

NAME:
 EMAIL:
 SIGNATURE:
 CIRCLE COURSE SECTION: MW 9-11 MW 11-1 MW 6-8
 TTh 9-11 TTh 1-3

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
Total	

Lehman College, CUNY
CMP 167 Final Exam, Version 2, Spring 2015

1. What will the following code print:

```
s = "oBJcBJaBJmBJl"
a = s[0:3]
print(a.title())
names = s.split("BJ")
print(names)
b,c,d = names[1],names[2],names[3]
print(c,d)
print(a[0]+b.upper()+c+d+names[4])
print('print_endline "', a.lower(),'")')
```

Output:

2. Write a **complete program** to calculate how much something will weigh on the Moon. Your program should prompt the user for the weight on the Earth and then print out the weight on the Moon. For example, if the user enters 100, your program should print out 17.


The weight of an item on the Moon is 17% of its weight on earth.

3. What is output of the code below:

```
def prob4(amy, beth):  
    if amy > 4:  
        print("Easy case")  
        kate = -1  
    else:  
        print("Complex case")  
        kate = helper(amy,beth)  
    return(kate)
```

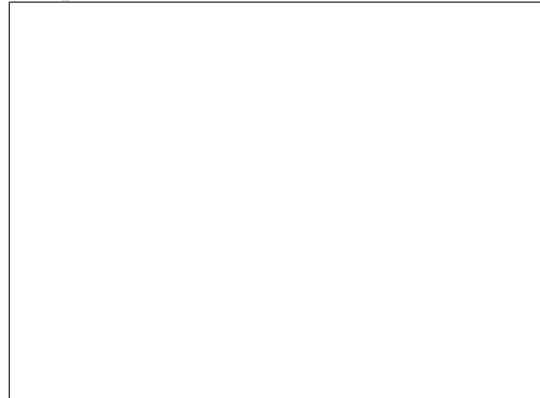
```
def helper(meg,jo):  
    s = ""  
    for j in range(meg):  
        print(j, ": ", jo[j])  
        if j % 2 == 0:  
            s = s + jo[j]  
            print("Building s:", s)  
    return(s)
```

Output:



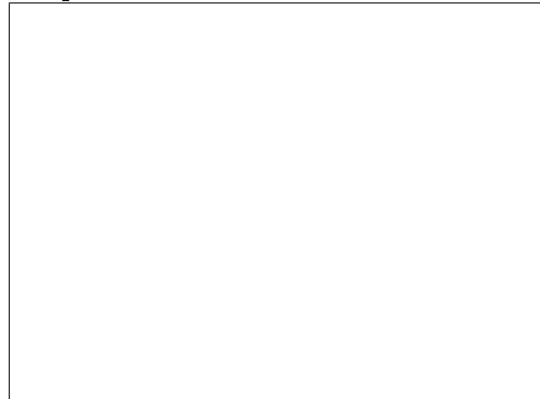
(a) `r = prob4(6,"city")`
`print("Return: ", r)`

Output:



(b) `r = prob4(2,"university")`
`print("Return: ", r)`

Output:



(c) `r = prob4(4,"new york")`
`print("Return: ", r)`

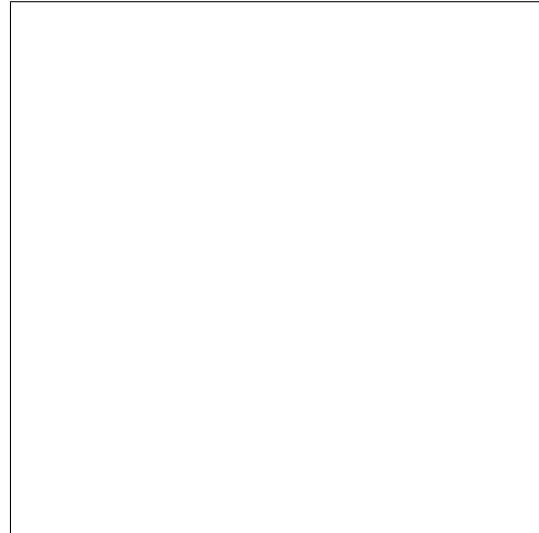
4. Given the following program and input file, what is printed:

```
def prob5V1():  
    c = 0  
    infile=open("places.txt","r")  
    for line in infile.readlines():  
        if len(line) < 10:  
            print("Short Line: ", end = "")  
            c = c + 1  
            print(line)  
    print("Num short lines is", c)  
  
prob5V1()
```

places.txt

Ontario
Quebec
Nunavut
Yukon
Alberta
New Brunswick

Output:



5. (a) Write a function that takes number between 1 and 7 as a parameter and returns the corresponding ordinal number as a string. For example, if the parameter is 1, your function should return **"first"**. If the parameter is 2, your function should **"second"**, etc. If the parameter is not between 1 and 7, your function should return the empty string.
- (b) Write a `main()` that allows the user to enter a number and calls your function to show that it works.

6. Complete the following program, which sets up a graphics window and turtle, draws an octagon (8-sided figure) to the window, and then prints a closing message and closes the graphics window when mouse is clicked. That is, write the functions `setUp()`, `drawOctagon()`, and `conclusion()`:

```
import turtle

def main():
    w,t = setUp()    #sets up a graphics window and turtle
    drawOctagon(t)   #draws a octagon using the turtle
    conclusion(w)     #prints goodbye and closes window on click

main()
```

7. (a) Write a **complete** program that prompts the user for a file name and prints the number of lines in the file.

- (b) Write a **complete** program that prints the total area stored in a data file. Your program should open the file, `cityData.csv`, and calculate the sum of the areas, where the area is the last value in each line. Note that the first line should not be used since it contains the column headers and not data. The data is separated by commas (","). Your program should **print** the sum that you calculated.

cityData.csv:

```
Borough, Population, Area (square miles)
Bronx, 1385108, 42
Brooklyn, 2504700, 71
Manhattan, 1585873, 23
Queens, 2230722, 109
Staten Island, 468730, 58
```

8. Write the Python code for the algorithms below:

(a) `getInput()`

```
Ask user for number between 0 and 100
Until they enter a number between 0 and 100
    Print error message
    Ask user for a number between 0 and 100
Return the number entered
```

(b) `merge(ls, mid)`

```
Initialize the variables: set newList to be an empty list, set counters i to be 0
and j to be mid.
While i < mid and j < len(ls):
    If ls[i] >= ls[j], append ls[i] to the newList and increment i.
    Else: append ls[j] to the newList and increment j.
While i < mid:
    Append ls[i] to the newList and increment i.
While j < len(ls)
    Append ls[j] to the newList and increment j.
Return newList
```

9. In lab, we wrote a Tic-Tac-Toe program. Modify the program to stop the game when someone has won. Your program should check for a winner each move. Your program should continue playing until there is a winner or until all squares are filled.

Clearly mark your changes to the design below:

```
#Second Version of Tic-Tac-Toe
from turtle import *
def setUp():
    win, tic = Screen(), Turtle()
    tic.speed(10)
    win.setworldcoordinates(-0.5,-0.5,3.5, 3.5)
    for i in range(1,3):
        tic.up()
        tic.goto(0,i)
        tic.down()
        tic.forward(3)
    tic.left(90)
    for i in range(1,3):
        tic.up()
        tic.goto(i,0)
        tic.down()
        tic.forward(3)
    tic.up()
    board = [["", "", ""], ["", "", ""], ["", "", ""]]
    return(win,tic,board)
def playGame(tic,board):
    for i in range(4):
        x,y = eval(input("Enter x, y coordinates for X's move: "))
        tic.goto(x+.25,y+.25)
        tic.write("X",font=('Arial', 90, 'normal'))
        board[x][y] = "X"
        x,y = eval(input("Enter x, y coordinates for O's move: "))
        tic.goto(x+.25,y+.25)
        tic.write("O",font=('Arial', 90, 'normal'))
        board[x][y] = "O"
        x,y = eval(input("Enter x, y coordinates for X's move: "))
        tic.goto(x+.25,y+.25)
        tic.write("X",font=('Arial', 90, 'normal'))
        board[x][y] = "X"
def checkWinner(board):
    for x in range(3):
        if board[x][0] != "" and (board[x][0] == board[x][1] == board[x][2]):
            return(board[x][0]) #we have a non-empty row that's identical
    for y in range(3):
        if board[0][y] != "" and (board[0][y] == board[1][y] == board[2][y]):
            return(board[0][y]) #we have a non-empty column that's identical
    if board[0][0] != "" and (board[0][0] == board[1][1] == board[2][2]):
        return(board[0][0])
    if board[2][0] != "" and (board[2][0] == board[1][1] == board[2][0]):
        return(board[2][0])
    return("No winner")
def main():
    win,tic,board = setUp() #Set up the window and game board
    playGame(tic,board) #Ask the user for the moves and display
    print("\nThe winner is", checkWinner(board)) #Check for winner
```


10. (a) Write a **complete** class that keeps tracks of information about junk food. Your class, `JunkFood` should contain instance variables for the `name`, `calories`, `weight` and `expirationDate`, and should have a constructor method as well as a method, `calorieDensity()`, that returns the calories per ounce (`calories/weight`) for the junk food and a method, `getExpiration()`, that returns the expiration date.

- (b) Write a function that takes as input a list of `JunkFood`, called `shoppingList`, and returns the most calorie rich food, by weight (i.e. the maximum of all the calorie densities of the junk food in the inputted list):

```
def bestValue(shoppingList):
```

Useful String Methods: (from p 140 of textbook)

Function	Meaning
<code>s.capitalize()</code>	Copy of <code>s</code> with only the first character capitalized.
<code>s.center(width)</code>	Copy of <code>s</code> is centered in a field of given width.
<code>s.count(sub)</code>	Count the number of occurrences of <code>sub</code> in <code>s</code> .
<code>s.find(sub)</code>	Find the first position where <code>sub</code> occurs in <code>s</code> .
<code>s.join(list)</code>	Concatenate <code>list</code> into a string using <code>s</code> as a separator.
<code>s.ljust(width)</code>	Like <code>center</code> , but <code>s</code> is left-justified.
<code>s.lower()</code>	Copy of <code>s</code> with all characters converted to lowercase.
<code>s.lstrip()</code>	Copy of <code>s</code> with leading whitespace removed.
<code>s.replace(oldsub,newsub)</code>	Replace all occurrences of <code>oldsub</code> in <code>s</code> with <code>newsub</code> .
<code>s.rfind(sub)</code>	Like <code>find</code> , but returns rightmost position.
<code>s.rjust(sub)</code>	Like <code>center</code> , but <code>s</code> is right-justified.
<code>s.rstrip()</code>	Copy of <code>s</code> with trailing whitespace removed.
<code>s.split()</code>	Split <code>s</code> into a list of substrings.
<code>s.title()</code>	Copy of <code>s</code> with first character of each word capitalized.
<code>s.upper()</code>	Copy of <code>s</code> with all characters converted to uppercase.

Useful Turtle Methods: (from <http://docs.python.org/3.0/library/turtle.html>)

Function	Meaning
<code>t.forward(d)</code>	Move turtle forward <code>d</code> steps
<code>t.backward(d)</code>	Move turtle backward <code>d</code> steps
<code>t.right(angle)</code>	Turn turtle <code>angle</code> degrees to the right
<code>t.left(angle)</code>	Turn turtle <code>angle</code> degrees to the left
<code>t.up()</code>	Pull the pen up: no drawing when moving
<code>t.down()</code>	Pull the pen down: drawing when moving
<code>t.color(c)</code>	Change pen color to color <code>c</code>
<code>t.goto(x,y)</code>	Move turtle to coordinates <code>(x,y)</code>
<code>w.bgcolor(c)</code>	Change background color to color <code>c</code>
<code>w.setworldcoordinates(x1,y1,x2,y2)</code>	Resize drawing area with lower left corner as <code>(x1,y1)</code> and upper right corner <code>(x2,y2)</code>
<code>w.exitonclick()</code>	Closes graphics window on mouse click