Name : _____

Grading TA: _____

- 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 6 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
1. Vocabulary	9	
2. Multiple Choice	10	
3. Short Answer	10	
4. SmallestOf3	6	
5. findFloats	10	
6. Rank Days	10	
Total:	55	

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] argument

   Solution: argument - A value provided to a function when the function is called. This value is assigned to the corresponding parameter in the function.

   (b) [3 pts] parameter

   Solution: parameter - A name used inside a function to refer to the value passed as an argument.

   (c) [3 pts] dictionary

   Solution: A mutable compound data type that associates keys with values. They map keys, which can be any immutable type, to values, which can be any type, just like the values of a list or tuple.

2. *(10 points)*
   For each of the following multiple choice questions, indicate the best answer by circling it.

   (a) [1 pt] Which variable name(s) below is/are invalid in Python?

   A. `1st_num`
   B. `num_1st`
   C. `num-1st`
   D. `firstNum`
   E. A, B, and C
   F. A and C

G. A,B,C and D

(b) [1 pt] Which of the following data types is/are immutable?

A. Lists **B. Tuples** C. Dictonaries D. All of the above E. None of the above

(c) [1 pt] What is printed by the following lines of code?

```
myWords = ["What", "Is", "Your", "Name"]
newWords = myWords[0] + myWords[-1]
print(newWords)
```

A. `['What', 'Your']`
B. `['What', 'Name']`
C. `WhatName`
D. `'WhatName'`
E. `WhatYour`
F. `'WhatYour'`
G. `IndexError: list index out of range`

(d) [1 pt] Which of the following code fragments correctly prints the sentence contained in the list below WITHOUT commas or brackets on a single line?

```
aList = ['GO', "Yellowjackets", '!']
```

A.
```
print(str(aList))
```
B.
```
aStr = ""
for word in aList:
   aStr = aStr + word
print(aStr)
```
C.
```
for word in aList:
    print(word)
```
D.
```
print(aList[:])
```

(e) [1 pt] Examine the following code segments. Which of the following will successfully convert "CS 2315" into "CS 2316"?

```
orig = "CS 2315"
```

A. `new = orig[0:len(orig)-1] + "6"`
B. `new = orig[0:len(orig)] + "6"`
C.
```
new = orig
new[6] = "6"
```
D. A and C
E. B and C
F. A, B and C

(f) [1 pt] Which of these would you use to format a number to three places after the decimal places?

A. `"{}".format(3.14159)` B. `"{.3}".format(3.14159)` C. `"{.3f}".format(3.14159)`
D. `"{:.3f}".format(3.1459)` E. `"{1:.3f}".format(3.14159)`

(g) [1 pt] Given the following code, what is the data type of the value stored in the "myAns" variable?

```
def factorial(myInt):
   fact=1
   for i in range(myInt):
      fact = fact * (i+1)
   print(fact)

myAns = factorial(4)
```

A. Generator B. Range C. Bool D. Int E. Float **F. NoneType**

(h) [1 pt] Pretend you are the python interpreter and the following code is executed. What happens?

```
myList = [0,10,-2]
for x in range(10,15):
   if x % 3 == 0:
      print("Yes")
   if myList[x//10] == 10:
      print("Second")
```

A. An exception is generated

B. "Second" is never printed.

C. "Second" is printed exactly twice.

D. "Second" is printed exactly four times.

E. "Second" is printed exactly five times.

F. "Yes" is printed exactly twice.

(i) [1 pt] Which of the following is true about the keys in a dictionary?

A. An integer can be a key.

B. A list can be a key.

C. A dictionary can be a key.

D. All keys must be mutable.

E. All of the above are False.

(j) [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. *(10 points)*

For each of the following questions, give a brief answer:

(a) [2 pts] What is the type of `x` in the following line of code? `x = input("enter a number:")`

Solution:
str or String
Grading: +2 if correct.

(b) [2 pts] What is a boolean expression?

Solution: "An expression that evalutes to a boolean value"
Grading: +2 if correct. +1 if they say something about true/false but don't mention expressions or boolean conditionals (`<`, `<=`, `>`, `>=`, `==`, `!=`), or if they mention expressions or boolean conditionals but don't mention true/false.

(c) [3 pts] List the three logical operators (NOT comparison operators) in Python. Then, give an example expression for each that evaluates to False.

Solution:
and, or, not
FALSE and TRUE (or TRUE and FALSE)
FALSE or FALSE
not TRUE
Grading: +1 for each operator AND correct example. (we will accept much more complicated examples, or a combined example.)

(d) [3 pts] What is the difference between printing a value in a function and returning that value?

Solution:
Sample answer: Printing displays the value you printed to the shell. This value is not able to be saved in any way. Returning will give the value being returned back to whatever line of code called the function, allowing this value to be used in another expression.
Grading: +1 if their answer demonstrates understanding.
+1 for identifying that printing displays to the user/shell.
+1 for identifying that returning returns a value that can be used (in an expression, assigned to a variable, etc...)

4. *(6 points)*

Write a function called **smallestOfThree** that accepts three integer parameters. The function will return the smallest of the three parameters. If several of the paramters are the same (and the smallest) it may return either of the same (smallest) parameters.

Example test cases:

```
>>>smallestOfThree(1,5,10)
1
>>>smallestOfThree(5,5,5)
5
>>>smallestOfThree(5,5,1)
1
```

Solution:

```
def smallestOf3(a,b,c):
   smallest = a
   if b <= smallest:
      smallest = b
   if c <= smallest:
      smallest = c
   return smallest
```

Grading: 1 points for a correct header.
1 points for returning (instead of printing)
2 points for working in the case where all three numbers are different (1,5,4)
2 points for also working when some of the numbers are the same (e.g. 5,5,1), which typically means they have to check that using `<=` instead of just `<`, unless the `<` check doesn't lead to an error when items are the same... try with 5,5,1 for example!.

5. *(10 points)*

Write a function named `findFloats` that takes in list of data, and returns a new list containing only the floats. Your function should go through each item in the original list and check to see if it is a float. If it is a floating point number, you should copy it to a new list. After you have done this for all items in the original list, return the new list.

Example run:

```
>>> result = findFloats( [10.0, 5, True, 'Testing', 11.2] )
>>> print( result )
[10.0, 11.2]
>>>
```

Solution:

```
def findFloats( aList ):
   newList = []
   for item in aList:
      if type(item) == float:
         newList.append(item)

   return newList
```

Grading:
+1: Correct function header
+1: Makes an empty list
+2: Iterates through the contents of the list
+2 check for float type correctly.
+2: correctly appends item to the list.
+1: Does not append non-floats to the list.
+1: returns the new list.

6. *(10 points)*

Write a function named `rankDays` which takes in one parameter, aList. The list will be a list of tuples, with each tuple containing information in the format (Number Processed, Day of Week). The Day of Week will be a string (either "Monday", "Tuesday", "Wednesday", "Thursday", or "Friday"), and the Number Processed will be an integer. You should create and return a dictionary which has the day's "rank" (a number 1-5, with 1 corresponding to the day with the largest number of items processed) as a key and the corresponding day of the week as the value. The ranking criteria will be on the numbers of items processed each day. You may assume that no two days will share the same number of items processed (i.e. each day will have a uniquely different number of items processed).

Example test case:

```
>>> aList = [(456, "Monday"), (371, "Tuesday"), (2316, "Wednesday"),
(1371, "Thursday"), (17, "Friday")]
>>> myDict = rankDays(aList)
>>> print(myDict)
{1: 'Wednesday', 2: 'Thursday', 3: 'Monday', 4: 'Tuesday', 5: 'Friday'}
```

Solution:

```
def rankDays(aList):
   myDict = {}
   aList.sort(reverse = True)
   rank = 1
   for item in aList:
      myDict[rank] = item[1]
      rank = rank+1
   return myDict
```

Grading: +1 correct function header
+1 uses dictionary to store values
+2 the keys in the dictionary are the numbers 1..5
+3 correctly ranks days based upon number of items processed.
(-1 if sort order is backwards)
+2 the value of each key is the correct day in the week.
+1 returns a dictionary
-2 if assigning none to a critical variable (e.g. alist=alist.sort())