Name : _____

1. *(2 points)*
Section/Grading TA Name: _____

- Integrity: By taking this exam, you pledge that this is your work and you have neither given nor received inappropriate help during the taking of this exam in compliance with the Academic Honor Code of Georgia Tech. Do NOT sign nor take this exam if you do not agree with the honor code.
- Devices: If your cell phone, pager, PDA, beeper, iPod, or similar item goes off during the exam, you will lose 10 points on this exam. Turn all such devices off and put them away now. You cannot have them on your desk.
- Academic Misconduct: Academic misconduct will not be tolerated. You are to uphold the honor and integrity bestowed upon you by the Georgia Institute of Technology.
  - Keep your eyes on your own paper.
  - Do your best to prevent anyone else from seeing your work.
  - Do NOT communicate with anyone other than a proctor for ANY reason in ANY language in ANY manner.
  - Do NOT share ANYTHING during the exam. (This includes no sharing of pencils, paper, erasers).
  - Follow directions given by the proctor(s).
  - Stop all writing when told to stop. Failure to stop writing on this exam when told to do so is academic misconduct.
  - Do not use notes, books, calculators, etc during the exam.
- Time: Don't get bogged down by any one question. If you get stuck, move on to the next problem and come back once you have completed all of the other problems. This exam has 8 questions on 9 pages including the title page. Please check to make sure all pages are included. You will have 50 minutes to complete this exam.

I commit to uphold the ideals of honor and integrity by refusing to betray the trust bestowed upon me as a member of the Georgia Tech community. I have also read and understand the requirements outlined above.

Signature: _____

Question	Points	Score
TA Name	2	
2. Vocabulary	6	
3. Create Nested	5	
4. Dictionary Work	6	
5. Objective Cars	5	
6. GUI drawing Shoes	12	
7. Print v. Return	3	
8. CSV Average Ints	12	
Total:	51	

2. *(6 points)*
For each of the following vocabulary terms, write a concise 1-2 sentence definition. Be brief, and to the point.

(a) [3 pts] sequence

Solution: Any of the data types that consist of an ordered collection of elements, with each element identified by an index.

(b) [3 pts] raise

Solution: To create a deliberate exception by using the raise keyword.

3. *(5 points)*
Write a function called **createNested** that accepts a single integer parameter N. It must create an N by N (2 dimensional) data structure made out of nested lists that is filled with zeros and return it. As an example, when `createNested(3)` is called, it will return:

```
[ [0,0,0], [0,0,0], [0,0,0] ]
```

Solution:

```
def createNested(N):
  outputList = []
  for x in range(N):
     row = []
```

```
    for j in range(N):
       row.append(0)
    outputList.append(row)
  return outputList
```

Grading:
+1 for returning a new list (with any contents)
+1 for adding N sublists to it (even if aliases or empty)
+1 for having N zeros in each sublist.
+2 for not creating aliased sublists lists!

4. *(6 points)*
Will the following code produce an error when executed? If yes, explain why, if not, what will be printed?

```
aDict={ }
aList=[1,[3,4],5]
bList=[7,6,'Jay']
for item in range(len(aList)):
     aDict[ bList[item] ]= aList[item]
print(aDict)
```

Solution:

```
{'Jay': 5,    6: [3, 4],   7: 1}
```

Grading:
+1 for each correct key/value pair (in any order) -1 for each incorrect key/value pair.

Will the following code produce an error when executed? If yes, explain why, if not, what will be printed?

```
bDict={ }
cList=[1,2,[3,4] ]
dList=[9,8,7]
for item in range(len(dList)):
     bDict[cList[item]]=dList[item]
print(bDict)
```

Solution: An error will occur because the list `[3,4]` in cList is a mutable data type and can not be hashed (so can not be a key in the dictionary).

Grading: +1 point if they say an error will occur.
+1 point for identifying the list [3,4] as the reason.
+1 point if they explain why (a list can not be a key because it is mutable).

5. *(5 points)*
Examine the code below. Write down exactly what would be printed if the code was executed. (If it would not run, explain why not.)

```
class E2:

    def __init__(self, paint, make):
        self.paint = paint
        self.make = make

    def printMyCar(self):
        print("I'm driving a {0} {1}, bro.".format(self.paint,self.make))

car1 = E2('red','Mercedes')
car2 = E2('blue','Audi')

car1.paint = car2.make
car2.make = car1.make

car1.printMyCar()
car2.printMyCar()
```

Solution: I'm driving a Audi Mercedes, bro. I'm driving a blue Mercedes, bro.

Grading:
+1 for Audi
+1 for Mercedes
+1 for blue
+1 for the Mercedes on the 2nd line.
+1 for getting the rest of the string "I'm driving a X Y, bro." right.
-1 for each extra incorrect lines.
-1 for writing quotes around the printouts.

6. *(12 points)*
Given the following code, draw the GUI that is produced when it is ran on the next page. Include the window with any decorations. Indicate colors, shading, or state with arrows and labels. After you draw the GUI, answer the question on the next page.

```
from tkinter import *
class MyGUI:
    def __init__(self, win):
        Button(win, text="Do it!", command=self.clicked).pack(fill=X)
        self.e1= Entry(win)
        self.e1.pack()

        AFrame=Frame(win)
        AFrame.pack()
        frame2=Frame(win)
        frame2.pack()

        self.V1=IntVar()
        self.V2=StringVar()
        self.V2.set("0")

        r1=Radiobutton(AFrame, text='Nike', variable=self.V1, value=0)
        r1.grid(row=2, column=2)
        r2=Radiobutton(frame2, text='Adidas', variable=self.V2, value='200')
        r2.grid(row=0, column=0)
        r1=Radiobutton(AFrame, text='Reebok', variable=self.V1, value=300)
        r1.grid(row=1, column=1)
        r2=Radiobutton(frame2, text='Converse', variable=self.V2, value='0')
        r2.grid(row=1, column=1)

    def clicked(self):
        self.Value=self.e1.get()
        self.V2.set(self.Value)

mainWin=Tk()
MyGUI(mainWin)
mainWin.mainloop()
```

Draw your GUI here:

Solution:

The GUI: 

Grading:
Before button press:
+1 window correctly drawn with decorations
+1 window title (tk)
+1 Button text says "Do it!" and located at top of window.
+1 Button fills whole window horizontally
+1 Nike radio button selected
+1 Converse radio button selected.
+1 Reebok and Adidas radio buttons both NOT selected
+1 Reebok RB above and to the left of NikeRB.
+1 Adidas RB above and to the left of Converse RB.
+1 Reebok/Nike both above Adidas/Converse.

If you were to type "200" into the Entry box and press the button, what would change on the GUI?

Solution: The Converse radio button would become unselected (+1) and the Adidas radio button would become selected. (+1).

7. *(3 points)*
Pretend you are the Python interpreter and the following code has been entered and executed. Write down exactly what would be printed in the shell!

```
def return1():
    print(1)
```

```
    return 1

def someFunc():

    if return1() == 1:
        print('hi')

    if return1() == 0:
        print('bye')


someFunc()
```

Solution:

```
1
hi
1
```

Grading:
+1 for each correct line. -1 for each extra/incorrect item.

8. *(12 points)*

Write a function `averageInts` that takes in one parameter, the name of a CSV file as a string. The CSV file will contain entries similar to this example (Note that a colon is used as the delimiter!):

```
1:2:3
4:9
b:c:d:e
4:8:7
```

Your function must use the csv module to parse the file, read in the data, find all of the integers (NOT FLOATS!), and return the average of any integers in the file. Ignore any non-integer values. The CSV file can have any number of rows, and each row can have any number of items, so do not hard code a function that only works for the above example. If the function were run on the example data above, it would return 4.75 .

Solution:

```
def averageInts(name):
    import csv
    f=open(name,'r')
    reader=csv.reader(f,delimiter=':')
    counter=0
    total=0
    for row in reader:
        for thing in row:
            try:
                thing=int(thing)
                counter= counter + 1
                total= total + thing
            except:
                pass
    f.close()
    return total / counter
```

Grading:
+1 Correct function definition
+1 Opens the file
Using CSV Reader:
+1: Imports CSV
+1: Correctly creates CSV reader
+1: Iterates through the reader/gets rows out.

+1: Tries to converts each item to a int
+2: protects from exceptions with a try/except
+1: Sums all ints
+1: keeps track of how many ints were found.
+1: Closes the file
+1: Returns the correct average.