CS 1803 Spring 2011
February 14, 2011
Exam 1

Name: _____
Section:_____
Grading TA:_____

Question	Possible Points	Earned Points	Graded By
Section and TA Information	5		
Multiple Choice	20		
Code Reading #11	15		
Code Reading #12	14		
Code Writing cumSum	10		
Code Writing uniqueChars	16		
Total	80		

Multiple Choice- 20 points

Write your answers in the spaces below

1._____ 2._____ 3._____ 4._____ 5._____

6._____ 7._____ 8._____ 9._____ 10._____

1. An empty print statement print()
   A. Returns a newline character ('\n')
   B. Prints a newline character ('\n')
   C. Both A and B
   D. None of the above

2. Which index is out of range when fruits=['banana', 'apple']?
   A. fruits[2]
   B. fruits[-1]
   C. fruits[-2]
   D. all of the above
   E. both A and C
   F. both B and C

3. Which of the following are valid method headers?
   A. def myFunc()
   B. def myFunc(a,b):
   C. def myFunc(a = 7)
   D. myFunc():
   E. both A and B
   F. both B and C

4. myList = [0,1,2,3,4,5]
   Which of the following lines of code will create a new list that has the elements of myList reversed?
   A. myList.reverse()
   B. r = myList[::-1]
   C. r = myList[-1:0:-1]
   D. r = myList.flip()

5. For some number x, the following code will make at *most* ___Q___ function call(s) and at *least* R__ function call(s).

```
if x > 20:
    some_func()
if x > 10:
    some_func()
elif x > 7:
    other_func()
elif x > 5:
    another_func()
```

```
else:
   pass
```

A. Q = 2, R = 0
B. Q = 3 , R = 0
C. Q = 4, R = 1
D. Q = 3, R = 1

6. Which of the following is not true about dictionaries?
   A.Keys must be immutable
   B.All entries are key:value pairs
   C.Specific entries are accessed by index position
   D. Defined with {}

7. a = return print("CS IS FUN")
   What is the value of a?
   A. there is no value because this code produces an error
   B."CS IS FUN"
   C. print("CS IS FUN")
   D. None

8. Which of the following are NOT true about conditional statements?
   A. You can have unlimited elif per if
   B. You can only have 1 else per if
   C. Each If must have an elif or else
   D. elif statements must specify a boolean expression

9. Given the following lines of code which statements are true?
   ```
   for var in [4,12,28,9]:
       print(var)
   ```
   A.In the iterations var will equal 1,2,3,4 – in that order
   B. In the iterations var will equal 4,12,28,9 – in that order
   C.This loop prints 5 times
   D. The same result/set of print lines could not be obtained with a while loop

10. To write a while loop that is not infinite you must:
    A. specify a terminating condition
    B. have a colon at the end of the while statement line
    C. move towards the terminating condition in the loop
    D. know how many iterations will be executed
    E. A,B, and C
    F. All of the above

Code Reading - 29 points

11. (15 points) After the following lines are executed, what is the value stored in each of the

variables?
A =[“CS”, “1803”, “is”, “so”, “fun!”]
B = A
C = B[:]
B[3] = A[4]
A = B[0:5:2]
D = C[0] + 3*C[1]
E = D
D = C[-1]

A =

B =

C =

D =

E =

Correct answers:
A ▪ ['CS', 'is', 'fun!']
B ▪ ['CS', '1803', 'is', 'fun!', 'fun!']
C ▪ ['CS', '1803', 'is', 'so', 'fun!']
D ▪ 'fun!'
E ▪ 'CS180318031803'
+3 for each correct answer

12. (14 points) Your friend is writing a program to see if one string can be created by rearranging the characters in another string (regardless of capitalization), but it's not quite working.

```
def is_anagram(word1, word2):
    # check that lengths are the same
    if len(word1) != len(word2):
        return False
    # check that everything in word1 is in word2
    for c in word1:
        if c not in word2:
            return False
    #check that everything in word2 is in word1
    for c in word2:
        if c not in word1:
            return False
    return True
```

Provide the following to help your friend test the function:
(if such a pair does not exist, write “does not exist”)
*note: A pair of anagrams is a pair of strings that have all of the same letters but not necessarily

in the same order.

A pair of anagrams for which the function would return True
word1 = APPLE
word2 = PLEAP
A pair of anagrams for which the function would return False (a false negative result)
word1 = cs1803
word2 = 1803CS
*anything with different capitalizations

Why does the code return that they are not anagrams when they are?
The code is case sensitive so anything with different capitlaizations will be negative

A pair of non-anagram strings for which the function would return True (a false positive result)
word1 = AAAAAB
word2 = BBBBBA

Why does the code return that they are anagrams when they are not?
There are the same characters but not the same number of each one, and the code only checks to see if the same characters are used regardless of the number of times

A pair of non-anagram strings for which the function would return False
word1 = BOBBY
word2 = BOB

grading:
+2 each for true returns that should be true and false that should be false
+3 each for false positive and false negative inputs
+2 each for explanation of false positive and negative

Code Writing (26 points)

13. Cumulative Sum (10 points)
Write a function named cumSum that takes in one parameter, a list of floating point numbers. Your objective is to make a new list of the same length that has the cumulative sums from the input list, meaning that the value at index 3 is the sum of the values in positions 0,1, 2 and 3. Be sure not to alter the data in the original list, and return the new list of cumulative sums.

ex: cumSum([0,10,23,-9,3]) returns [0,10,33,24,27]

Sample Solution:

```
def cumSum(inList):
    outList = inList[:]
    for index in range(1,len(outList)):
        outList[index] = outList[index] + outList[index-1]
    return outList
```

+2 header and parameter
+2 make a new list that has the same length as the input
+2 loop through the list
+1 no indexing errors
+2 generate correct cumulative sums
+1 return the new list

14. Remove Duplicates (16 points)
    Write a function named uniqueChars that takes in a string as a parameter. The purpose of the function is to create a list of all the unique characters in the string in the order in which they appear in the string parameter. For this question, an upper case character is the same as the lower case character (so if you had the string "Abbbbba Baaaa", the resulting list would be ['A', 'b',' '], not ['A', 'b', 'a', 'B', ' ']); you must preserve the case of the first instance of the character in your list. Once you have finished retrieving all the unique characters from the string, you should return the list of unique characters.
    Ex: uniqueChars("Hello World") returns ['H', 'e', 'l', 'o', ' ', 'W', 'r', 'd'].

SAMPLE SOLUTION:

```
def uniqueChars(string):
    chars = []
    for character in string:
        if not ((character.upper() in chars) or (character.lower() in chars)):
            chars.append(character)
    return chars
```

GRADING RUBRIC:
+2 for correct function definition
+2 for declaring a list (somewhere)
+2 for declaring the list outside of the loop
+4 for checking to see whether or not the character exists in the list already before attempting to adding
+2 if case is successfully ignored
+3 for adding the character to the list
+1 for returning the list of characters
-2 for trying to .lower a list
-1 for not preserving capitalization when adding to result list
-2 for errors in looping.