CS 2316
Exam 2 Summer 2014

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 4 questions on 8 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
1. Vocabulary	9	
2. Multiple Choice	8	
3. GUI drawing	15	
4. Phone Bill	14	
Total:	46	

1. *(9 points)*
For each of the following vocabulary terms, write a concise 1-2 sentence definition. Be brief, and to the point.

(a) [3 pts] object

Solution: A compound data type that is often used to model a thing or concept in the real world. An object is instantiated from a class (definition/template). It bundles together the data and the operations that are relevant for that kind of data. Instance of a class and objects are used interchangeably. You can create multiple objects from the same class definition.

Looking for: Compound Data Type, Includes data & operations/functions/methods, created by instantiating a class definition.

(b) [3 pts] class

Solution: A user-defined compound type. A collection of cooperating methods. A class is a template for the objects that are instances of it. (A class is used to instantiate objects.)

Looking for: Compound Data Type , can specify functions and group data, used to instantiate objects.

(c) [3 pts] instantiate

Solution: To create an instance of a class, and to run its initializer.

2. *(8 points)*
For each of the following multiple choice questions, indicate the single most correct answer by circling it!

(a) [1 pt] An error in a program that makes it impossible to parse – and therefore impossible to interpret, is a:

A. Syntax Error B. Semantic Error C. Runtime Error D. Parse Error

(b) [1 pt] An error (in code) that leads to unexpected behavior. The program functions correctly (does what the code says) but the code does not actually perform the action that the programmer intended, is a:

A. Syntax Error **B. Semantic Error** C. Runtime Error D. Unexpected Error

(c) [1 pt] An error raised by the python interpreter while the program is executing if something goes wrong. For example, a divide by zero error, is a:

A. Syntax Error B. Semantic Error **C. Runtime Error** D. Interpreter Error

(d) [1 pt] Which of the following is/are valid functions to use with writing to a CSV file using a CSV Writer?

A. write
B. writeline
C. writerow
D. writelines
E. writerows
F. A and C are valid choices.
G. B and D are valid choices.
H. C and E are valid choices.

(e) [1 pt] What is the name of the option used in the constructor to associate a radiobutton with a Tkinter variable in order to keep track of which radiobutton is pressed?

A. `value` B. `StringVar` **C.** `variable` D. None of these

(f) [1 pt] When an entry box has state set to DISABLED, the user cannot alter the text it contains, but the program can by using the `insert` method.

A. True **B. False**

(g) [1 pt] What is pointed at by the `myList` and `sortedList` variables after the following code is executed?

```
myList = [128,100,111,31]
sortedList = myList.sort()
```

A. myList = [128,100,111,31] and sortedList = [31,100,111,128]
B. Both myList and sortedList = [31,100,111,128]
C. myList = None and sortedList = [31,100,111,128]
D. sortedList = None and myList = [31,100,111,128]
E. sortedList = [0,1,2,3] and myList = [128,100,111,31]
F. myList = [128,100,111,31] and an exception occurs, preventing sortedList from being assigned.

(h) [1 pt] Values stored in dictionaries can be which of the following?

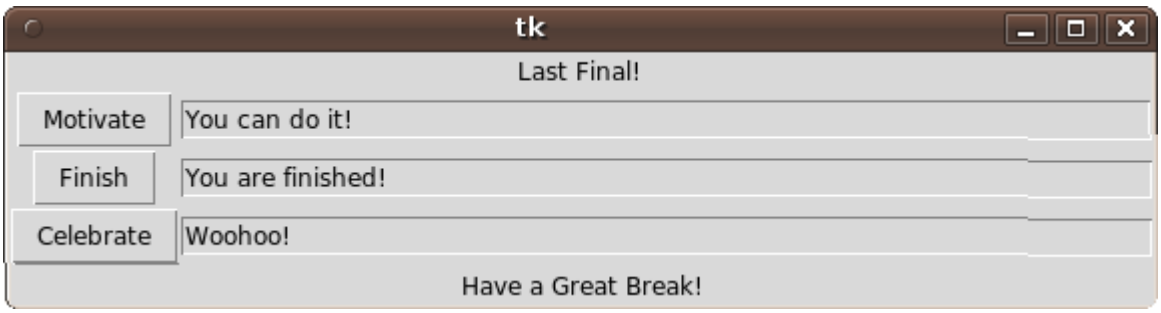
A. Any data type
B. Mutable data types
C. Immutable data types
D. Strings, ints and lists only
E. None of the above

3. *(15 points)*
Given the following code, draw the GUI that is produced after each of the buttons is clicked. (Your single drawing should incorporate any changes that are made after all of the buttons are clicked once).

```
from tkinter import *
class FinalGUI:
  def __init__(self,root):
    self.frame = Frame(root)
    self.lastOne = Label(root,text = "Last Final!")
    self.summer = Label(root, text = "Have a Great Break!")
    self.lastOne.pack()
    self.frame.pack()
    self.motivate = Button(self.frame, text = "Motivate",
                           command = self.motivation)
    self.finish = Button(self.frame, text = "Finish", command = self.finished)
    self.celebrate = Button(self.frame,text = "Celebrate",
                            command = self.celebration)
    self.m = StringVar()
    self.f = StringVar()
    self.c = StringVar()
    self.motivateE = Entry(self.frame, width = 60, textvariable = self.m)
    self.finishE = Entry(self.frame, width = 60, textvariable = self.f)
    self.celebrateE = Entry(self.frame, width = 60, textvariable = self.c)
    self.motivate.grid(row = 0, column = 0)
    self.motivateE.grid(row = 0, column = 1)
    self.finish.grid(row = 1, column = 0)
    self.finishE.grid(row = 1, column = 1)
    self.celebrate.grid(row = 2, column = 0)
    self.celebrateE.grid(row = 2, column = 1)
  def motivation(self):
    self.m.set("You can do it!")
    self.motivateE.config(state = "readonly")
  def finished(self):
    self.f.set("You are finished!")
    self.finishE.config(state = "readonly")
  def celebration(self):
    self.c.set("Woohoo!")
    self.celebrateE.config(state = "readonly")
    self.summer.pack()

root = Tk()
final = FinalGUI(root)
root.mainloop()
```

Solution:



Grading:

```
+1 for the frame with decorations
+1 for the Last Final! label
+3 for the buttons (1 per button)
+6 for the entries (2 each- 1 for the text, 1 for correct read-only formatting)
+2 for the Have a Great Break! Label
+2 for correct placement of all widgets
```

4. *(14 points)*

Write a function named **phoneBill** that takes in a list of tuples. Each tuple will consist of a name, the number of text messages, and the number of minutes that a particular person has used.

Your function should return a list of tuples that consist of a name and the dolar amount of that person's phone bill as a floating point number.

All users are on the same plan which costs $25 a month, and includes 500 texts and 300 minutes. If users go over their limits, they are charged $0.10 for each additional text message (over 500). For minutes 301-400 (the first 100 minutes over) users are charged $0.10 a minutes. For minutes 401- and up they are charged $0.25 a minute.

Example test case:

```
>>> result =  phoneBill( [ ('Bryan',500,300), ('Jay',501,200), ('Will',0,301),
      ('Jarrod',510,310), ('Nolan',100,410),('Alex',550,450) ]  )
>>> result
[ ('Bryan', 25.0), ('Jay', 25.1), ('Will', 25.1),
      ('Jarrod', 27.0), ('Nolan', 37.5), ('Alex', 52.5) ]
```

Solution:

```
def phoneBill(userData):
    billData=[]
    for user in userData:
        name=user[0]
        texts=user[1]
        mins=user[2]
        bill = 25.0
        if texts > 500:
           bill = bill + 0.1 * (texts-500)

        if mins > 400:
           bill = bill + (0.1 * 100) + (mins-400) * 0.25
        elif mins > 300:
           bill = bill + 0.1 * (mins-300)

        billData.append( (name,bill) )
    return billData
```

Grading: 1 point for a correct header.
3 points for extracting the name/texts/minutes numbers correctly.

1 point for charging everybody the base $25
2 points for charging 0.10 for all text messages ABOVE 500
2 points for charging 0.10 for all minutes in the 301-400 range.
2 points for charging 0.25 for all minutes in the 401-max range.
1 point for creating a tuple with (name,bill).
1 point for appending to a list.
1 point for returning the list of tuples.