UNIVERSITY OF ALBERTA
AUGUSTANA CAMPUS

Augustana Computing Science 111

Assignment #9 (last one!!)

10 November 2015 (due 24 November)

Goals:
- Continue growing with computational thinking (object oriented programming)
- Learn more Python: class structure

Instructions:
Complete the following Python program. Print out your program, and print out a sample of testing it to show me that it works. Also submit your program on eClass.

<u>Document ALL of your code</u>! Make sure that you add file headers (with your name(s), id #, date, and a brief summary). Also document each of the main steps in your solutions. Ensure that all your variables are named descriptively. All functions should also have a docstring.

Furthermore, I will mark your programming style: eliminate useless steps, organize your solution and divide each problem into appropriate (not a gazillion) subproblems (with their own functions). A great deal of your mark will come from using objected oriented programming correctly. This means that getting the right output to print is not the only goal. It is how you go about getting that output using proper class and function definitions, together with inheritance, where possible.

References:
Textbook, chapter 8
Class notes

Program to make:
Construct a Python program that implements an `Employee` and a `SalariedEmployee` class. The `SalariedEmployee` class will inherit properties (data and methods) from the `Employee` class.

You must have the following methods in the <u>`Employee` class</u> (you may also have additional methods, depending on how you break these down):

`__init__`
: A constructor method that takes as input the employee's last name (string), first name (string), and hourly pay rate (float). It initializes any object variables that you may need. If the hourly pay rate is missing, initialize it to minimum wage ($11.20).

`__repr__`
: Returns a string describing the Employee object as the word "Employee" followed by the employee's name. E.g. Employee: Santa Claus

`__add__`
: Sets the '+' operator to return the sum of two employee's hourly pay rates. This might be used, for example, when calculating whether it works to move a group of employees onto a certain project and you need to know their combined hourly cost.

`printCheque`
: This is the main function for the Employee. It will print out a cheque indicating the current pay amount, along with the gross pay and deductions. This function takes as

input the number of hours the employee has worked for this pay period (i.e. this week). The hours are given as a floating point number.

The function calculates the gross pay as hours multiplied by salary. The employee receives double salary for any time over 40 hours.

A tax amount must also be calculated and then deducted from the gross pay, to achieve the net pay amount which you write on the cheque. Tax is calculated as 15% of pay, unless the yearly pay is over $42,000, or would be over $42,000 if the pay each week is this amount. If yearly pay would be over $42,000, then tax is 22% on the full amount of the gross pay.

The cheque must also contain a summary of the gross pay and deductions.

For example,

```
>>>santa = Employee('Claus', 'Santa', 55)
>>>santa.printCheque(40)
-----------------------------------------------------------------

PAY TO:  Santa Claus                   AMOUNT:  $1716.00


Gross Pay:  $2200.00
Deductions:
   Tax       $484.00
-----------------------------------------------------------------

>>>rudy = Employee('Reindeer', 'Rudolf', 10)
>>>rudy.printCheque(52)
-----------------------------------------------------------------

PAY TO:  Rudolf Reindeer               AMOUNT:  $544.00


Gross Pay:  $640.00
Deductions:
   Tax        $96.00
-----------------------------------------------------------------

>>>'{:.2f}'.format(santa + rudy)
65.00
```

The class `SalariedEmployee` inherits from the class `Employee`. However, there are a few changes in the way that a `SalariedEmployee` is treated:

1. Salaried employees are initialized with their last name, first name, and yearly salary amount (not hourly amount). If no yearly salary is specialized, initialize the employee with a salary of $25,000.
2. Salaried employees always get paid for 40 hours a week (1/52 of their salaried amount), no matter how many hours they actually work.
3. Salaried employees have the regular tax deduction, but also have a benefits deduction worth 1.5% of their gross pay.
4. Salaried employees have vacation time that builds up or decays with each pay period. The vacation time extends beyond the current cheque, i.e. it is always there. Vacation time is earned for each hour worked over 45 hours, within a week. Similarly, vacation

time is lost for each hour not worked, below 35 hours within a week. If a salaried employee works between 35 and 45 hours in a week, vacation time does not change.

5. These employees must have their pay cheque indicate the benefits deduction and the vacation time available.

For example:

```
>>>tf = SalariedEmployee('Fairy', 'Tooth', 67000)
>>>tf.printCheque(60)
-----------------------------------------------------------------

PAY TO:  Tooth Fairy                   AMOUNT:  $985.67


Gross Pay:  $1288.46
Deductions:
   Tax       $283.46
   Benefits   $19.33
Vacation:    15 hours
-----------------------------------------------------------------

>>>'{:.2f}'.format(santa + tf)
87.21

>>> tf.printCheque(32)
-----------------------------------------------------------------

PAY TO:  Tooth Fairy                   AMOUNT:  $985.67


Gross Pay:  $1288.46
Deductions:
   Tax       $283.46
   Benefits   $19.33
Vacation:      12 hours
-----------------------------------------------------------------

>>>bb = SalariedEmployee('Bunny', 'Bugs')
>>>bb.printCheque(40)
-----------------------------------------------------------------

PAY TO:  Bugs Bunny                   AMOUNT:  $401.44


Gross Pay:   $480.77
Deductions:
   Tax        $72.12
   Benefits    $7.21
Vacation:       0 hours
-----------------------------------------------------------------
```

Note that the '+' operator still works for finding combined hourly pay rates.

When you are printing dollar amounts, you may use one of the Python libraries called `locale`, which will give you nice amounts with a "`$`" in front. Here is an example of using `locale`:

```
>>> import locale
>>> locale.setlocale(locale.LC_ALL, '')
'English_Canada.1252'
>>> locale.currency(35.67897)
'$35.68'
```

Here is an outline from which to start your program:

```
## Put file header here

import locale #for currency ($number)
locale.setlocale(locale.LC_ALL, '')

class Employee(________):

    def __init__(________):
        #initialize the object

    def __repr__(________):
        #remember to return a string

    def __add__(________):
        #function to return the combined salary of two objects

    #Other auxilliary functions for things like finding the pay
    #and the tax amount, etc.


    def printCheque(________):
        #print out a cheque for an employee, with pay amount and
        #summary of gross pay and all deductions




class SalariedEmployee(________):

    def __init__(________):
        #initialize new parts of objects in this class


    #auxilliary functions


    def printCheque(________):
        #print out a cheque for an employee, with pay amount and
        #summary of gross pay and all deductions.  Mostly use function
        #written for Employee, but have a few own steps due to benefits
        #deduction and vacation tracking
```