Python Programming Exam 3

Instructions:

Please read the instructions on Canvas, as well as the directions on this exam carefully. Context and Programming Instructions begins on page 1. Sample outputs begin on page 2. Contact the instructor if there are any questions.

Employee Salary (100 pts)

You are interested in working for the XYZ company, and, during the interview, the XYZ company representative asks you to write a program that calculates an employee's weekly salary in one of their departments. This department has three types of Engineering employees, who have the following titles and who are paid the hourly wages indicated below:

Engineering Intern - $20 per hour
Senior Engineer - $45 per hour
Lead Engineer - $70 per hour

Objective:

Your program should do the following tasks (in one file):

1. Create a **class** called Employee.
    a. In this class, define an **initializer method** that has two protected attributes: name and employee ID. The method definition should include parameters that will update the protected attributes.
        - You do not need to create any other methods for this class.

2. Create another **class** called Engineer that inherits the attributes from the Employee class.
    a. In this class, define an **initializer method** that has two additional hidden attributes: title, hours worked. The method definition should include code that accesses the parent class's attributes, as well as parameters that will update the two hidden attributes.
    b. This class will also contain one more hidden attribute: payrate. This attribute will be updated via a call to an **accessor method** that will assign a payrate based on the title entered. (This is to ensure that the user does not assign their own payrate!)
    c. Define the **accessor method**. The method should accept a parameter for the employee's title and return the corresponding hourly payrate to the hidden attribute from part b. Create a **dictionary** in this method that stores the three hourly payrates based on the title of the employee.

d. Define another **method** that calculates the employee's weekly pay. The method does not require any parameters (other than the object itself) and returns the calculated pay.
    - The regular pay is calculated as the number of hours worked times the payrate. If the number of hours worked is over 40, the company pays "time and a half" for every additional hour over 40.
e. Create a **str method** that displays the employee's name, employee ID, and weekly pay. Format the pay to two decimal places.

3. In the 'driver portion' of the program, perform the following tasks:
    a. create local variables to get an employee's name, their employee ID, their title, and the number of hours worked.
        - Include input validation to ensure that the number of hours worked is not negative or greater than 60 hours.
    b. Create an **object** of type Engineer
    c. Using the object, calculate and display the employee's weekly pay, using the objects methods (created in part 2d and 2e respectively).

Samples of the output are shown below. (Note, your program does not have to have the exact wording in the output but should demonstrate that the code performs the required tasks.)

Sample 1 (checks input validation and calculation for 'time and a half'):

```
Enter the employee's name: Casey Brooks
Enter the employee's ID number: XYZ000123
Enter the employee's title: Lead Engineer
Enter the number of hours the employee worked this week: 65
Invalid. Hours worked can't be negative or greater than 60.
Enter the number of hours the employee worked this week: 60
Payroll Data for employee XYZ000123:
Casey Brooks's pay this week is: $4,900.00
```

Sample 2 (checks another title and payrate):

```
Enter the employee's name: Jamie Paul
Enter the employee's ID number: XYZ0010789
Enter the employee's title: Engineering Intern
Enter the number of hours the employee worked this week: 30
Payroll Data for employee XYZ0010789:
Jamie Paul's pay this week is: $600.00
```