Spring 2017 CS 1110/1111 Exam 3

Bubble in your computing ID, top to bottom, in the footer of this page. We use an optical scanner to read it, so fill in the bubbles darkly. If you have a shorter ID, leave some rows blank.

In case we have trouble with the scanner, please also legibly print

Your name: ___________________________________________

Your computing ID: _______________________

Please observe the following directions throughout the exam:

- Write legibly; we deduct points if we are unsure what you wrote.
- Indentation and punctuation do matter.
- Write on the lines, where possible.
- If you need to insert a line between two you’ve written, make it clear that that is what you are doing.
- We grade one page at a time. Do not spill answers onto another page.
- Don’t add features we didn’t request: only print if we ask you to print, etc.

The exam is being given in multiple locations simultaneously, so we cannot fairly answer student questions during the exam. If you find a question ambiguous or unclear, write that down on your exam and we’ll give it due consideration during grading.

Pledge

On my honor as a student, I have neither given nor received help or assistance on this exam.

Signed: ___________________________________________
Question 1 (10 points)

When the following program reaches the line labeled “inspect variables here”, what is the value and type of each variable? Write “Error” if the variable does not exist.

```python
x, y = 3, '3'
def f(z):
    x = z + 5
    print('w =', 7)
    return z // 2
z = 7.0 * 2
a = f(x)
# inspect variables here
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>w</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Question 2 (10 points – new question)

Write a function named `middle` that accepts a single string as its argument and returns either the middle character (for an odd-length string) or the middle two characters (for an even-length string) of the string. For example, given "power" return "w" and given "glow" return "lo".

```python
# Write your function here
```

Question 3 (4 points)

Write a single expression (no more than one line of Python code) on the line below that evaluates to True if either of the following are true and False if neither is true:
- x is True
- y and z have different signs (one negative, one positive; you may assume neither is zero)
**Question 4 (6 points)**

Assume \( x = \{1:2, 3:4\}; y = [5, 6, 7, 8]\); and \( z = \text{range}(9, 19) \). Fill in the following tables.

<table>
<thead>
<tr>
<th>Expression</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 in x</td>
<td></td>
</tr>
<tr>
<td>( z[3] )</td>
<td></td>
</tr>
<tr>
<td>( \text{len}(y) - \text{len}(x) )</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Expression</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>19 in z</td>
<td></td>
</tr>
<tr>
<td>( y\text{.index}(8) )</td>
<td></td>
</tr>
<tr>
<td>( y[x[1]] )</td>
<td></td>
</tr>
</tbody>
</table>

**Question 5 (10 points)**

Write a program that prompts the user to type integers, one per line, continuing until 0 is typed. Then print a list of the numbers typed, smallest-to-largest, including the 0. You may assume the user only types integers.

Example run:

Type a number: 3
Type a number: -8
Type a number: 0
[-8, 0, 3]
Question 6 (9 points)

Give an example invocation that will cause the following function to return the int-value 3.
Note: f("2", 1.0, 3) is not a correct answer: it returns "3" not 3.

```python
def f(a, b, c):
    x = a * int(a)
    if len(x) > 3:
        try:
            return float(b)
        except:
            return c
    else:
        return str(c)
```

Answer: f( ___________ , ___________ , ___________ )

Question 7 (10 points)

Write a function stack that accepts three gamebox arguments, a, b, and ground. Remove all overlaps, moving a and b but not ground. To prevent the upper box pushing the lower box into the ground, resolve the overlaps of the lower box first (lower = larger y coordinate).

The only gamebox or pygame methods you’ll need are
- box1.touches(box2)
- box1.move_to_stop_overlapping(box2) – which you may abbreviate box1.mso(box2) for brevity
- box1.move_both_to_stop_overlapping(box2) – which you may abbreviate box1.mbso(box2)
Question 8 (11 points)

Trace through the code below, recording the lines of output, in the order they would appear if you ran the code. We have provided more blank lines than you will need.

```python
def foo(x):
    lis = [5, 6, 7]
    try:
        print(x in lis)
        print(lis[x])
    except IndexError:
        print("bad index")
    print(x)
    return 3

try:
    print(foo(4))
except:
    print("caught exception")
print("done")
```

Prints:

Question 9 (9 points)

Each row represents one regular expression; each column on string to match with that regular expression. In each cell, enter the number of matches `finditer` would return. The first row is done for you.

<table>
<thead>
<tr>
<th>Regular Expression</th>
<th>&quot;12_13&quot;</th>
<th>&quot;Hi_there!&quot;</th>
<th>&quot;[1,2,3,4]&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>[a-z]</td>
<td>0</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>[A-Z][a-z]*</td>
<td>[a-z]_]+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>([0-9]+_?)*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(^,_)[,]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Question 10 (11 points)

Give three different example strings that match the following regular expression, including at least one that is an English sentence and one that is not.

```
(Teacher|Student|TA) (of|in) [a-z]+( [A-Za-z]+)+\.
```
Question 11 (10 points)

Write a function named `save_list` that takes two arguments: a filename and some data in the form of a list of lists of integers. It should write a CSV file with that name containing those integers; for example, invoking `save_list('tmp.csv', [[2, 3], [5, 7, 11], [13]])` should create a file named `tmp.csv` that contains

```
2,3
5,7,11
13
```

You should not include any spaces within each line, but you may end each line with a comma if you wish.

```python
def save_list(filename, data):
```

```python
```