CS 1301 Individual Homework 3 – Conditionals & Loops
Due: Monday, June 3rd, before 11:55pm
Out of 100 points

File to submit: HW3.py

Students may only collaborate with fellow students currently taking CS 1301, the TA's, and the lecturer. Collaboration means talking through problems, assisting with debugging, explaining a concept, etc.

For Help:

- TA Helpdesk – Schedule posted on class website.
- Email TA’s or use Piazza

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*
- **Read the entire specifications document before starting this assignment.**

Functions

You will write a few python functions for practice with the language. In your HW3.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 below. . For purpose of this homework, you may assume that all inputs will be valid.

Function Name: **checkHeight**

Parameters:

height - an integer representing the user's height in centimeters

Return Value:

Either the string “Have a great ride!” or the string “Sorry. You must be at least 1 meter 32 cm to ride.”

Test Cases:

checkHeight(125) --> “Sorry. You must be at least 1 meter 32 cm to ride.”
checkHeight(169) --> “Have a great ride!”

Description:

Write a function for the Superman ride at Six Flags that determines whether the user is taller than 1 meter 32 centimeters so that he or she can ride a roller coaster. If the user's height, which is provided by the parameter height, is greater than or equal to the minimum height, return the string 'Have a great ride!'. Otherwise, return the string 'Sorry. You must be at least 1 meter 32 cm to ride.'

Function Name: **countDownByThrees**

Parameters:

start - an integer greater than 0 representing the starting number of the countdown

Return Value:

None

Test Cases:

```
 >>> countDownByThrees(5)
5
2
Blast Off!

>>> countDownByThrees(1)
1
Blast Off!

>>>countDownByThrees(6)
6
3
Blast off!
```

Description:

Write a function to count down from a given number by threes. The function should print the numbers from the given number to 1 (decreasing by 3 each time...if you go past 1, don't print it!) in descending order, with each number being printed on its own line. After printing the required numbers, on a separate line, print the string 'Blast off!'

Function Name: **allLetters**

Parameters:

    userString - A String.

Return:

    A String.

Test Cases:

    allLetters("gburdell3") --> “gburdell”
    allLetters("Hello@World.com") --> “HelloWorldcom”
    allLetters("2012") --> “”

Description:

Write a function that uses a while loop to create and return a new string that contains only the letters of the original input. If the input string has no letters, you must return an empty string.

You **MUST use a while loop** for this problem! Hint: "import string" and use the "in" check along with the "string.ascii_letters" constant to determine if each character is a letter or not.

Function Name: **complimentMaker**

Parameters:

answer1 – a boolean (True or False) representing whether the user is "super"
answer2 - a boolean (True or False) representing whether the user is "nice"
answer3 - a boolean (True or False) representing whether the user is "smart"
answer4 - a boolean (True or False) representing whether the user is "cool"

Return Value:

The string “You are ” + the designated compliments + “.”

Test Cases:

1. complimentMaker(True, True, True, True) --> “You are super nice smart cool.”
2. complimentMaker(True, False, True, False) --> “You are super smart.”
3. complimentMaker(False, False, False, False) --> “No Comment.”

Description:

Write a function that **returns** a string of compliments based on the adjectives selected by the inputs. Use the inputs True and False. The function should return the string “You are ” concatenated with the compliments that are true. The four compliments should be: "super" "nice" "smart" and "cool". If none of the compliments are true, return the string “No comment.” instead.

Function Name: **multiplicationTables**

Parameters:

number – an integer representing the number for which you want to create a multiplication table
limit – an integer representing how high you want the multiplication table to go

Return Value:

None

Test Cases:

```
>>>multiplicationTables(3, 4)
3*0 = 0
3*1 = 3
3*2 = 6
3*3 = 9
3*4 = 12

>>>mutiplicationTables(5, 2)
5*0 = 0
5*1 = 5
5*2 = 10
```

Description:

Write a function that takes in a two numbers. The first number is the number for which you wish to make a multiplication table; the second is how far you want the table to go. Have your function print out lines of the multiplication table as shown in the test case, by **printing** the number, the multiplication sign, the number you are multiplying it by, the equal sign, and what they equal. Note that the number*number do not have spaces between them, while the space-equal-number does!

Function Name: replaceLetter(10pts)

Description:
Write a function that takes in three parameters: a string that consists of one letter (the letter that will be replaced), a second string that consists of one letter (the replacement letter), and a string. Your function should find all the occurrences of your first parameter in the string. Every time that the first parameter letter occurs, replace that letter with the second parameter‘s letter. Note that uppercase letters and lowercase letters are considered different letters.

Parameters:
-oldLet (String): The letter you want to replace
-newLet (String): The letter that will replace oldLet
-aStr (String): A string

Return Value:
(String) The new string with all the correct letters replaced

Test Cases:

1. replaceLetter("e", “a“, “I like CS2316!”) returns “I lika CS2316!”
2. replaceLetter("L", “c“, “HELLO world”) returns “HEccO world”
3. replaceLetter(“Q”, “R”, “MY NAME IS MILES”) returns “MY NAME IS MILES”

Function Name: numMountainRange (10pts)

Description:

Write a function that takes in the number of rows of the mountain range as a parameter. The function will then draw a number mountain range on screen using the print function. See screenshots below in the test cases for clarification. DO NOT HARD CODE THE PRINTOUTS.

Parameter:

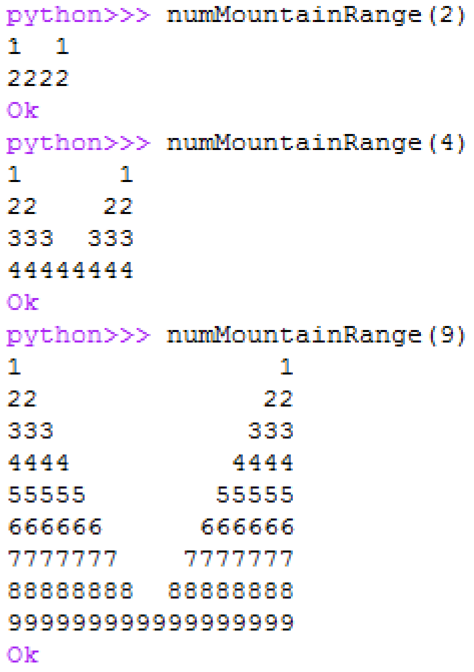
X (Integer): An integer that specifies the number of rows of the mountain range. You may assume the number is an integer between 2-9.

Return Values:

None

Test Cases:

You have X number of rows, but note that there are two 1s, four 2s, six 3s, eight 4s, etc.



```
python>>> numMountainRange(2)
1  1
2222
Ok
python>>> numMountainRange(4)
1      1
22    22
333  333
44444444
Ok
python>>> numMountainRange(9)
1                1
22              22
333            333
4444          4444
55555        55555
666666      666666
7777777    7777777
88888888  88888888
999999999999999999
Ok
```

Function Name: **printTimestable**

Parameters:
    none
Return Value:
    none

You are hired to develop an educational software package. Your first job: Write a function printTimestable() that will *print* the times tables (up to 9) on the screen. When your function is called, 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...9) and a first column number that goes from 1..9, 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 to space your grid out correctly.

Function Name: **printTimes**

Parameters:

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

Return Values:

none

Your boss was impressed with your 9x9 times table function. Now he wants you to modify the function so that it will work for for any sized times table. Write a printTimes(N) function that will print a times table from 1 up to N, for any positive number N.

For example, when your function is called as printTimes(4) , it should print the following:

```
Times: 1    2    3    4
1      1    2    3    4
2      2    4    6    8
3      3    6    9    12
4      4    8    12   16
```

Grading Rubric

checkHeight		**5pts**
- function takes in a height in centimeters	2	
- function returns correct output for all valid inputs	3	
countDownByThrees		**5pts**
- function prints numbers starting at specified parameter	2	
- function print decreases by 3 every time	2	
- function stops printing at 1 and ends with "blastoff"	1	
allLetters		**10 pts**
- uses a while loop	4	
- returns correct output for any valid input	6	
complimentMaker		**10pts**
- function accepts parameters as booleans	4	
- function correctly generates string output	6	
multiplicationTables		**10pts**
- function accepts two parameters	2	
- function correctly generates multiplication tables	5	
- function displays the multiplication table in proper format	3	
replaceLetter		**10pts**
- Finds all letters in the string that need to be replaced.	5	
- Returns the correct string with the replaced letters	5	
numMountainRange		**20pts**
- Correct number of rows and correct number in rows	10	
- Correct shape (-5 if hard coded)	10	
printTimestable		**10pts**
- function prints correct multiplication output	5	
- function prints with correct formatting	5	
printTimes		**20pts**
- function accepts an integer *n* as a paremeter	5	
- function correctly prints *n x n* times table	5	
- function nicely formats the output	5	
- function does not return any value	5	

Elements of this homework created by Catherine Hwang and James Moore