Junior Knights Function Homework

Objectives

1. To fill in functions given pre-conditions and post-conditions.
2. To test those functions separately.
3. To integrate the working functions into a full program

Problem A: Regular Hourly Pay (reghour.py)

Write a function that calculates an employee's regular pay and returns that value. If an employee works less than the number of hours in a regular work week, then her pay is just the number of hours worked times her pay rate. If she works a number of hours equal to or greater than the regular work week, her pay is the regular work week multiplied by her regular pay rate. (Note: She will get paid more than this overall, but that extra pay is called overtime pay, for the purposes of this problem.)

The input to the function is as follows:

```
payPerHour - dollars per hour (float)
regWorkWeek - the number of hours in a regular work week
hoursWorked - the number of hours worked in a single week
```

Here is the function prototype with pre-conditions and post-conditions specified:

```
# Pre-conditions: payPerHour is a positive floating pt
#  number representing the pay per hour. regWorkWeek
#  represents the number of hours in a regular work week
#  (int). hoursWorked is the number of hours the employee
#  worked (int).
# Post-condition: Returns the regular pay for the employee.
def regularPay(payPerHour, regWorkWeek, hoursWorked):
    # Put your code here.
```

Here is a testing function to test your code and what it should print out:

```
def testRegularPay():
    print(regularPay(8.25, 40, 30))
    print(regularPay(12.50, 30, 40))
    print(regularPay(9.99, 80, 80))
testRegularPay()
```

Here is the expected output:

```
247.5
375.0
799.2
```

Problem B: Overtime Pay (overtime.py)

If an hourly employee works longer than the regular work week, then she gets overtime pay. For the purposes of this problem, there is an overtime rate which is greater than 1. For each hour past the regular work week, instead of getting paid the regular hourly rate, an employee gets paid the regular hourly rate times the overtime rate, per hour. For example, if the regular work week was 50 hours/week, the employee worked 60 hours for one week, the employee had a regular hourly rate of $10/hour and the overtime rate was 2.5, then the employee would get (60 hrs - 50 hrs) x $10/hr x 2.5 = $250 total in overtime pay. If an employee works less than the regular work week, she gets $0 in overtime pay.

Write a function that takes in these four quantities and calculates overtime pay:

```
payPerHour - dollars per hour (float)
regWorkWeek - the number of hours in a regular work week
hoursWorked - the number of hours worked in a single week
overtimeRate - overtime hour multiplier factor
```

Here is the function prototype with pre-conditions and post-conditions specified:

```
# Pre-conditions: payPerHour is a positive floating pt
#  number representing the pay per hour. regWorkWeek
#  represents the number of hours in a regular work week
#  (int). hoursWorked is the number of hours the employee
#  worked (int). overtimeRate is the multiplicative rate
#  for overtime pay.
# Post-condition: Returns the overtime pay for the employee.
def   overtimePay(payPerHour,   regWorkWeek,   hoursWorked,
overtimeRate):
    # Put your code here.
```

Here is a testing function to test your code and what it should print out:

```
def testOvertimePay():
    print(overtimePay(8.25, 40, 30, 1.5))
    print(overtimePay(12.50, 30, 40, 2.0))
    print(overtimePay(9.99, 80, 100, 2.5))
testOvertimePay()
```

Here is the expected output:

```
0
250.0
499.5
```

Problem C: Payment for several weeks (payment.py)

For each week, the total amount of regular pay plus overtime pay represents an employee's gross earnings. (In real life, unfortunately, money is taken from your gross pay and what ends up in your paycheck is called "net pay.") Write a program that calls the functions from problems A and B and calculates the gross earnings for an employee over several weeks. Your program should ask the user to enter each of the following pieces of information once, at the beginning:

1) Hourly Pay Rate in dollars
2) Regular Work Week
3) Overtime Rate
4) Number of Weeks they are working

Then, for each week, your program should ask the user how many hours they worked. At the end of each week, print out the employee's regular pay, overtime pay, and total payment.

After reading in information for each of the weeks, print out the employee's total payment over the whole program.

Your program should be written in a function called main, and main should call the functions from program A and program B.

Input Specification

Note: It is guaranteed that whoever uses your program will adhere to these specifications. This means that you do NOT have to check for them!

Hourly Pay Rate will be a float in between 8.0 and 100.0.
The regular work week will be an integer in between 30 and 80.
The overtime rate will be a float in between 1.5 and 10.
The number of weeks will be in between 2 and 10.

For each week, the number of hours worked will be an integer in between 0 and 120.

Output Specification

Prompt the user to enter each of the four items designated above in the order specified in the input specification. (Come up with your own reasonable prompts.)

For each week, after asking the user to enter the number of hours they worked for that week, print out a statement with the following form:

```
Week k regular pay = $X, overtime pay = $Y, total pay = $Z.
```

where k is the week number starting with 1, X is the regular pay for week k, Y is the over time pay for week k and Z is the total payment. Each of X, Y and Z should be floating point values.

After the output for the last week, conclude with a single line with the following format:

```
Total pay for W weeks = $D.
```

Where W is the total number of weeks worked and D is the total pay.

Sample Run

```
How much (in dollars) are you paid per hour?
8.85
How many hours in the regular work week?
40
What is the overtime rate?
1.5
How many weeks are you working?
3
How many hours did you work in week 1?
20
Week 1 regular pay = $177.0, overtime pay = $0.0, total pay = $177.0.
How many hours did you work in week 2?
42
Week 2 regular pay = $354.0, overtime pay = $26.55, total pay = $380.55.
Total pay for 2 weeks = $557.55.
```

Note: Don't worry if some of the numbers print to lots of decimals and are close to what's printed above but not exact.