Test 1: Compsci 101/06

Owen Astrachan

February 21, 2012

Name: __________________________________________

NetID/Login: __________

Honor code acknowledgment (signature) ________________________________

<table>
<thead>
<tr>
<th></th>
<th>value</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem 1</td>
<td>32 pts.</td>
<td></td>
</tr>
<tr>
<td>Problem 2</td>
<td>14 pts.</td>
<td></td>
</tr>
<tr>
<td>Problem 3</td>
<td>12 pts.</td>
<td></td>
</tr>
<tr>
<td>Problem 4</td>
<td>18 pts.</td>
<td></td>
</tr>
<tr>
<td>TOTAL</td>
<td>76 pts.</td>
<td></td>
</tr>
</tbody>
</table>

This test has 9 pages (and there is an APT handout), be sure your test has them all. Do NOT spend too much time on one question — remember that this class lasts 75 minutes.

In writing code you do not need to worry about specifying the proper import statements. Don’t worry about getting function or method names exactly right. Assume that all libraries and packages we’ve discussed are imported in any code you write.
**PROBLEM 1 :** (Simple PI-man (32 points))

**Part A (22 points)**

Each of the variables below has a *type* and a *value*. The type is one of: list, boolean, int, string, float. For example, consider the assignment to variable `x` below:

\[ x = \text{len}([5,3,1]) \]

The type and value are shown in the first row of the table below. Fill in the other type and value entries based on the variable/expression in the first column.

<table>
<thead>
<tr>
<th>variable/expression</th>
<th>type</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>x = \text{len}([5,3,1])</code></td>
<td>int or integer</td>
<td>3</td>
</tr>
<tr>
<td><code>a = 32/13</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>h = 57 \% 12</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>e = \text{sum}(\text{range}(8))</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>d = 42 &gt; 17</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>c = &quot;do re me&quot;.\text{split}()</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>i = &quot;rabbit&quot;[-1]</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>g = &quot;dark&quot; + &quot;night&quot;</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>b = &quot;chocolate&quot;[5:8]</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>f = 0.5*10</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>k = [2,3,5,7,11,13,15][2:4]</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>j = 2**10</code></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Part B (4 points)
The volume of a cylinder, e.g., as shown below, is given by the formula $\pi \cdot r^2 \cdot h$ where $r$ is the radius of the base of the cylinder and $h$ is the height of the cylinder. For example the volume of a cylinder whose radius is 3 and whose height is 6 is $\pi \cdot 9 \cdot 6 = 169.64586$. Write the function `cylinder_volume` below so that it returns the volume of a cylinder whose radius and height are the parameters. Use `math.pi` for $\pi$. The parameters are float values.

```python
def cylinder_volume(radius, height):
```

Part C (6 points)
(There are two parts to this question.)
Consider what happens when the three lines below are executed in Python.

```python
s = "fun"
s = s + " " + s
print s, len(s)
```

Part C.1 (2 points)
The line printed is "fun fun" 7. What is returned by the function call `repeat("basket",1)` for the function `repeat` shown below. The answer alone is sufficient.

```python
def repeat(s, n):
    first = s
    for i in range(n):
        s = s + " " + s
    return s.split().count(first)
```

Part C.2 (4 points)
What value is returned by the call `repeat("fun",10)` for the function `repeat`? Explain your reasoning for full credit (answer without explanation is worth half the points).

```python
def repeat(s, n):
    first = s
    for i in range(n):
        s = s + " " + s
    return s.split().count(first)
```
PROBLEM 2:  (Talking Bout My Generation (14 points))

Part A (8 points)
Consider the Python expression and list comprehension

\[ \text{sum}([2 \times x \text{ for } x \text{ in vals}]) \]

There are many lists \text{vals} that could make the value of this expression 14, one such list with three elements is [1,1,5]. You'll be asked to find lists that generate values for expressions, any list will do. In the expression above, another three-element list that generates 14 is [4,1,2]

- Show a list \text{words} of 5 strings such that the Python expression below evaluates to 2.

\[ \text{sum}([1 \text{ for } x \text{ in words if } \text{len}(x) > 3]) \]

- Show a list \text{nums} of 6 integer values such that the Python expression below evaluates to 56.

\[ \text{max}([x \times 8 \text{ for } x \text{ in nums}]) \]

- Show a list \text{some} of 4 string values such that the Python expression below evaluates to 49.

\[ \text{min}([\text{len}(x) \times 2 \text{ for } x \text{ in some if } x \text{ starts with } \text{"a"}]) \]

- Show a list \text{primes} of 5 integer values such that the Python expression below evaluates to 30.

\[ \text{sum}([x \text{ for } x \text{ in primes if } x \% 3 == 0]) \]
Part B (6 points)
The label *Generation Z* is applied to anyone born between 1992 and 2010 (inclusive). Write the function `zcount` below that returns the number of entries in parameter `folks` who are part of Generation Z as indicated by their birth-year, the last part of each colon-separated string in `folks`. The parameter `folks` is a list of strings in which each string has the format "firstname:lastname:birthyear".
For example, for the list below the value returned should be 2 since both Elroy Jetson and Angie Smith have birth-years that make them part of Generation Z.


Complete the function below,

```python
def zcount(folks):
    #
    # return number of entries in folks, a list of strings, where
    # birth-year is between 1992 and 2010, inclusive
    #
```

PROBLEM 3 :  (compunction, injunction, disjunction)

Part A (6 points)
Write a function nocommas that returns the int value that corresponds to string parameter num, which has commas inserted appropriately for a number. See the examples.

<table>
<thead>
<tr>
<th>call</th>
<th>return value</th>
</tr>
</thead>
<tbody>
<tr>
<td>nocommas(&quot;1,234&quot;)</td>
<td>1234</td>
</tr>
<tr>
<td>nocommas(&quot;987&quot;)</td>
<td>987</td>
</tr>
<tr>
<td>nocommas(&quot;9,876,543&quot;)</td>
<td>9876543</td>
</tr>
</tbody>
</table>

You can use any method to convert the string to an integer value. Two ideas are:

• use .split("," ) to break the string into parts, create a new string without commas, and then rely on the python operator int where int("531") is the integer 531.

• use .split("," ) to break the string into parts, loop over the parts and on each iteration multiply a running total by 1000 and add the part. For example, for the string "9,876,543" the parts are represented by ["9", "876", "543"] and the running total would have the values shown below as the loop iterates over the parts (the running total is initially zero). Note that each value of the running total is 1000 times the previous value plus the current part.

```
0
9
9876
9876543
```

def nocommas(num):
    ""
    return int equivalent of string num,
    which represents a comma-separated number
    ""
Part B (6 points)
The inverse of no commas is a function `insert_commas` that returns a string with commas inserted representing the parameter `num`.

<table>
<thead>
<tr>
<th>call</th>
<th>return value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>insert_commas(1234)</code></td>
<td>&quot;1,234&quot;</td>
</tr>
<tr>
<td><code>insert_commas(12)</code></td>
<td>&quot;12&quot;</td>
</tr>
<tr>
<td><code>insert_commas(12345)</code></td>
<td>&quot;12,345&quot;</td>
</tr>
<tr>
<td><code>insert_commas(1234567)</code></td>
<td>&quot;1,234,567&quot;</td>
</tr>
</tbody>
</table>

Note that 12345 % 1000 = 345 and that 12345/1000 = 12.

There are many ways to solve this problem. One idea uses this hint: consider the output of executing the Python code below on the left, the output is shown on the right.

```python
n = 123 output: 3 2 1
while n != 0:
    print n % 10,
    n = n / 10
```

Complete the function below.

```python
def insert_commas(num):
    ""
    returns a string representing int num with commas properly inserted
    ""
```
**PROBLEM 4 : (Condominium, House, or Apartment? (18 points))**

**Part A (9 points)**

The code below is an all green solution to the SandwichBar APT whose writeup can be found at the end of this test. You'll be asked some questions about the code.

```python
def whichOrder(available, orders):
    for index in range(len(orders)):
        parts = orders[index].split()
        makes = [1 for p in parts if p in available]
        if sum(makes) == len(parts):
            return index
    return -1
```

1. Explain in a sentence or two the purpose of the loop in line 1, why is the loop over `range(len(orders))`?

2. Explain in a sentence or two the purpose of the list comprehension in line 3 and the if statement in line 4 and how they work together.

3. Explain in a sentence or two why the return -1 statement is after the loop, be sure to explain when the statement would be executed in terms of the APT.
Part B (9 points)
A new APT, titled *Pikachu* is attached to the end of this test. Write the function `check`, in the module `Pikachu.py`, to solve the APT.

```python
def check(word):
```