CSCI2202 Lab 6: Exceptions, Files, and More Functions!

Part 1: Exceptions

Q1 [2 points]. Write a function which takes in a temperature in Celsius and checks whether input is valid and then converts it to fahrenheit. If the input isn't valid then raise an appropriate exception that explains the problem. Conditions to check:

input is not a float (or cannot be converted to a float).

Input >= absolute 0 (input is possible!)

Q2 [3 points]. Improve the following function:

- Add the line `return result` in the most appropriate location - use a comment to explain why.
- Add an appropriate docstring
- Complete each of the print statements to output appropriate messages to the user.

```
def divide(x, y):
    try:
        result = x // y
    except ZeroDivisionError:
        print("")
    else:
        print("")
    finally:
        print("")
```

Part 2: Files and Filesystems

Q3 [4 points]. Using `pathlib` create a function which:

- Takes a string containing a folder name as an input
- Convert that string to a `pathlib.Path` object
- Checks whether that folder exists
- If it does, then it iterates over each item within that folder and:
    - prints out the item path
    - prints out whether each item is a folder or a file.
    - If an item is a file then read the file and print out contents.

Q4 [2 points]. Using your internet browser and file browser/explorer, download this zip file to the folder that contains your currently running notebook (i.e., your current working directory) and unzip it. There should now be a folder called `` `directory_treasure_hunt` `` visible to your notebook. Using the function you created in the previous question, navigate the folders and list their contents until you find a file called `little_brother.txt`. Provide the path to this file.

Q5 [2 points]. Read the contents of `little_brother.txt` into a list where each line is an individual item. Add a new line of text containing "and then we made the mushroom soup" to other lines of text in the list. Then create a new file called `little_brother_safe.txt` in `the_village` folder and write each line of text into this new file.

Q 6[3 points]. Write a function which creates a file called `lots_of_numbers.txt` that contains the integers 1 to 10,000,000. However, if the integer is divisible by 4 then print the word "notfizz" instead of the number. Similarly, if the integer is divisible by 7 then print the word "notbuzz". Finally, if the number is divisible by both 4 **and** 7 then print "butter" instead of "notfizz" or "notbuzz"

Part 3: Function Practice

We can determine how many combinations of size `k` there are in a set `n` using binomial coefficient $\binom{n}{k}$ (aka "n choose k"). For example, if $\binom{4}{2}$ is 6 because there are 6 different combinations of item pairs in a list of length 4: ([1,2,3,4] - [1,2], [1,3], [1,4], [2,3], [2,4], [3,4])

We can calculate this in 3 different ways:

1. Recursively

$\binom{n}{k} = 1$ if k=0 or k=n

$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$ otherwise

2. Factorial (you can use `math.factorial`)

$\binom{n}{k} = \frac{n!}{(n-k!) \cdot k}$

3. Recursive Product

$\binom{n}{k} = \binom{n-1}{k-1} \cdot \frac{n}{k}$

Remember: we are doing integer calculations so use `//` for division instead `/` to avoid accidentally converting n and k to floats.

Each of the calculation methods will take different amounts of time to run due to differences in how python performs certain calculations. If we want to measure how long a cell takes to run in jupyter we can put the special command `%%timeit` at the top of the cell. The speed that code runs can vary due to lots of things like your computer doing other things so `%%timeit` will run the cell many times and calculate the mean and standard deviation of how long it takes to run. Depending on how long it takes to run a cell once, `%%timeit` will automatically repeat the cell from 10s to millions of times in batches of "runs".

```
•[1]: %%timeit
      def func(x):
          return x / 2
      func(10)
```

```
62.8 ns ± 0.864 ns per loop (mean ± std. dev. of 7 runs, 10,000,000 loops each)
```

Q7 [10 points]. Using no python modules (apart from `math.factorial`) implement each of these methods in a separate function (e.g., `binomial_recursive`, `binomial_factorial`, and `binomial_product`) and cell - then use `%%timeit` to compare how quickly each method calculates the following:

$\binom{10}{4}$ # 210

$\binom{58}{9}$ # 10,648,873,950

$\binom{14}{10}$ # 1,001

Practice Mid-Term Questions

The following questions are designed to help you test your knowledge and prepare for the mid-term after reading week. You don't have to submit your answers to these (and can just run the code to work out the correct answer) but you are welcome to use the practical time to get help from the TAs about any answers you don't understand.

Remember: you won't have access to anything other than paper and writing implement during the mid-term

Q1. What is the type of each of the following expressions?

```
'5'
5.0
5
[]
[5]
[5,10,15]
(5,10)
{5: 10}
{}
lambda x: x +5
```

Q2. Which of the following types are mutable and which are immutable?

```
Integers (int)
Strings (str)
Floating point numbers (float)
```

```
Lists (list)
Tuples (tuple)
Dictionaries (dict)
Boolean (bool)
```

Q3. What is the boolean value of the following?

```
"hello" in "hello world"
len([1,2,3]) == 4
3 > 2 > 1
[] == False
bool("False")
0.0 or True
False and False or True
```

Q4. What does z contain?

```
z = 2 + 2 * 2 ** 2
```

Q5. What is the value of x, y, and z after this code finishes?

```
x = ['a', 'b']
y = x
z = x + ['c']
y.append ('c')
```

Q6. What will this code print?

```
print (3 * [1, 2, 3])
```

Q7. What does the following code print?

```
for x in range (3):
    for y in range (2):
         print(x, y)
```

Q8. What will each of the following output? Why?

Block A:

```
for i in range(3):
   for j in range(3):
```

```
      if j == 1:
          break
      print(i, j)
```

Block B:

```
for i in range(3):
   for j in range(3):
      if j == 1:
          continue
      print(i, j)
```

Q9. What does rec print(3) print?

```
def rec_print (x):
    print(x, end=' ')
    if x > 1:
        rec_print(x - 1)
        rec_print(x - 1)
```

Q10 For this code:

```
def f(x, y=3, z=5):
    print(x + y + z)
```

What does each of the following inputs result in (TypeError is a possible answer)?

```
f()
f(1)
f(1, 2)
f(1, x=2)
f(1, y=2)
f(1, z=2)
f(1, 2, 3)
f(1, 2, z=3)
f(1, 2, y=3)
```

Q11. What does the following code return?

```
list(zip((1, 2), (3, 4)))
```

Q12. What will this code print?

```
print(sum([2 * x if x % 2 == 0 else 3 * x for x in [2, 3]]))
```

Q13. What will my_list contain after this code has been run?

```
my_list = ["abx", "cdk", "ghy"]
my_list.sorted(key=lambda x: x[2])
```

Q14. What does this print?

```
class A:
    name = "My name"
    def __init__(self, s):
        self.name = s

a = A('Hugo')
print(a.name)
print(A.name)
```

Q15. What does z evaluate to after this code is run?

```
class C:
    def __init__(self, x=1, y=2):
        self.x = x
        self.y = y
    def sum(self):
        return self.x + self.y

z = C(y=3).sum()
```

Q16. What does this print?

```
class A:
    def say(self ):
        print("Hi there")

class B(A):
    def say(self):
        print("Howdy")
    def __init__(self):
        self.say()
        super().say()
A()
B()
```

Q17. What does `data['z']` contain?

```
data = {'x': [1, 2, 3], 'y': [4, 5, 6]}
```

```
data['x'].append(4)
data['z'] = data['x']
data['x'][0] = 10
```

Q18. What does `my_dict` and `new_dict` contain?

```
def modify_dict(d):
    d['a'] = 5
    d = {'a': 3, 'b': 4}
    return d

my_dict = {'a': 1, 'b': 2}
new_dict = modify_dict(my_dict)
```

Q19. What will this code print?

```
prices = {'apple': 0.5, 'banana': 0.3, 'orange': 0.6}
quantities = {'apple': 5, 'orange': 3, 'banana': 2}
print(sum(prices[fruit] * quantities[fruit] for fruit in prices.keys()))
```

Q20. Write a docstring for this function

```
def calculate_discount(price, percentage):
    """
    [Write your docstring here]
    """
    if not isinstance(price, (int, float)) or not isinstance(percentage, (int, float)):
        raise TypeError("Price and percentage must be numbers")
    if percentage < 0 or percentage > 100:
        raise ValueError("Percentage must be between 0 and 100")

    discount = price * (percentage / 100)
    return price - discount
```