

Answer Key: CIS 166 Final Exam, Version 1, Spring 2015

1. What will the following code print:

```
s = "Ada=>Lovelace=>Charles=>Babbage"
a = s[0:3]
print(a.upper())
names = s.split("=>")
print(names)
b,c,d = names[1],names[2],names[3]
print(c,d)
print(b[-1]+"n"+d[-2]+"ine")
print('Put_line: ('', a.lower(),'''))
```

Answer Key:

```
ADA
['Ada', 'Lovelace', 'Charles', 'Babbage']
Charles Babbage
engine
Put_line: ( " ada ")
```

2. Write a **complete program** to calculate how much something will weigh on Mars. Your program should prompt the user for the weight on the Earth and then print out the weight on Mars. For example, if the user enters 100, your program should print out 38.

The weight of an item on Mars is 38% of its weight on earth.

Answer Key:

```
#Computes weights on Mars
def main():
    earthWeight = eval(input('Enter earth weight: '))
    marsWeight = earthWeight * 0.38
    print('The weight on Mars is:', marsWeight)

main()
```

3. What is output of the code below:

```
def prob4(fred, george):
    if fred < 2:
        print("Small case")
        harry = -1
    else:
        print("Complex case")
        harry = helper(fred,george)
    return(harry)

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

(a) `r = prob4(0,"herbert")`
`print("Return: ", r)`

Output:

Answer Key:

Small case
 Return: -1

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

Output:

Answer Key:

Complex case
 0 : l
 Building s: l
 1 : e
 Return: l

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

Output:

Answer Key:

Complex case
 0 : c
 Building s: c
 1 : o
 2 : l
 Building s: cl
 3 : l
 Return: cl

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) > 7:
            print("Long Line: ", end = "")
            c = c + 1
        print(line)
    print("Num long lines is", c)

prob5V1()
```

places.txt

Vandenberg
 Wright-Patterson
 Laughlin
 Dover
 Charleston
 San Antonio

Output:

Answer Key:

Long Line: Vandenberg

Long Line: Wright-Patterson

Long Line: Laughlin

Dover

Long Line: Charleston

Long Line: San Antonio

Num long lines is 5

5. (a) Write a function that takes number between 1 and 7 as a parameter and returns the corresponding day of the week as a string. For example, if the parameter is 1, your function should return "Monday". If the parameter is 2, your function should return "Tuesday", etc. If the parameter is not between 1 and 7, your function should return the empty string.

Answer Key:

```
def returnDay(num):  
    if num == 1:  
        return "Monday"  
    elif num == 2:  
        return "Tuesday"  
    elif num == 3:  
        return "Wednesday"  
    elif num == 4:  
        return "Thursday"  
    elif num == 5:  
        return "Friday"  
    elif num == 6:  
        return "Saturday"  
    elif num == 7:  
        return "Sunday"  
    else:  
        return ""
```

- (b) Write a `main()` that allows the user to enter a number and calls your function to show that it works.

Answer Key:

```
#intro comment  
def main():  
    num = eval(input("Enter a number"))  
    test1 = returnDay(num)
```

```

        print ("Testing my function:",num,"is", test1)
    main()

```

6. Complete the following program, which sets up a graphics window and turtle, draws a hexagon (6-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()`, `drawHexagon()`, and `conclusion()`:

```

import turtle

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

main()

```

Answer Key:

```

def setUp():
    trey = turtle.Turtle()
    win = turtle.Screen()
    return(win,trey)

def drawHexagon(t):
    for i in range(6):
        t.forward(100)
        t.right(360/6)

def conclusion(w):
    print("Goodbye!")
    w.exitonclick()

```

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

Answer Key:

```

#some comments

def main():
    fileName = input('Enter file name: ')
    infile = open(fileName)
    data = infile.read()
    print("Number of lines:", data.count("\n"))

    infile.close()

```

- (b) Write a **complete** program that prints the total population stored in a data file. Your program should open the file, `population.csv` and sum the last values 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 total sum that you calculated.

population.csv:

```
Borough, 2000 Population, 2010 Population
Bronx, 1332650, 1385108
Brooklyn, 2465326, 2504700
Manhattan, 1537195, 1585873
Queens, 2229379, 2230722
Staten Island, 443728, 468730
```

Answer Key:

```
#some comments
```

```
def main():
    sum = 0
    infile = open("population.csv")
    infile.readline() #Ignore first line, since no numbers
    lines = infile.readlines()
    for l in lines:
        cells = l.split()
        sum = sum + eval(cells[2])

    print("Total population:", sum)

    infile.close()
```

8. Write the Python code for the algorithms below:

- (a) getInput()
Ask user for an even number
Until they enter an even number
 Print error message
 Ask user for an even number
Return the even number entered

Answer Key:

```
def getInput()
    x = int(eval('Enter an even number: '))
    while x % 2 != 0:
        print('Not an even number!')
        x = int(eval('Enter an even number: '))
    return(x)
```

- (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], then 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

```

Answer Key:

```

def merge(ls, mid):
    newList = []
    i, j = 0, mid
    while i < mid and j < len(ls):
        if ls[i] < ls[j]:
            newList.append(ls[i])
            i += 1
        else:
            newList.append(ls[j])
            j += 1
    while i < mid:
        newList.append(ls[i])
        i += 1
    while j < len(ls)
        newList.append(ls[j])
        j += 1
    Return newList

```

9. In lab, we wrote a Tic-Tac-Toe program. Change the program to check for a winner after each move and keep track of the number of times this occurs. Your program should print out a message if someone has a winning configuration, print out the total winning configurations seen so far, and continue playing. 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

```

Answer Key:

```

#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):
    numWinners = 0                                ###ADDED
    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"
        if checkWinner(board):                    ###ADDED
            print('X has a winning configuration!') ###ADDED
            numWinners = numWinners + 1           ###ADDED
        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"
        if checkWinner(board):                    ###ADDED
            print('O has a winning configuration!') ###ADDED
            numWinners = numWinners + 1           ###ADDED
        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"
        if checkWinner(board):                    ###ADDED
            print('X has a winning configuration!') ###ADDED
            numWinners = numWinners + 1           ###ADDED
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) In lab, we processed name data maintained by the Social Security Administration. Write a **function** that takes as input a string of Social Security Administration name data and returns the gender ('F' for female and 'M' for male).

Here are some sample lines from the NY.txt file containing the data for New York State:

NY.txt
NYM1910Herbert,83
NYM1910Leo,80
NYM1910Andrew,79
NYM1910Ernest,79
NYM1910Milton,79

Answer Key:

```
#Function that extracts gender from the line:  
def getGender(line):  
    return(line[2])
```

- (b) Write a **complete program** that uses your function above to count the number of entries by gender. Your program should open the file `NY.txt`, count the number of lines for males and the number of lines for females, and print the results.

Answer Key:

```
def main():  
    infile = open("NY.txt", "r")  
    countM, countF = 0,0  
    for line in infile:  
        if getGender(line) == 'M':  
            countM = countM + 1  
        else:  
            countF = countF + 1  
    print("Number of males is:", countM)  
    print("Number of females is:", countF)  
  
main()
```