22C:16 Exam 1

Feb 18th, 9:30-10:20

This is an open notes exam. You have 50 minutes to complete it.

1. [**20 points**] For each of the following expressions, first determine if evaluating it would cause Python to report an error; if so, explain in a sentence what it was about the expression that caused the error. Otherwise, if you think the expression will evaluate in Python in an error-free manner, write down the *value* that the expression evaluates to and its *type*. Assume that the modules `math` and `sys` has already been imported prior to the evaluation of these expressions.

   (a) `float("0." + "55" + "8")`: 0.558, float

   (b) `bool("hello") and (10 <= a)`: True, bool (Assume that the variable `a` has value `30`)

   (c) `math.floor(-8.89) - math.ceil(-8.89)`: -1.0, float

   (d) `87L%5`: 2L, long

   (e) `0.5 + int(raw_input("Enter an integer:  "))`: 30.5, float (Assume that user inputs `30`, when prompted)

   (f) `(x < 20) or (x > 50)`: False, bool (Assume that the variable `x` has value `20`)

   (g) `not bool(0.1)`: False, bool

   (h) `math.round(3.565656, 1)`: Error, because `round` is an in-built Python function, not a function that is part of the `math` module.

   (i) `2*sys.maxint`: $2(2^{31} - 1)$, long

   (j) `0.5 + input("Enter something:  ")`: 20.5, float (Assume that user inputs `10*2`, when prompted to enter something)

2. [**25 points**] This problem is about the following program.

```
word1 = ""
word2 = ""
letter = raw_input("Enter a letter: ")
while letter:
        word1 = word1 + letter
        word2 = letter + word2
        letter = raw_input("Enter a letter: ")

if word1 == word2:
        print "A message"
```

   (a) I run this program and when prompted to enter a letter, I enter the letters `b`, `e`, `s`, and `t` in that order and then when prompted for a letter, I just hit the `enter` key. What are the values of the variables `word1` and `word2` at the beginning of each iteration of the `while`-loop (i.e., when statement `while letter:` is executed). Write these values down neatly, maybe in the form of a table, so that it is clear what value each variable took on first, which values they took on next, etc.

word1	word2
""	""
"b"	"b"
"be"	"eb"
"bes"	"seb"
"best"	"tseb"

(b) Write down a sequence of three or more letters you could input to the program to make it print `A message`.

`m`, `o`, `m`

(c) In one sentence describe the kinds of letter sequences for which the program prints `A message`. Based on this characterization, suggest a better output than `A message` to replace what the program currently prints out.

The program outputs `A message` for letter sequences that are *palindromes*. A better message might be `A palindrome!`

3. **[25 points]** This problem is about the following program.

```
n = int(raw_input("Enter a positive integer:"))
prev = 1
current = 1
count = 2
print current
while count <= n:
    print current
    saved = current
    current = current + prev
    prev = saved
    count = count + 1
```

(a) Make a table showing the values of the variables `prev` and `current` at the beginning of each iteration of the `while`-loop, when the program is executed with input 8.

prev	current
1	1
1	2
2	3
3	5
5	8
8	13
13	21
21	34

(b) What is the output produced when the program is run with input `10`.

```
1
1
2
3
5
```

```
8
13
21
34
55
```

(c) Look at the sequence of numbers you wrote down as answer to the above question. If you have the correct answer, then each term in the sequence should bear a simple relationship to the previous two terms. In other words, the 10th term in the sequence is easily obtained from the 9th term and the 8th term. Describe in one sentence what this relationship is.

The nth term in the sequence, for all $n > 2$, is the sum of the pervious two terms, i.e., term $n-1$ and term $n-2$.

4. **[25 points]** I have been asked to write a program that reads a bunch of positive integers input by the user and count those that are are even and between 1000 and 2000 (inclusive of 1000 and 2000). I have been told that the user will end their input by typing `0`. I made an attempt at writing this program - it is missing two important pieces. Your task is to fill in these two blanks.

```
counter = 0
n = int(raw_input())
while n:
    if (n % 2 == 0) and (n <= 2000) and (n >= 1000):
        counter = counter + 1
    n = int(raw_input())

print counter
```

5. **[25 points]** In many dice games, players use two dice and move their pieces according to the sum of the two numbers that show up on the dice. Since we are talking about 6-sided dice here, the sum of the numbers that show up on two dice can be any number between 2 and 12. I want to write a program that simulates the roll of two 6-sided dice to find out if every number between 2 and 12 is equally likely to show up as the sum. In particular, I want my program to simulate 100 rolls of two 6-sided dice and I will count the number of times the number 4 shows up as the sum (I picked 4 somewhat arbitrarily). Here is a partial version of this program. Your task is to complete it, by filling the three blanks below.

```
import random
counter = 0 # tracks the number of times 4 shows up as the sum
n = 0
while n < 100:
    roll1 = random.randint(1, 6)   # random number produced by dice 1
    roll2 = random.randint(1, 6)   # random number produced by dice 2
    if roll1 + roll2 == 4:
        counter = counter + 1
    n = n + 1
print counter
```

6. [**30 points**] Write a program that starts by prompting the user for a positive integer, let us call this n. The program then reads n floating point numbers input by the user (typed one in each line) and outputs the average of the *positive* numbers input by the user. Here is an example interaction between the program and the user. In this example, the negative number -9.23 is excluded from the computation of the average.

```
Enter the positive integer that denote the length of your sequence: 5
Now input your sequence, one number per line.
3.4
4.5
-9.23
3.456
11.8
The average of the positive numbers is 5.789
```

```
n = int(raw_input("Enter a positive integer:"))
count = 0
positiveCount = 0
sum = 0
while count < n:
    dataPoint = float(raw_input())
    if dataPoint >= 0:
        positiveCount = positiveCount + 1
        sum = sum + dataPoint
    count = count + 1

print "The average is", float(sum)/positiveCount
```