CS 1110 Prelim 1 Grading Guide (3/10/2015)

Name: ______________________ NetID: ______________

Problem 1	15 points	
Problem 2	20 points	
Problem 3	20 points	
Problem 4	15 points	
Problem 5	20 points	
Problem 6	10 points	

Name: ____________________ NetID: ____________

1 String Manipulation

(a) Implement the following function so that it performs as specified.

```
def Q1(s):
    """ Returns True if the characters at the start and end of s are the
    same and occur nowhere else in s

    PreCondition: s is a string with length greater than or equal to 3.
    """
```

```
##################################################################

Typical Solutions

       n = len(s)
       t = s[1:n-1]
       return s[0]==s[n-1] and t.count(s[0])==0

       return s[0]==s[-1] and s.count(s[0])==2


Scoring = 8 pts Total


     3 pts for checking that the first and last character are equal
     3 pts for the "other part" of the and
     1 pt for correctly combining the two subexpressions using "and"
     1 pt for remembering to "return" the value

    Deductions:
         -1 if they end return a non-boolean value
         -1 for index off-by-one issues
```

(b) Imagine playing around with this script:

```
s = raw_input('Enter a string that has length greater than or equal to 2: ')
t = s.replace(s[0],'x')
u = t.replace('x',s[0])
print s,u
```

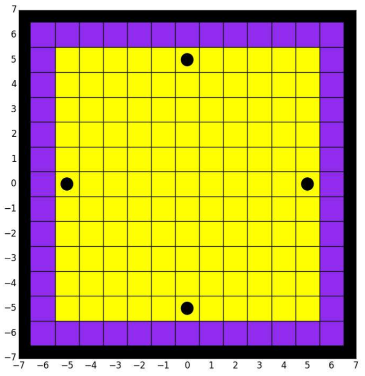
Sometimes it is the case that the printed values of `s` and `u` are the same and sometimes it is observed that they are different. Give a Boolean expression that is `True` if `u` and `s` have the same value and is `False` otherwise. Hint. Consider some small examples.

```
#####################################################
Scoring: 7pts Total

   Solution:

           s[0]=='x' or s.count('x') ==0

3 pts for counting that the x occurrences are equal to zero
  award 2 points  for any example that confirms this provided it shows t and u,
  e.g.,   s = 'abca'  t = 'xbcx'  u = 'acda'

2 pts for checking that the zeroth character in s is equal to 'x'
  award 1 point for an example, e.g., s = 'xabx', t = 'xabx', u = 'xabx'


2 pts for combining the subexpressions using "or"
  award 1 point if chit chat has an "or"
```

Name: ______________________________ NetID: ______________

2 Random Walk

Consider the random walk simulation in Assignment 4. Recall that the simulation produces a travel string comprised of the characters `N`, `S`, `E`, and `W`. The travel string encodes the hop directions associated with the robots journey from (0,0) to a purple boundary tile. Here is a display of an $n = 5$ playpen highlighting its four *middle edge* tiles:



(a) Assume that `x` and `y` are initialized with the (x, y) coordinates of the robot's location and that the value of `n` is the size of the playpen. Give a Boolean expression that is `True` if the robot is on a middle edge tile and `False` otherwise.

```
################################

Scoring:

Solutions

    (abs(x)==n and  y==0) or (abs(y)==n and abs(x)==0)


    (x==n and y==0) or (x==-n and y==0) or (x==0 and y==n) or (x==0 and y==-n)

    1 point for each "and"  giving 4 total.  If those 4 ands" are "ors", then just 2 points
    2 points  for gluing them together with or's. If those three "ors are ands", award 1 point
```

Name: ______________________________ NetID: ______________

(b) A hop is "predictable" if it is in the same direction as the previous hop. Here is a travel string that includes 3 predictable hops: `'EWNNNWWN'`. Complete the following function so that it performs as specified.

```
    def nPredictable(s):
       """ Returns an int that is the number of predictable hops in s.
       Precondition: s is a travel string.
       """
```

```
#############################################

Scoring: 6 pts total

Solution:


    count = 0
    for k in range(1,len(s)):
         if s[k] == s[k-1]:
             count = count + 1
    return count


1 pt for initializing a count variable
1 pt for using the correct range in the for loop
2 pts for checking the current location against the previous (it is also okay to do the reverse check p
  everything else was modified accordingly)
1 pt for incrementing the counter every time a repetition is observed
1 pt for remembering to "return" the final value


4 points for something like this (which doesnt work if you have a run like 'NNN'):

  return s.count('NN") + s.count('EE') + s.count('SS') + s.count('WW')
```

Name: ______________________________ NetID: ________________

(c) Assume that

- s is initialized and contains a travel string.
- xf is initialized and contains the x-coordinate of the robot's final location.
- yf is initialized and contains the y-coordinate of the robot's final location.

Write Python code that assigns to variables x and y the x-coordinate and y-coordinate of the last yellow tile on the robot's journey. Points will be deducted for solutions that involve a loop.

```
Scoring:  8 pts

Solution:

     m = len(s)-1
     if s[m] == 'E'
         x = xf-1
     elif s[m] = 'W':
         x = xf+1
     elif s[m] = 'N':
         y = yf-1
     elif s[m]=='S'
         y = yf+1

1 pt  for finding the index of the last position (len(s)-1)
2 pts  for the four comparisons to figure out whether the character at the last position is
       {'N','S','W','E'}
2 pts  for applying the correct offset to the appropriate value {xf,yf} and assigning it to
       the variables x and y
1 pt   for giving some inline comments #this does something
2 pts  for using "if / elif"  statement combination

Deductions:

-1 pt   if they decide to just use "if" statements for each separate case
-4 pts  if they end up using loops in the solution
-2 pt   for unnecessary computation (e.g. recomputing the length or scanning
        along the length of the string etc)
```

Name: ______________________________ NetID: ____________________

3 Short Answer

(a) Assign a value to `x` so that the character `'A'` is printed out:

```
    x =  _______________________

    if x%2==0 and x%3==1:
       print 'A'


#####################
Scoring 2pts  total

   1 if x is even
   1 if x%3==1
```

(b) Assign values to `x` and `y` so that the character `'D'` is printed out:

```
   x = ____________________


   y = ____________________

   if  not ((0<=x<=3) and (0<=y<=3)):
      print 'A'
   elif y<=1 or y>=2:
      print 'B'
   elif x<=1 or x>=2:
      print 'C'
   else:
      print 'D'

 ###########################

Scoring:
 4pts

 Solution:  must have 1<x<2   and 1<y<2

 Don't worry about < vs <= in this problem

 2 points if there x and y satisy 0<x<3  and 0<y<3

 1 point in addidtion if their x satisfies  1<x<2

 1 point in addition if their y satisfie  1<y<2
```

Name: ______________________________ NetID: ________________

(c) What would be the output if the following code is executed?

```
    x = float(10/4)
    print x
```

```
###########################
Scoring 2 pts total

Full credit for 2 or 2.0

1 point for any  chit chat about types

2 pts for correct division result i.e. has integer
```

(d) Suppose the functions in modules `M1.py` and `M2.py` are to be used by module `M.py`. Briefly explain why it is safer to implement `M.py` with

```
     import M1
     import M2
```

than with

```
     from M1 import *
     from M2 import *
```

```
#####################################
Scoring

4 pts Any thing that basically says "with the dot notation, there is no ambiguity
with respect to function names. "

2 points for chit chat with things like M1.f and M2.f
```

Name: ______________________________ NetID: ________________

(e) Indicate what the output would be if the following application script is run:

```
def F(x,y):
    x = y
    y = x
    z = x+2*y
    print x,y,z
    return z

if __name__ == '__main__':
    x = 1
    y = 2
    print x,y
    x = F(y,x)
    print x,y
    if x<y:
        print 'A'
    else:
        print 'B'
```

```
###########################
Scoring:  8 point total

Solution:

1,2
1,2,5
5,2
B

2 pt for the 1st print output
2 pt for the 2nd line
2 pts for the 3rd line of print output from the function
2 pt for printing the final char 'B'


-1 pt if they mutate the x,y values after returning from the
   function (i.e.) pass by reference
-2 pts for mixing up the x,y value order during the function call
-2 pts if  assignment statements print to the terminal
-1 pt for printing out 'A'  in the x<y comparison
```

Name: ______________________________ NetID: ________________

4 Testing

In the "You Are Late" problem in assignment A1 you essentially implemented the following function:

```
def YouAreLate(s):
   """ Returns a time string that specifies a time that is
   90 minutes earlier than the time specified by s.

   A time string has these properties:

   (a) its length is 5
   (b) its middle character is ':'
   (c) its first two characters are '01', '02',...,'09','10', '11',  or '12'
   (d) its last two characters are '00','01',...,'58', or '59'

   Precondition: s is a time string
   """
```

In assignment A2 you assembled 10 "representative" test cases that could be used to help affirm the correctness of your `WindChill` implementation. This problem is about applying the representative test idea to the function `YouAreLate`. In particular, complete the following table so that it specifies a collection of six representative test cases that could be used to check the correctness of a `YouAreLate` implementation.

Test Case	Expected Output	What it Checks
01:15	11:45	

Points will be deducted for test cases that do not contribute any new information about correctness when compared to the other test cases. Test cases that check for valid input will not add to your score on this problem. To enforce brevity, we will not consider anything written outside of the table.

Name: ________________________ NetID: ______________

```
Starting from 15 points, -2 for each one of these that is NOT covered

   M1  appointment  minute is one digit
   M2  appointment minute is two digits

   H1  appointment hour is one digit
   H2  appointment hour is two digits

   D1  appointment hour is two less than arrival hour
   D2  appointment hour is 1 less that arrival hour

   D3  appointment hour is on the "other side" of 1 oclock

and -1 for each test that is "covered" by the other 5 tests
```

Name: ______________________ NetID: ______________

5 Loops

(a) Consider the following script

```
t = 'x'
s = raw_input('Enter a string: ')
for c in s:
    t =  t + c + t
```

Assuming that `'ba'` is assigned to `s`, what is the final value of `t`? Show work.

```
###################################################
Scoring   10 pts

    t --> 'x
    t --> 'xbx'
    t --> 'xbxaxbx'




2 pts for saying what t is before the loop starts
3 pts for 1st step 'xbx' first iteration (show concatenation pieces)
3 pts for second step 'xbxaxbx' second iteration (show concatenation pieces)
2 pts for the correct final answer

-5 pts for a correct final solution without showing work
-2 pts for growing the string only on one side
-2 pts for reversing the 'ba' character order 'ab'
-2 pts for mixing up the 'ba'  string with the 'x' string
```

Name: ________________________________ NetID: ____________________

(b) Write a script that is equivalent to the script in part (a) but which uses a while-loop instead of a for-loop.

```
###################################################
Scoring   10 pts

     t = 'x'
     s = raw_input('Enter a string: ')
     k=0
     while k<len(s):
         t =  t + s[k] + t
         k = k+1



2 pts for initialization of t variable
2 pts for using the raw_input() cmd in an identical way as in (a)
1 pt for initialization of the termination condition variable (k in our soln)

2 pts for a correct while termination condition k < len(s)
2 pts for the appropriate concatenation accumulator line
1 pt for the termination condition update k+=1

No points off for missing ":"

-1 pt for off-by-one termination comparison
-1 pt for incorrect indexing into the s string

(up to -2 pts for wrong loop syntax i.e.  indentation error

etc)
```

Name: ______________________________ NetID: ______________

6 A Graphics Procedure

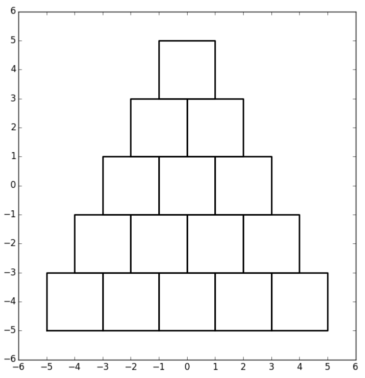
Assume the availability of the following procedure:

```
def DrawRow(x0,y0,s,n):
    """ Draws a horizontal row of n squares that are each s-by-s. The
    center of the leftmost square is (x0,y0).

    Precondition: x0, y0, and s are floats, n is a positive integer"""
```

By making effective use of `DrawRow`, implement the following procedure so that it performs as specified

```
def DrawPyramid(x0,y0,s,n):
    """ Draws a pyramid of s-by-s squares. The bottom row consists
    of n squares and the lower left corner of the leftmost square is at (x0,y0).
    There are n rows of squares and each row has one less square than the row beneath it.
    The centers of each row are vertically aligned.

    Precondition: x0, y0, and s are floats, n is a positive integer"""
```

For your information, the call `DrawPyramid(-5.,-5.,2.,5)` would produce this figure:



Name: ______________________________ NetID: ________________

```
############################################
Scoring:  10 pts total


def DrawPyramid(x,y,s,n):
    xc = x+s/2
    yc = y+s/2
    for k in range(n):
        DrawRow(xc,yc,s,n-k)
        yc = yc+s
        xc = xc+s/2



2 pts for xn, yn initialization
2 pts for correct for / while loop termination condition
2 pts for updating the row posn xn, yn inside the loop
1 pt for using a DrawRow() call
1 pt for giving DrawRow the appropriate inputs
2 pts for incrementing/decrementing n

-1 pt if loop terminates early
-2 pts for using size 1 blocks instead of s
-2 pts

if pyramid offset by s/2
-2 pts if pyramid is not centered
-2 pts if bottom left corner is not at x_0, y_0
-2 pts for confusing DrawRow() with DrawPyramid() calls / parameters
```