CS 2316
Individual Homework 2 – Conditionals & Loops
Due: Wednesday, September 7th, before 11:55pm
Out of 100 points

Files to submit: 1. HW2.py

This is an INDIVIDUAL assignment!
Collaboration at a reasonable level will not result in substantially similar code. Students may only collaborate with fellow students currently taking CS 2316, the TA's and the lecturer. Collaboration means talking through problems, assisting with debugging, explaining a concept, etc. You should not exchange code or write code for others.

For Help:

- TA Helpdesk – Schedule posted on class website.
- Email TA's or use T-Square Forums

Notes:

- **Don't forget to include the required comments and collaboration statement (as outlined on the course syllabus).**
- **Do not wait until the last minute to do this assignment in case you run into problems.**

Simple Functions
You will write a few python functions for practice with the language. In your HW2.py file, include a comment at the top with your name, section, GTID/Email, and your collaboration statement. Also include each of the following functions.

1. **gtUsername**
2. **calcGrade**
3. **magicList**
4. **wanted**
5. a**ppleFan**
6. **xmassTree**
7. **printTimes**
8. p**olyTurtleGon**

Function Name: **gtUsername**
Parameters:
Input1 - first name (str)
Input2 – last name (str)
Input3 – an integer (int)
Return Value:
A string representing GT usernames. (str)

Test Cases:
gtUsername(“George”, “Burdell”, 3) --> “gburdell3”
gtUsername(“Barrack”, “Obama”, 1) --> “bobama1”
gtUsername(“Steve”, “Jobs”, 17) --> “sjobs17”

Description:
Write a function that takes in 3 parameters – student’s first name, last name and an integer. The function returns a string having a format similar to that used for a GT username: first letter of first name + last name + number.
Note that letters must be lowercase.

Function Name: **calcGrade**
Parameters:
NumGrade - a list of integers representing one’s grades.
Return Value:
A list of Letter Grades. (list of single letter strings)

Test Cases:
calcGrade([99,78,87,65]) --> ['A','C','B','D']
calcGrade([99,99,89]) --> ['A','A','B']

Description:
Write a function to convert number grades into letter grades based on the following grade breakdown. Note that the letter grades are upper case.
“A” = [90,100]
“B” = [80,90)
“C” = [70,80)
“D” = [60,70)
“F” = [0,60)

Function Name: **magicList**
Parameters:
Input- A list of integers
Return Value:
A Boolean (True or False) that shows whether the input is a Magic List or not.

Test Cases:
magicList([2,1,6,5,4,7,8,3,9,0]) --> True
magicList([3,2,1,4,5,7,6,9,0]) --> False
magicList([2,2,1,4,5,6,3,7,3,4]) --> False

Description:
Write a function that takes in a list of numbers as parameter and returns a Boolean value. A magic list is defined as a list of numbers where all the numbers from 0-9 are represented and do not repeat (and no other extraneous numbers appear). For example, [0,1,2,3,4,5,6,7,8,9] is definitely a Magic List.

Note: you might find build-in function sorted() or .sort() useful.

Function Name: **wanted**
Parameters:
String – a given string
Letter- a target letter (string of length 1)

Return Value: return the integer index position of the first target letter, or return None (of NoneType) if the target letter is not found.

Test Cases:
wanted(“Je t'aime”, “a”) --> 5
wanted(”Rolling in the deep”, “g”) -->6

Description:
Write a function that takes in a string and a single letter string, then returns the index position of the given target letter or None if the letter is not found. Use

looping, you may not use the built in .find() method. If the target letter appears in more than one place in the string, return the index of the first occurrence.

Function Name: **appleFan**
Parameters:
answer1 – a boolean (True or False) representing whether the user has “IPods”
answer2 - a boolean (True or False) representing whether the user has “ITouchs”
answer3 - a boolean (True or False) representing whether the user has “IPhones”
answer4 - a boolean (True or False) representing whether the user has “IQuits” (in memorial of The Steve Jobs)

Return Value:
The string “I have” + the designated Apple products + “.”
Test Cases:

1. appleFan(True, True, True, True) --> “I have IPods, ITouchs, IPhones, IQuits.”
2. appleFan (True, False, True, False) --> “I have IPods, IPhones.”
3. appleFan (False, False, False, False) --> “Not an Apple fan.”

Description:
Write a function that returns a string based on the apple products selected by the inputs. Use the inputs True and False. The function should return the string “I have ” concatenated with the designated apple products. The four apple products should be: "IPods" "ITouchs" "IPhones" and "IQuits" (which Apple Co. released on August 24th in memorial of the resignation its CEO Steve Jobs). If the user doesn’t have any Apple products, return the string “Not an Apple fan.”.

Function Name: **xmassTree**
Parameters:
X – an integer that specifies the size of the xmass tree base.
(you may assume that the input will be between 3 and 61 inclusive,
and that it will be an odd number)
Return Values:
none

Description: Your function will draw a Christmas tree on-screen using the print function. Your Christmas tree will have one Asterisk character at the top level, three at the 2nd to top level, five at the 3rd level, and so on, until it reaches the bottom level. After the bottom level you should have three lines with a single Asterisk to represent the trunk. Note that your xmass tree is NOT centered on the screen. The left hand side of the base of the tree will line up with the left hand side of the python window. You will have to figure out for yourself how many spaces to leave for the 1st and subsequent level of the xmass tree so that everything works out right!
Here are some example runs:

```
>>>xmassTree(5)
  *
 ***
*****
  *
  *
  *
```

```
>>>xmassTree(9)

    *
   ***
  *****
 *******
*********
    *
    *
    *
```

Function Name: **printTimes**

Parameters:

N – an integer that limits the upper bound of the times table (inclusive)

Return Value:

none

Description:

You are hired to develop an educational software package. Your first job: Write a function printTimes(N) that will print the times tables (1 to N) on the screen for any positive N. When your function is called with N = 9 for example, it should print the following:

```
Times: 1        2       3       4       5       6       7       8       9

1       1       2       3       4       5       6       7       8       9

2       2       4       6       8       10      12      14      16      18

3       3       6       9       12      15      18      21      24      27

4       4       8       12      16      20      24      28      32      36

5       5       10      15      20      25      30      35      40      45

6       6       12      18      24      30      36      42      48      54

7       7       14      21      28      35      42      49      56      63

8       8       16      24      32      40      48      56      64      72

9       9       18      27      36      45      54      63      72      81
```

Note that your function must **print** a header (Times: 1...N) and a first column number that goes from 1..N, while the interior of the grid is the X * Y value. Hint:

Using two loops (one inside of the other) is an easy (but not the only) way to accomplish this. You may want to use tab characters (“\t”) to space your grid out correctly.

Function Name: **polyTurtleGon**
Parameters:
Input1- an integer representing the number of sides
Input2- an integer representing the length of the sides
Return Values:
none

Description:
Write a function that uses the turtle module to draw a polygon with the given number and length of sides. You may want to calculate the angles first and then plot the polygon.

Hint: Each angle in the n-polygon is 180(n-2)/n.

Grading

You will earn points as follows for each function that works correctly according to the specifications.

gtUsername	**5**
calcGrade	**5**
magicList	**10**
wanted	**10**
appleFan	**10**
xmassTree	**20**
printTimes	**25**
p**olyTurtleGon**	**15**