CS 1110 Prelim 1 March 11th, 2014

This 90-minute exam has 7 questions worth a total of 55 points. When permitted to begin, scan the whole test before starting. Budget your time wisely. Use the back of the pages if you need more space. You may tear the pages apart; we have a stapler at the front of the room.

When asked to write Python code on this exam, you may use any Python feature that you have learned about in class (if-statements, for-statements, map, lists, and so on).

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:

<p>| s[i:j] | Returns: A new string s[i] s[i+1] ... s[j-1] under ordinary circumstances. Returns '' if i ≥ len(s) or i ≥ 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.spli() | Returns: a list of the words in string s, using whitespace as the delimiter. |
| s.join(slist) | Returns: a string that is the concatenation of the strings in list slist separated by string s. |
| lt[i:j] | Returns: A new list[lt[i], lt[i+1], ..., lt[j-1]] under ordinary circumstances. Returns [] if i ≥ len(lt) or i ≥ 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 lt, False otherwise. |
| lt.append(x) | Append object x to the end of list lt. |
| lt.pop(i) | Returns: item at position i in list lt, removing it from lt. If i is omitted, returns and removes the last item. |
| lt.sort() | Sort the items of lt, in place (the list is altered). |
| sum(lt) | Returns: the sum of the items in lt, which must all be numbers. |</p>
<table>
<thead>
<tr>
<th>Question</th>
<th>Points</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Total:</td>
<td>55</td>
<td></td>
</tr>
</tbody>
</table>

Page 2
1. [2 points] When allowed to begin, write your last name, first name, and Cornell NetID at the top of each page, and circle your lab time on the top of the first page of the exam.

2. [12 points] Two students were assigned to diagram the execution of the following code.

```python
1 def h(y):
2     y = x[1] - y
3     return y
4
def g(z, x):
5     x[0] = x[0] - z
6     return x
7
def f(x, y, z):
8     x = g(y, z)
9     return h(x[0])
10
x = [2,3]
y = [4,5]
x[1] = f(x, 3, y)
```

After executing the whole program they handed in the diagrams below. You are their grader; please mark up each student’s solution as follows: (1) Draw an X through anything that is present that should not be; (2) circle anything that is present and should be, but has the wrong value or name; and (3) write in anything that is missing. You may wish to do this question by first drawing the relevant frames and objects yourself.
3. [13 points] Implement `reshmuplify` and `lower_one_char` so that they meet their specifications. Your solution can call the function `find_first_vowel` below; you may assume it’s already been implemented according to its specification.

```python
def find_first_vowel(s):
    """Return: the index of the first vowel in s, or -1 if s has no vowels."""
    # Assume this function has already been implemented correctly, so you can call it.

def reshmuplify(s):
    """Return: the string s followed by a ', ' and then the same string with the substring before the first vowel replaced by 'shm'.
    Precondition:
    s contains only letters
    s contains at least one vowel.
    Examples:
    'apple' -> 'apple, shmapple'
    'banana' -> 'banana, shmanana'
    'prelim' -> 'prelim, shmelim'
    """

def lower_one_char(s, n):
    """Return: the string s with the character at index n converted to lowercase.
    Precondition:
    s is a string containing only letters, and has at least one character in it.
    n is a valid index into s.
    n > 0 and n < len(s) - 1.
    Examples:
    lower_one_char('FOO', 1) -> 'FoO'
    lower_one_char('HELLo', 2) -> 'HElLo'
    lower_one_char('HELLo', 4) -> 'HELLo'
    lower_one_char('TEST', 0) violates the precondition
    lower_one_char('TEST', 3) violates the precondition
    """
```
4. [13 points] Assume that inside a module named transcript2 is the definition of class 
Titem2, which is an extension of class Titem from Lab 3. Titem2s have the following attributes:

<table>
<thead>
<tr>
<th>attribute</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>non-empty string of lowercase letters followed by numbers</td>
</tr>
<tr>
<td>credits</td>
<td>positive int</td>
</tr>
<tr>
<td>gradeval</td>
<td>positive float</td>
</tr>
</tbody>
</table>

and are created by calls like transcript2.Titem2('CS1', 4, 'A+') (if transcript2 has been imported).

We define the GPA contribution of a Titem2 as its number of credits times its gradeval. For example, for the Titem2 created by the call above, the GPA contribution is $4 \times 4.3 = 17.2$.

Complete the function definition below so that it meets its specification. Note that we already wrote a line of code for you.

```python
def separate(sourcelist, threshold, highlist, lowlist):
    """Appends to list highlist the Titem2s from sourcelist whose GPA contribution [see definition above] is greater than or equal to threshold, and appends to list lowlist the other items in sourcelist.
    Precondition:
    sourcelist is a (possibly empty) list of Titem2s
    threshold is a float or int
    highlist and lowlist are (possibly non-empty) lists.""
    # For an example, see text at the bottom of this page.

    # **MAKE SURE YOU SEE THIS LINE, AND INDENT RELATIVE TO IT**
    for item in sourcelist:
        # Your code here...
```

Illustrative example: let id1 be the ID of a Titem2 with GPA contribution 8, and id2 be the ID of a Titem2 with GPA contribution 16. Suppose x is a 2-item list holding id1 and id2, y is an empty list, and z is the list [7]. Then, the result of the call separate(x, 10, y, z) is that:
ex remains the same;
The list that y refers to is modified to be a 1-item list containing id2;
The list that z refers to is modified to be a 2-item list containing the number 7 and id1.
5. [5 points] Assume that `separate` from the previous question is (correctly) defined in module `prelim1`. Now, suppose the following sequence of statements is executed.

```python
import transcript2
import prelim1
nextlist = [transcript2.Titem2('class1', 12, "A+"),
            transcript2.Titem2('class2', 1, "B-"),
            transcript2.Titem2('class3', 1, "B")]
high = []
low = [transcript2.Titem2('engl9999', 2, "B-"),
       transcript2.Titem2('is666', 2, "B-")]
prelim1.separate(nextlist, 12, high, low)
high[0].name = 'FAKE'
print nextlist[0].name
```

Write down what the resulting printout(s) or error(s) are. Then explain your answer in one to three sentences.

6. [5 points] Complete this function definition according to its specification. One or two lines of code suffices.

```python
def avg(inlist):
    """Returns: float value of the average of the values in list inlist.

    Pre: inlist a non-empty list, each item either an int or a float."""
    # The average of a list of numbers is the sum of the values in the list
    # divided by the length of the list.
```
7. [5 points] Consider the following function definition, which makes use of the `avg` function from
the previous question.

```python
def string_avg(nums_as_str):
    """Returns: float value of the average of the values represented
    by nums_as_str.
    """

    Examples: input ' 1. 3.5 6 ' -> output 3.5
               input '2 17.6' -> output 9.8

    Pre: nums_as_str is a string representing a non-empty sequence of numbers
         separated by whitespace.
    """
    return avg(nums_as_str.split())
```

Unfortunately, even if function `avg` is implemented correctly, `string_avg` is not correct, because
`nums_as_str.split()` is a list of strings, which `avg` is not expecting as input.

Rewrite the last line of `string_avg`, using a call to `map`, to fix this error. One to three lines of
code suffices.

---

*Did you write your name & netID on each page, circle your lab on the front, and carefully re-read
  all instructions and specifications?*