CS 1400-03 Introduction to Programming and Problem Solving
Project #7
(Due: 11:59 PM, Monday, 5/3/2021)

(a) Design a class named `Employee`. The class should keep the following information in private fields:

- Employee name (a `String`)
- Employee number (a `String`) in the format XXX-L, where each X is a digit within the range 0-9 and the L is a letter within the range A-Z.
- Hire date (a `String`)

Write a private method using the wrapper class testing methods to validate employee number. If it is not valid, assign empty string to the corresponding instance field. Write a no-argument constructor that sets all fields to empty string, a 3-argument constructor, appropriate setter and getter methods for each instance field, and the `toString` method. The `toString` method should concatenate "INVALID EMPLOYEE NUMBER" if the employee number is invalid.

(b) Write a class named `ProductionWorker` that extends the `Employee` class. The `ProductionWorker` class should have private fields to hold the following information:

- Shift (an integer)
- Hourly pay rate (a double)

The workday is divided into two shifts: day and night. The shift field will be an integer value representing the shift that the employee works. The day shift is shift 1 and the night shift is shift 2. Write a no-argument constructor that sets the default shift to 1 and pay rate to 0.0, a 5-argument constructor, appropriate setter and getter methods for each instance field, and the `toString` method. The `toString` method should concatenate "Day" if shift is 1, "Night" if shift is 2, or "INVALID SHIFT NUMBER" if the shift number is neither 1 nor 2. Print hourly pay rate with $ in the front and two digits after decimal point.

(c) Write a driver program, called `WorkerTest.java`, to create three `ProductionWorker` objects and test all features and capabilities described above. Create the first `ProductionWorker` object and pass the initialization data to the 5-arg constructor. Now create another `ProductionWorker` object and use the no-argument constructor and appropriate setter methods. Finally, create a third `ProductionWorker` object with wrong employee number and invalid shift integer. The `toString` methods should generate the following output:

```
fcsang@garrison ~/cs1400/project $ java WorkerTest
Here's the first production worker.
Name: John Smith
Employee Number: 123-A
Hire Date: 11-15-2005
Shift: Day
Hourly Pay Rate: $23.50

Here's the second production worker.
Name: Joan Jones
Employee Number: 222-L
Hire Date: 12-12-2018
Shift: Night
```

```
Hourly Pay Rate: $18.50

Here's the third production worker.
Name: Tony Gaddis
Employee Number: INVALID EMPLOYEE NUMBER
Hire Date: 1-23-2006
Shift: INVALID SHIFT NUMBER
Hourly Pay Rate: $19.50
```

In your `cs1400/project` directory, create three programs named `Employee.java`, `ProductionWorker.java` and `WorkerTest.java`. Your Java programs must begin with the comments below and follow the naming and coding conventions posted on Blackboard.

```
// your name
// CS1400, section 03
// Project 7 - Employee Inheritance
// date
```

Generate a `script` file `pj7.txt` with appropriate time stamps and the following steps visible:

1. a `pwd` to show the current working directory
2. an `ls -l` to show in long format the files in your `cs1400/project` directory
3. display `Employee.java`, `ProductionWorker.java` and `WorkerTest.java`
4. compile `WorkerTest.java`
5. run `WorkerTest` to display the above output

Submit `pj7.txt` on Gradescope.