Name:_____

CSCI-UA.0002-008 – Midterm Exam #1 (v2)

October 13th, 2015

Instructor: Joseph Versoza

Ask the person to your left for their first name
(leave blank if next to empty seat or wall):

_____

Ask the person to your right for their first name
(leave blank if next to empty seat or wall):

_____

Keep this test booklet closed until the class is prompted to begin the exam

- Computers, calculators, phones, textbooks or notebooks are **not allowed** during the exam
- Please turn off your phone to avoid disrupting others during the exam

1. Write the result, `True` or `False` (or error if applicable) for the following boolean expressions and statements. (2 points)

a) `'42 < '42'` **Error**

b) `'bank' < 'bask'` **True**

c) `42 == '42` **Error or False**

d) `True and Not True or False` **False or error**

2. Read the code in the first column. Answer questions about the code in the second and third columns. (6 points total)

Code	Question #1	Question #2

`result = 0`
`for num in range(18, 3, -3):`
`  if num % 2 == 0:`
`  result += 2`
`  else:`
`  result += num`
`print(result)`
	How many times will this loop run? (1 point)
	
	5
		What is the output of this program? Show your calculations/work. (2 points)
		
		30

count = 1
 	
`while count <= 5:`
 	
`  if count != 3:`
 	
`  print(count)`
`  count += 1`
or only move count to above if
	What is the output of the code on the left? (1 point)
	
	0
	1
	2
	(then loop forever)
		Change/fix the program (you can do this directly in the code in left-most column) so that the output is the same as the following (**do not use** a for loop, and do not use multiple consecutive if statements): (2 points)
		
		1
		2
		4
		5

3. Your friend is part of an avant-garde acapella group, and they've written a program to write the lyrics to their next song. The song's lyrics consists of numbers, "mmmm" and "bzzzz" (um, what? Art!). Your friend's program is supposed to:

a) **print out** numbers from **50** down to (and including) **0**, by **5**'s...
b) after each number, **add** a **random number** (1-5) of **exclamation** points
c) if the number is **greater** than or **equal** to **40**, always print out **bzzzz** (instead of the number and instead of mmmm)
d) however, for the remaining numbers, if the **number ends** in a **0**, print out **mmmm** instead of the number

Unfortunately, their program (shown below) is full of errors. It does not produce the expected output! **Circle 3 errors** (there are more than 3), **identify** if they're a syntax, runtime or logical error... and **briefly explain** why. Draw arrows or label with numbers to associate error with explanation (6 points)

Expected Output

```
bzzzz
bzzzz
bzzzz
35!!!!!
mmmm
25!!!!!
mmmm
15!!!
mmmm
5!
mmmm
```

Broken Code (should produce output on left, but does not!)

```
for i in range(50, 0, -5): 1
    if i % 10 == 0: 2
        print("mmmm")
    else if i => 40: 3, 4
        print("bzzzz")
    else:
        num = random.randint(0, 5) 5
        print(i + num * '!') 6
```

Error #	Type	Explanation
1	**Logical**	**Should be range(50, -1) to include 0**
2	**Logical**	**Switch condition with i >= 40 (>= 40 takes priority for bzzzz)**
3	**Syntax**	**else if should be elif**
4	**Syntax**	**=> should be >=**
5	**Logical**	**Use randint to generate 1 to 5 (not 0 to 5)**
6	**Runtime**	**TypeError: adding non-string (i) to string**

4. Using DeMorgan's laws followed by equivalent expressions with logical opposites, rewrite the boolean expression below: (1 point)

`not (x < 10 and y < 10)`

(a) Using DeMorgan's Laws: **`not (x < 10) or not (y < 10)`**

(b) Removing not's from (a) using logical opposites: **`x >= 10 or y >= 10`**

5. Answer the following questions about loops? (3 points)

   a) In Python, define **count-controlled loop**. What is the construct/control structure (keyword) that represents it?

   A loop that repeats a specific number of times... a for loop in Python

   b) Define **condition-controlled loop**. What is the construct/control structure (keyword) that represents it?

   A loop that repeats as long as a condition is true... a while loop in Python

   c) Explain why you would use one kind of loop over the other?

   Use a for loop when you know how many iterations you want; use a while loop when the number of iterations is based on a condition (number of iterations is not known beforehand)

6. **Name 4 built-in functions** that are not `int`, `str` `float`, or `bool`. Give the type of the return value for each function. (3 points)

	Built-in Function	Type of Return Value (`None` if no value is returned)
#1	print, input	None, str
#2	abs, len	float or int, int
#3	type, format	type, str

7. What is the output of the following code (no output and error are possible)? Note the number of spaces if there is left or right padding. (3 points)

   a) `print(format(0.05, '.2%'))` → 5.00%

   b) `print(format('hello', '<8s') + format(42, '.1f'))` → hello 42.0 3 spaces between

   c) `print(format('25', '.2f'))` → Error

8. In the **truth table** below, fill out all of the possible Boolean values for **p** and **q**, as well as the result of **not p or not q**. (2 points)

p	q	not p or not q
TRUE	TRUE	FALSE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	TRUE

9. Determining what the following program will print out based on the user input specified in the 1st column of the table below. **Show your work for partial credit**. (4 points)

```
n = int(input('Gimmeh a number!\n> '))
if n == 1 or n == 2:
    print(n - 1)
elif n > 0:
    prev = 0
    cur = 1
    for i in range(0, n - 2):
        cur, prev = (prev + cur), cur
    print(cur)
else:
    print('Invalid Input')
```

User Input	Resulting Output to Screen
6	5
1	0
-1	Invalid Input

10. Convert the following numbers . Show work for partial credit. (2 points)

    a) 10000001 is 129 in decimal. b) 14 is 00001110 in binary.

11. Circle all of the **valid variable names** (1 point): $foo Foo _foo 2_foo foo2

12. Name two data **types** in Python that **are not** *numeric*, and give a **syntactically correct** literal example of each. (2 points)

    - type: str, bool example: 'hello', True
    - type: range example: range(5)

13. Write a program that computes that **asks** the user for **two numbers**. It will **compute** the **greatest common factor** that divides evenly into both numbers. (5 points)

    a) Ask for the first number ('Enter num …')
    b) **No validation is necessary** – you can assume that the numbers coming in are whole numbers greater than 0
    c) Print out the largest number that divides evenly into both numbers entered ('The gcf that divides both <num 1> and <num 2> is <greatest common factor>')
    d) Hint: one strategy might be to try all possible numbers (what are the boundaries… does it matter which factor is smaller?) to see which ones are divisors
    e) Hint: what operator would you use to determine if a number is a divisor (divides evenly)?
    f) Hint: your algorithm can be entirely inefficient (that is, you can try factors that you know won't work!)
    g) Example output below:

Run 1:
```
Enter num 1
> 17
Enter num 2
> 4
The gcf that divides both 17 and 4 is 1
```

Run 2:
```
Enter num 1
> 30
Enter num 2
> 36
The gcf that divides both 30 and 36 is 6
```

```
x = int(input('Enter num 1\n> '))
y = int(input('Enter num 2\n> '))
gcf = 1
for factor in range(1, x + 1):
    if x % factor == 0 and y % factor == 0:
        gcf = factor
print('The gcf that divides both', x, 'and', y, 'is', gcf)


# or count down and break
# optionally check to see which one is smaller
```

14. Write a program that continually rolls 2 six-sided dice. Stop rolling dice once at least two “easy fours” (1 and 3) are rolled, and at least one “hard four” (2 and 2) is rolled. (10 points)

a) **Continually** roll two six-sided dice
b) Print out the dice roll: '`rolled <roll1> and <roll2>`'
c) If it's an easy four (one die is 1, and the other is 3), print out '`easy four`'
d) If it's a hard four (both dice are 2), print out '`hard four`'
e) If there are at least **2** “**easy fours**” and **1** “**hard four**” rolled, **stop rolling**!
f) Print out the number of “easy fours”, “hard fours” and total number of rolls (see output below)
g) Hint: notice that you can go over those limits if the other isn't reached
h) Hint: pay close attention to the condition that keeps your loop going!
i) Hint: an “easy four” can be 3, 1 … or 1, 3, while a “hard four” is always just 2, 2
j) Example output below (extraneous output omitted, expanded to two columns to save space – output should be single column)

```
rolled 5 and 4
rolled 5 and 6
rolled 3 and 1
easy four
rolled 5 and 6
<more dice rolls, output omitted>
rolled 3 and 1
easy four
rolled 3 and 4
rolled 3 and 4
rolled 3 and 5
rolled 4 and 6
```

```
<continued from left side>
<more dice rolls, output omitted>
rolled 1 and 3
easy four
rolled 1 and 4
<more dice rolls, output omitted>
rolled 2 and 2
hard four
=====
easy fours: 3
hard fours: 1
total rolls: 44
```

```
import random
easy_min, hard_min = 2, 1
easy_fours, hard_fours = 0, 0
total = 0
while easy_fours < easy_min or hard_fours < hard_min:
    d1, d2 = random.randint(1, 6), random.randint(1, 6)
    print('rolled ' + str(d1) + ' and ' + str(d2))
    if d1 == 1 and d2 == 3 or d1 == 3 and d2 == 1:
        easy_fours += 1
        print('easy four')
    elif d1 == 2 and d2 == 2:
        hard_fours += 1
        print('hard four')
    total += 1
print('=====')
print('easy fours: ' + str(easy_fours))
print('hard fours: ' + str(hard_fours))
print('total rolls: ' + str(total))
```

15. You're writing an essay, and you'd like to avoid using short words (so you can sound super smart!). You decide to write a program to help you out. The program will **ask** you **for words**, one-at-a-time**,** to create an essay **until** the total number of characters (including spaces) of all words entered reach a specified threshold (that is, the **length** of your **essay** is **greater** than the **threshold**). However, your program will **reject words** that are **3 letters or less**. (5 points)

a) Ask the user how many characters they want the essay to be...
b) Continually ask the user for a word until that number of characters is equalled or exceeded: '`Give me word <word #>`'
c) When a user enters a word, only add it to the essay if it's more than 3 characters
d) If it's not more than 3 characters, say '`Oops, that word isn't long enough!`', don't add the word to the essay, and keep going (the word number should not have increased)
e) Add a space after every word (the spaces are included in the calculations for the total number of characters in the essay); it's ok if there's a space at the end of the essay
f) Print out the essay once the number of characters is equalled or exceeded
g) Example output below:

```
How many characters do you want?
>10
Give me word 1
>Hey
Oops, that word isn't long enough!
Give me word 1
>Hello
Give me word 2
>There
Hello There
```

```
+0.5: input
+1:   string accumulator and addition
+1:   while loop condition
+0.5: conditional for less than 4
+1:   add word and space
+0.5: add to count
+0.5: print out sentence
```

```
max_chars = int(input('How many characters do you want?\n>'))
sentence = ''
count = 1
while len(sentence) <= max_chars:
    word = input('Give me word ' + str(count) + '\n>')
    if len(word) >= 4:
        sentence += word + ' '
        count += 1
    else:
        print('Oops, that word isn\'t long enough!')
        print(sentence)
```