22C:16 CS:1210 Exam 1

Feb 21st, 6:30 pm to 8:30 pm

Instructions:

- This is an open notes exam and you have 2 hours to complete it. There are 4 problems in the exam and these appear on 6 pages. The exam is worth 120 points (12% of your grade) equally distributed among the 4 problems.
- Make sure that you do not have any electronic devices (laptops, phones, calculators, dictionaries, etc.) on your desk; you are not allowed to access these during your exam.
- Write as neatly as you can.
- Show all your work, especially if you want to receive partial credit.

Name:

Circle your section:

SCA (M, W evening) A01 (9:30-10:20) A02 (11-11:50) A03 (12:30-1:20)

A04 (2:00-2:50) A06 (3:30-4:20)

---

1. **[30 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`, `sys`, and `random` have already been imported prior to the evaluation of these expressions. Further, assume that the value of `sys.maxint` is `9223372036854775807`.

   (a) `not (10 >= 0) and (100/0)`

   (b) `long(3**2)/2.0 + 1/2`

   (c) `4 <= int(math.sqrt(random.randint(20, 30)))`

(d) `float(str(5%2) + str(5/2))`

(e) `sys.maxint + 2 - 3`

(f) `bool(0.0001) and ("hello" - "he")`

(g) `-3+-2**3*4%9---10%3`

(h) `random.random()*len(bin(9)) > 9`

(i) `5 + ** 8 - 22`

(j) `(len("forty") + len("four"))%len("three")**2`

2. **[30 points]** For each of the programs below, write down the output produced by the program.

(a) What output does this program produce?

```
n = 1
while n <= 6:
    m = 1
    output = str(n) + ":"

    while m <= n:
        output = output + " " + str(m)
        if (n % 3 == 0) or (m % 3 == 0):
            break

        m = m + 1

    print output
    n = n + 1
```

(b) What output does this program produce?

```
s1 = "ok"
s2 = "bye"
while s1 and s2:
    print s1, s2
    if (len(s1)**2 + len(s2)**2 <= 100):
        s1 = s1 + "bye"
        s2 = s2 + "ok"
    else:
        s2 = ""
```

(c) What output does this program produce?

```
output = ""
n = 7
while True:
    if n < 1:
        break

    s = str(n/2)
    counter = 1
    while counter <= n:
        output = s + output
        counter = counter + 1.5

    print output

    if n%2 == 0 or True:
        n = n/2
```

3. **[30 points]** In this problem, you are given two partially completed programs. Your task is to complete each program.

   (a) The following program reads three strings and prints the shortest string. If there is more than one shortest string, it does not matter which shortest string the program prints. The program is missing boolean expressions that control the **if**-statements in the program. Your task is to fill these blanks.

```
x = raw_input()
y = raw_input()
z = raw_input()

# Check if x is a shortest string

if ____________________________________________ :

    print x

else:
    # Check if y is a shortest string

    if ___________________________________________:

        print y
    else:
        print z
```

(b) The following program reads a floating point number and computes the sum:

$$1 + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \ldots$$

The program contains a `while`-loop that adds the next term of the sum to a variable called `series`. The program stops when it has added 1000 terms *or* when the term it needs to add has become too small, i.e., less than or equal to 0.0000001, whichever happens first. (Recall that the $n!$ you see in the denominator of the terms represents the product $1 \times 2 \times 3 \times \cdots \times (n-1) \times n$.)

```
x = float(raw_input())

series = 0.0
nextTerm = 1.0
counter = 1

# keeping adding nextTerm to series as long as nextTerm is large
# enough and we have not added 1000 terms

while _____________________________________________ :

    series = series + nextTerm

    # Compute a new value for nextTerm from its previous value

    nextTerm = ______________________________

    counter = counter + 1

print series
```

4. **[30 points]** Write a program that reads a sequence of positive integers, one per line. The sequence is terminated by 0. In other words, when the user thinks she is done, she will input 0. Your program should output all *prefix sums* of the input sequence. A prefix sum is simply the sum of the first i elements in the sequence for $i = 1, 2, \ldots$. Thus, if the input sequence is $3, 4, 5$ then the prefix sums are $3, 7, 12$. Here is an example of how the interaction between the user and the program looks. The user inputs the sequence 3, 4, 5 followed by 0 (to terminate the sequence). The program produces output `3 7 12`.

```
3
4
5
0
3 7 12
```

Note that the output prefix sums are expected to appear in the same line, one after the other, separated by a single blank.
Note: My program for this problem is 9 lines long.