# Lab 02 Inherit the Wind

## Learning Goals:

* Practice using inheritance to create sub-classes
* Create UML Class Diagrams to show the Inheritance Relationship

You will need the Person class from the previous lab!

## Functional requirements:

* Create each of the classes below in a single IntelliJ IDEA project. Name the Project **Inheritance.**
* Use GitHub to pull your Person.java file from your previous Lab.
* Create JUnit test suite for each of your classes. (You should already have one for Person.java from the last lab.)
* Note that there is a single java main program you will write to use all of the classes.

Here is the Person class from the previous lab:

### Person:

Project: Person  
Files: Person.java

Fields:

String firstName  
String lastName  
String ID // should never change  
String title // a prefix: Mr. Mrs. Ms, Prof. Dr. Hon. Etc.  
int YOB // Year of birth

Additional methods:

public String fullName() // returns firstName, space, lastName  
public String formalName() // returns title, space, fullName  
public int getAge() // uses YOB to calculate age for the current year  
// **Add this new overloaded method of getAge to the Person class**  
public int getAge(int year) // uses YOB to calculate age for a specified year

Revise your UML Class diagram for Person from the previous lab to reflect the new method. (You will need it to create a UML Class Diagram below.

### 2. Worker Class (inherits from Person):

Files: Worker.java WorkerTest.java

Fields:

double hourlyPayRate  
Methods:  
double calculateWeeklyPay(double hoursWorked)  
String displayWeeklyPay(double hoursWorked)

The constructor for Worker should use super() to call the constructor for Person and then go on to set the rest of the fields.

calculateWeeklyPay should return the pay total. Hours under 40 are at the HourlyRate, hours above 40 are at time and a half (1.5)

displayWeeklyPay should indicate the number of hours of regular pay (40) and the total and the number of hours of overtime pay and the total as well as the total combined pay.

### 3. SalaryWorker (inherits from Worker)

Files: SalaryWorker.java SalaryWorkerTest.java

Fields:

double annualSalary  
  
Methods:

The constructor for SalaryWorker should use super() to call the constructor for Worker  
double calculateWeeklyPay(double hoursWorked)

Override calculateWeeklyPay and displayWeeklyPay calculateWeeklyPay returns the pay total. Note that the parameter hoursWorked is not used here but is retained for polymorphism. Take the annualSalary and divide by 52 to get the weekly pay**.**

displayWeeklyPay should indicate that the weekly pay is a fraction of the yearly salary.

### 4. Main Program:

Create a new java main class within your same IntelliJ IDEA project **InheritanceDemo**

1. Create 3 workers and 3 salaryWorker instances and add them to an ArrayList<Worker>. Pick reasonable hourly rates for the workers and reasonable salary rates for the salaryworkers. Of course all workers have names, ids, and YOBs.

2. Write a loop that simulates 3 weekly pay periods. Week 1 is a 40 hour week. For week 2 assume it is crunch time and everyone worked 50 hours. Week 3 is back to normal with 40 hours. Generate a display showing the weekly pay for each of the workers for each week in a reasonable tabular format.

Screen shots: record screen shots of your Test file output for each of your test program at the end of this file.

### 4. UML Class Diagram:

Create a single UML diagram that shows the Inheritance hierarchy for Person Worker and SalaryWorker  
  
Insert the diagram here along with your screen shots.  
A paper with numbers and a few dollar bills

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated

A diagram of a computer program

Description automatically generated with medium confidence

## Submission:

Submit the link to your GitHub repo for this Lab. (Check the link to insure that it works correctly.)  
Submit this document with your screen shots and UML as **LastName\_Firstname\_Lab02.docx** using your name.