Last Name: _____ First Name: _____ Cornell NetID: _____

CS 1110 Prelim 2 April 26th, 2016

Circle your lab/situation:

ACCEL:	Tue 12:20	Tue 1:25	Tue 2:30	Tue 3:35		
ACCEL :	Wed 10:10	Wed 11:15	Wed 12:20	Wed 1:25	Wed 2:30	Wed 3:35
PHILLIPS :	Tue 12:20	Tue 1:25	Wed 12:20			

I'm a grad student and hence LABLESS

This 90-minute exam has 6 questions worth a total of 34 points. When permitted to begin, scan the whole test before starting. Budget your time wisely.

When asked to write Python code on this exam, you may use any Python feature that you have learned about in class.

Unless otherwise stated, you may write helper functions when asked to write code, but include specifications for them in their doc strings.

It is a violation of the Academic Integrity Code to look at any exam other than your own, to look at any other reference material, or to otherwise give or receive unauthorized help. We also ask that you not discuss this exam with students who are scheduled to take a later makeup.

Academic Integrity is expected of all students of Cornell University at all times, whether in the presence or absence of members of the faculty. Understanding this, I declare I shall not give, use or receive unauthorized aid in this examination.

Signature: _____ Date _____

For reference:

`s[i:j]`	Returns: A new string `s[i] s[i+1] ... s[j-1]` under ordinary circumstances. Returns `''` if `i` $\geq$ `len(s)` or `i` $\geq$ `j`.
`s.find(s1)`	Returns: index of the first character of the first occurrence of `s1` in `s`, or -1 if `s1` does not occur in `s`.
`s.index(s1)`	Like find, but raises an error if `s1` is not found.
`s.lower()`	Returns: a copy of `s` with all letters in it converted to lowercase.
`s.count(s1)`	Returns: the number of non-overlapping appearances of string `s1` in string `s`.
`lt[i:j]`	Returns: A new list`[lt[i], lt[i+1], ..., lt[j-1]]` under ordinary circumstances. Returns `[]` if `i` $\geq$ `len(lt)` or `i` $\geq$ `j`.
`lt.index(item)`	Returns: index of first occurrence of `item` in list `lt`; raises an error if `item` is not found.
`range(n)`	Returns: the list `[0, 1, 2, ..., n-1]`
`x in lt`	Returns: `True` if `x` is in list or string `lt`, `False` otherwise.
`lt.append(x)`	Append object `x` to the end of list `lt`.
`lt.sort()`	Sort the items of `lt`, in place (the list is altered).

Last Name: _____ First Name: _____ Cornell NetID: _____

Question	Points	Score
1	6	
2	4	
3	11	
4	6	
5	6	
6	1	
Total:	34	

Last Name: _____ First Name: _____ Cornell NetID: _____

1. [6 points] What is the output if the following module is run:

```
def F1(x):
    y = [] # empty list
    n = len(x)
    for k in range(n):
        if k==0 or k==n-1:
            y.append(x[k])
        else:
            y.append(x[k-1]+x[k+1])
    return y

def F2(x):
    y = x
    n = len(x)
    for k in range(n):
        if k==0 or k==n-1:
            y[k] = x[k]
        else:
            y[k] = x[k-1]+x[k+1]


if __name__ == '__main__':
    x = ["A","B","C","D"]
    z = F1(x)
    print x
    print z

    print "..."

    x = ["A","B","C","D"]
    z = F2(x)
    print x
    print z
```

2. [4 points] What is the output if the following module is run?

```
class C(object):
    def __init__(self,u,v):
        self.x = u
        self.y = v

    def Reflect1(self):
        temp = self.x
        self.x = self.y
        self.y = temp

    def Reflect2(self):
        temp = self.x
        self.x = self.y
        self.y = temp
        z = self
        return z

if __name__ == '__main__':
    x = 1
    y = 2
    P = C(1,2)
    P.Reflect1()
    print x,y
    print P.x,P.y

    x = 1
    y = 2
    P = C(1,2)
    Q = P.Reflect2()
    print P.x,P.y
    print Q.x,Q.y
```

No partial credit without a diagram that displays objects and references. You can put such a diagram in the whitespace to the right of the code above.

3. [11 points] Assume the availability of these classes:

```
class Point(object):
    """
    Attributes:
        x: float, the x-coordinate of a point
        y: float, the y-coordinate of a point
    """
    def __init__(self,x,y):
        """ Creates a point.
        PreC: x and y are floats
        """
        self.x = x
        self.y = y

class LineSeg(object):
    """
    Attribute:
         P1: endpoint [Point]
         P2: endpoint [Point]
    """
    def __init__(self,Q,R):
        self.P1 = Q
        self.P2 = R
```

(a) Write code that assigns to variable `Z` a reference to a `LineSeg` object that represents the line segment with endpoints (1,2) and (3,4).

(b) We say that a point (a, b) is positive if $a > 0$ and $b > 0$. Write a boolean-valued *method* `Pos` for the `Point` class that returns `True` if the referenced point is positive. Your answer must include the header definition ("`def Pos(`..."). Omit the docstring.

(c) Complete the following function:

```
def F(L):
   """Returns a list of all those references in L that are to LineSeg objects
    whose endpoints are both positive.

    PreC: L is a list of references to LineSeg objects.
   """
```

You may assume the availability of the method `Pos` in part (b).

(d) Complete the following function. Nested loops are allowed.

```
def F(LofP1,LofP2):
   """Returns a list of references to LineSeg objects that encodes all possible
   line segments obtained by connecting a point in LofP1 to a point in LofP2

    PreC: LofP1 and LofP2 are nonempty lists of references to Point objects. Assume
    that the points encoded by these lists are all distinct.
   """
```

4. [6 points] Assume the availability of the following function:

```
def randiList(L,R,n):
    """ Returns a length-n list of random integers from interval [L,R] inclusive.

    PreC: L,R,n ints with L<=R and n>=1
    """
```

Complete the following script so that it assigns to a float variable `p` an estimate of the probability that when you roll three dice, exactly two of them have the same value. Your solution must make effective use of the three generated lists of random integers created in the assignment to `D` below. No additional calls to `randiList` or `randi` are allowed. Don't print or return `p`.

If you choose to use a helper function, put its definition here, *before* your Application script code:

```
if __name__ == '__main__':
    """ Roll 3 dice many times and record the outcome using lists.
    """
    N = 1000000
    D = [randiList(1,6,N), randiList(1,6,N), randiList(1,6,N)]
    # Each of the three lists in D is the record of the N rolls of one of
    # the three dice.

    # Hint: assign to a variable m the number of times that EXACTLY two
    # of the dice have the same value.
```

5. [6 points] These classes were involved in Assignment 6:

```
class Speech(object):
    """
    attributes:
        theSpeaker       the name of the speaker [str]
        lines            each item is a (file) line in the speech [list of str]
    """
class Scene(object):
    """
    attributes:
        actNumber       the number of the act [int]
        sceneNumber     the number of the scene [int]
        location     the location of the scene [str]
    """

class Play(object):
    """
    attributes:
        theTitle        the name of the play [str]
        theSpeeches     a list of all the speeches in the play [list(Speech)]
        theScenes       a list of all the scenes in the play [list(Scene)]
        nLines          the total number of lines in the play [int]
    """
```

Complete the following function so that it performs as specified. Hint: you don't need `stringToWordList`.

```
def qMarks(playlist):
  """ Returns the total number of question marks that occur amongst all the
  lines in all the speeches in all the plays referenced by playlist.

  PreC: playlist is a list of references to Play objects.
  """
```

6. [1 point] Write your last name, first name, and Cornell NetID at the top of *each* page, and circle your lab time on the first page.

We suggest you carefully re-read all instructions and specifications before turning this exam in.