữ số?
* Email
  + Kiểm tra email phải đúng định dạng?
* Ngày tháng
  + Kiểm tra ngày tháng phải đúng định dạng dd/MM/yyyy?

**Yêu cầu giao diện:**

****

**Yêu cầu chức năng:**

**Điều kiện tiền đề:**

Người dùng chạy ứng dụng, chương trình cho phép nhập phone, email, date.

**Chức năng 1:** Kiểm tra Phone

* Chương trình yêu cầu nhập vào Phone
* Tên hàm trong code: **public** String checkPhone(String phone)
  + Đầu vào:
    - phone: Số điện thoại.
  + Giá trị trả về: thông báo lỗi.
* Nếu phone là 1 chữ cái thì hiển thị thông báo " Phone number must is number " và yêu cầu nhập lại
* Nếu phone là 1 số nhỏ hơn 1 chữ số thì hiển thị thông báo " Phone number must be 10 digits "và cho phép nhập lại

**Chức năng 2:** Kiểm tra email.

* Chương trình cho phép nhập email.
* Tên hàm trong code:**public** String checkEmail(String email)
  + Đầu vào:
    - email: Email
  + Giá trị trả về: thông báo lỗi.
* Nếu email nhập không đúng định dạng thì hiển thị " Email must is correct format " và cho nhập lại

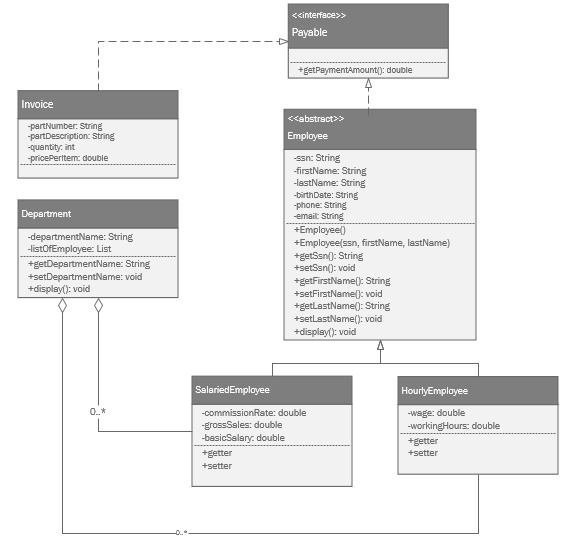
**Hướng dẫn cho học viên:**

**Gợi ý:**

* Sử dụng regex để kiêm tra định dạng của Phone và Email
* Sử dụng SimpleDateFormat để kiểm tra định dạng của Date

**2. Exercise 2**

For the class hierarchy is as follows, the trainee let's create the java classes install this class diagram to be able to relationship between it.



* **Employee** is an abstract superclass and has **six fields**: *ssn*, *firstName*, *lastName, birthDate, phone, email*;
* A **Salaried Employee** is paid annually. Salaried employees are usually supervisory, managerial, or professional employees who work on an annual basis and are not paid an hourly rate. **SalariedEmployee** is a concrete class that is a subclass of Employee and adds **three** **fields**: *commisstionRate, grossSales, basicSalary;*
* **HourlyEmployee**: Unlike a salaried employee who is paid a flat salary regardless of how many hours worked during a work month, an hourly employee is paid an hourly wage for each hour worked. This is a concrete class that is a subclass of Employee and adds **two** **fields**: *rate*, *workingHours*;
* **Department**: Each department will have **a list of employee** (contains: *salaried* and *hourly*);

*Noting that:*each abstract class and concrete class has a constructor and methods for getting and setting its fields (getters and setters) and a toString method.

Program requirements must validate the properties:

* BirthDate : correct date format (dd/MM/yyyy);
* Phone: minimum 7 positive integers;
* Email: correct email format.

**Functional/User Interface Requirement:**

The program has a screen console for UI

**The DepartmentManage main method:**

The main method in DepartmentManage class uses the Employee class and its subclasses, the main screen allows selecting the functions for:

1. **Input data from the keyboard**: create an employee list of all types as mentioned above and belong to several departments.
2. **Display employees**: displays information about each object polymorphically. The objects are stored in an Employee array.
3. **Classify employees**: the last for loop illustrates how to find out the specific class for each object.
4. **Employee Search**:

* Search for employees by enter the name of a specific department, display the list of employees by type to the screen;
* Enter the employee's name, display detailed information about the employee;

1. **Report**: display the list of departments and the number of employees for each.

**--THE END--**