**Name:**___________________________________ **Date:**____/____/________

# CSCSCI-UA.0002 – Midterm #2 – Practice Questions

1. If the following code is executed ...

```
a = [4, 5, 6]
b = [3, 1, 2]
a.extend([20, 15])
c = b.sort()
d = [1,2]
a.append(b.pop())
d.append(b)
```

What are the values of a, b, c, and d **after** running the code? (2 points)

(a)_________________________________ (b)_________________________________

(c)_________________________________ (d)_________________________________

2. What is a **list**? Name two **operations** or **constructs** that lists and strings share. (2 points)

3. The following program **draws a dashed line** horizontally across the window. There are five dashes. Each **dash is 20 pixels** long and the **gaps between each dash is 10 pixels** long. **Complete the missing portions** of the code below. (3 points)

```
import _____________________

raphaelle = __________________________

wn = turtle.Screen()

for i in ____________________:

    raphaelle.forward(20)

    raphaelle.____________________

    raphaelle.forward(___________)

    raphaelle.____________________

wn.mainloop()
```

4. Name 3 **list methods** and 3 **string methods**: (3 points)

| **List Methods** | **String Methods** |
|---|---|
| | |

5. What's the difference between a **method** and a **function**? (1 point)

6. Cross out all of the statements that are false or evaluate to False. (3 points)

| | |
|---|---|
| `(a) 7 not in ['hi', None, 7, 3]` | `(e)` strings are immutable |
| `(b) ['a', 6, 4, 5] > ['a', -2, 4, 8]` | `(f)` the last index in a list is the length of that list |
| `(c)` strings are an unordered sequence of characters | `(g) 'twelve'.isdigit()` |

7. Write the output of each line of code. If there's an error, write Error. (3 points)

```
vegetable = "turnip",
i = len(vegetable) – 1
```

```
 a) print(vegetable[2] + vegetable[-2] + vegetable[i])          (a)_____________
 b) print(vegetable[0])                                          (b)_____________
 c) print(vegetable[i + 1])                                      (c)_____________
 d) print("vegetable"[4:100])                                    (d)_____________
 e) print(vegetable[:3])                                         (e)_____________
 f) print(vegetable[0:(i-1)])                                    (f)_____________
```

8. There's an error in both code samples below. Circle the line where the error occurs. To the right of each code sample, explain why there's an error: (3 points)

```
 a) s = 'hello'
    s[0] = 'C'
    s = s.upper()
    print(s)

 b) c = '1'
    numbers = [1, 3, 2] + [4, 5, 6]
    numbers[5] = 9
    numbers[c] = 12

 c) a = say_hello('yo yo meow')
    def say_hello(name):
        print('hello %s' % (name))
```

9. Create a function called it **`unique_and_filter`**... (4 points)

a) it will expect a **list of *only* strings** as an input
b) it will filter that list by:
   - removing duplicates
   - ignoring any string that's three letters or less
   - ignoring any string that only consists of numbers (is numeric)
c) an empty list returns an empty list
d) write two assertions to test your code
e) Sample output:

```
>>> unique_strings = unique_and_filter(['cat', '23', 'four', 'four', 'hello', 'dog'])
>>> print(unique_strings)
['four', 'hello']
```

10. Name the following string methods: (1 point)

    (1) creates a string from a list: ____________________ (2) creates a list from a string ____________________

11. Using the code in the 1st column, answer the questions in the second and third columns. If the question asks for output, **error** is always a possible answer. (3 points)

| **Code** | **Question #1** | **Question #2** |
|---|---|---|
| `def say_cheese(n):`<br>`    s = n * 'cheese'`<br>`    print(s)`<br>`talk = say_cheese(3)` | What is the **output** of this program? | What is the **value** of the variable called **talk**? |
| `exclamation = 'boo'`<br>`whisper = 'shhh'`<br>`def do_something():`<br>`    exclamation = 'bye!'`<br>`    print(whisper)`<br>`do_something()`<br>`print(exclamation)` | What is the **first** line of **output** for this program? | What is the **second** line of **output** for this program? |
| `def join_three(a, b, c):`<br>`    return '%s, %s, %s' % (a, b, c)`<br>`c, b, a = 3, 2, 1`<br>`res = join_three(c, b, a)`<br>`print(res)` | What is the **output** of this program? | What data **type** is returned from join_three? |

12. Implement the following function (note that there are already built-in constructs, functions and methods in Python that provide similar functionality, but we'll be writing our own):

    a) Create a function called **is_in_list** (½ point)
    b) The function should take **two arguments**, an **integer** named **n** and a **list** named **numbers** (½ point)
    c) The function should **return True or False** depending on whether or not the number is in the list (2 points)
    d) Ignore the case where the function either receives non-integer values for the first argument or a value that is not a list of integers for the second argument.
    e) Create two assertions to test your function.
    f) Example Output:

```
>>> print(is_in_list(1, [1, 2, 3]))
True
>>> print(is_in_list(4, [1, 2, 3]))
False
>>> print(is_in_list(4, []))
False
```

13. What is the output of the following lines of code (error is possible)? (1 point)

```
template = '{0} foxes and {0} {1}'
print(template.format('four', 'flamingoes'))

print('%s foxes and %s %s' % ('four', 'flamingoes'))
```

14. Use the following list to answer the questions below...

```
a = [[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12], [13, 14, 15]]
```

**Part 1** - write the code described in the table below based on the list above (2 points)

| Using **list indexing**, write code below to change the number **`15`** to the word '**`fifteen`**' and the number **`4`** to the word '**`four`**'. | Write a loop below that would change the 3rd element in every inner list to an exclamation point: [[1, 2, '!'], [4, 5, '!'] ... ] |
|---|---|
| | |

**Part 2** - write the code described in the instructions below based on the list above (3 points)

Loop through every element in every nested list in the variable **a**.

a) Print out the inner list index before printing out every element of the inner list
b) Example output:

```
List at index 0:
1
2
3
List at index 1:
4
5
6
```

15. Write the following program... (2 points)

a) Stores a list of 4 words: `'dog', 'cat', 'bat', 'frog'`

b) It will then continually ask the user for a word. If the word, **in any casing, lower or upper,** appears in the stored list of words, stop asking for a word and prints out `'Done!'`.

c) Example output:

```
Word please!
> burrito
Word please!
> bat
Done!
```

16. Write a program that asks for exactly 5 words. After all five words are entered, print them out in alphabetical order (hint: there's a list method to do this). (2 points)

17. What is the output of the following program? Use the grid to the right of the program as a guide; **each individual character of output can be placed in a single box**. **Leave a box blank to represent a space character**. You do not have to use all of the boxes. (3 points)

```
def create_table(size, letters):
    table = ''
    for i in range(1, size + 1):
        row = ''
        for c in letters:
            row += str(i) + c + ' '
        table += row + '\n'
    return table

def main():
    num, s = 4, 'abc'
    print(create_table(num, s))
main()
```

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

18. Create a **function called `make_acronym`**. It takes a string as an argument. It extracts words from the string by using space as a word separator. It takes the first letter of each word that **starts in uppercase** to create a new word. (4 points)
See the following examples below (the first one omits 'of' because it does not start with an uppercase letter).

```
>>> print(make_acronym("Bachelor of Arts"))
'BA'

>>> print(make_acronym("Too Long Didn't Read"))
'TLDR'
```