12/14 Questions Answered
Saved at 11:45 PM

Final Exam Part 2

STUDENT NAME

Search students by name or email...

Q1 Welcome!

0 Points

Welcome to Part 2 of the Final Exam for CSE 160 21au! Have fun!
Don't forget to also submit Part 1 (function writing)!

Point values on Part 1 and Part 2 are not final and may be adjusted slightly.

(Optional - Just for fun!) If you were designing your own programming language, what animal would YOU name it after and why?

Enter your answer here

Save Answer

Q2 Test/debug/fix

6 Points

As a worker in 160CS corporation, you are tasked with helping customers resolve their errors. A customer sends you the following error after making the function call:

```
new_lst = combine_lists([1, 2, 3, 6], [2, 3, 4, 5], [1, 2])
```

```
Traceback (most recent call last):
  File "CompanyCodeSecrets.py", line 23, in <module>
    new_lst = combine_lists([1, 2, 3, 6], [2, 3, 4, 5], [1, 2])
  File "CompanyCodeSecrets.py", line 20, in combine_lists
    return odd_add_even_multiply(result_1, list3)
  File "CompanyCodeSecrets.py", line 14, in odd_add_even_multiply
    ret_list[i] = longer_list[i]
IndexError: list assignment index out of range
```

You've pinpointed the error back to the following block of code:

```
1 def odd_add_even_multiply(list1, list2):
2    ret_list = []
3    shorter_list = list1
4    longer_list = list2
5    if len(list1) > len(list2):
6        shorter_list = list2
7        longer_list = list1
8    for i in range(len(shorter_list)):
9        if i % 2 == 0:
10           ret_list.append(shorter_list[i] * longer_list[i])
11       else:
12           ret_list.append(shorter_list[i] + longer_list[i])
13   for i in range(len(shorter_list), len(longer_list)):
14       ret_list[i] = longer_list[i]
15   return ret_list
16
17
18 def combine_lists(list1, list2, list3):
19   result_1 = odd_add_even_multiply(list1, list2)
20   return odd_add_even_multiply(result_1, list3)
```

Answer the following questions:
What is another function call to `combine_lists` that could cause a similar error?

new_lst2 = combine_lists([1, 2, 3, 6], [2, 3, 4, 5], [1, 2, 3])
Any call with the third list a different length than the first two will cause this error

What is the cause of the error? **Be specific, refer to line numbers in the code above.**

In the last for loop of odd_add_even_multiply, on line 14, the index of ret_list[i] is reassigned without having been assigned a value in the first place.

How would you fix it? **Be specific, refer to line numbers in the code above.**

To fix it, you could change the code on line 14 to instead be:
ret_list.append(longer_list[i])

Save Answer Last saved on **Dec 20 at 11:36 PM**

Q3 More Test/debug/fix !

6 Points

The following function contains a bug:

```
def list_contains_value(lst, value):
    '''
    Returns True if the lst contains a given value,
    and False otherwise. Assume lst is a non-empty list.
    For example:
    - list_contains_value([3, 2, 1], 1) will return True
    - list_contains_value([3, 2, 1], 4) will return False
    '''
1   result = False
2   for i in lst:
3       if i == value:
4           result = True
5       else:
6           result = False
7   return result
```

Explain what the bug is:

If the value is present, True will only be returned if the value was found as the last value in the list. (otherwise it will overwrite result as False when examining the last element in the list)

Write an assert statement that calls `list_contains_value` that would catch this bug:

The function call is asserted to be True in a statement where the value is present in the list, but not in the last position in the list. e.g.

```
assert list_contains_value([1, 2], 1)
```

Describe how you would fix this bug. **Be specific, refer to line numbers in the given code.**

To fix it, you can remove the else clause on line 5 and anything inside of it. (So delete lines 5 & 6)
else:
  result = False

Save Answer Last saved on **Dec 20 at 11:38 PM**

Q4 List Comprehensions I

2 Points

Write a list comprehension that takes a list of words `lst` and creates a list containing only words that do not contain the letter 'a', preserving the order elements appeared in the original list. **Your solution should be one line of code.**

For example, if `lst = ['apple', 'orange', 'cookie', 'a donut', 'kiwi']`
Then your list comprehension should create the list: `['cookie', 'kiwi']`

```
[word for word in lst if 'a' not in word]
```

Save Answer Last saved on **Dec 20 at 11:38 PM**

Q5 List Comprehensions II

4 Points

You are given `people`, a list of people's names and ages, stored as a list of tuples where each tuple is of the form (string, integer).

Use list comprehensions to produce two new lists based on the contents of the list `people`.

- One list will contain a list of the ages of people whose names are less than 4 characters long.
- The other list will contain a list of the names of people whose ages are greater than 30.

Elements of both lists should occur in the same order as they did in the original list `people`. You should store the lists into the variables `ages_of_short_names` and `names_of_over_30`, respectively.

If `people = [('Ann', 42), ('Ryan', 61), ('Jan', 3), ('Kat', 17)]` then

`ages_of_short_names` should be: `[42, 3, 17]`
`names_of_over_30` should be: `['Ann', 'Ryan']`

Give a list comprehension that uses the list people to create these lists below. **Your solution should be one line of code.**:

`ages_of_short_names =`

```
ages_of_short_names = [age for name, age in people if len(name) < 4]
```

`names_of_over_30 =`

```
names_of_over_30 = [name for name, age in people if age > 30]
```

Save Answer Last saved on **Dec 20 at 11:38 PM**

Q6 Classes

10 Points

Given the classes `Airplane` and `Airport`:

```
class Airplane:
    '''
    Airplane objects represent individual airplanes.
    '''
```

```
    def __init__(self, max_speed, airline, id):
        """
        Construct a new airplane with given max_speed (in mph),
        airline, and unique id
        """
        self.max_speed = max_speed
        self.airline = airline
        self.id = id

    def get_airline(self):
        """
        Returns the plane's airline
        """
        return self.airline

    def get_max_speed(self):
        """
        Returns the plane's max_speed
        """
        return self.max_speed

    def get_id(self):
        """
        Returns the plane's id
        """
        return self.id


class Airport:
    '''
    Keeps track of the airplanes in an airport by storing them
    in a dictionary where the keys are the airlines and the values are
    lists of individual airplanes belonging to that airline.
    '''

    def __init__(self, id, location):
        """
        Constructs a new airport with the given location and id
        """
        self.id = id
        self.location = location
        self.airplanes = {}

    def add_airplane(self, plane):
        """
        Adds the given airplane to this airport
        """
        if plane.get_airline() in self.airplanes:
            self.airplanes[plane.get_airline()].append(plane)
        else:
            self.airplanes[plane.get_airline()] = [plane]
```

Q6.1 Doc String

2 Points

Is there something wrong with the doc string for the `Airport` class? If so, what is incorrect and why does it matter.

The docstring for the Airport class contains too many implementation details (e.g. mentioning that a dictionary is used). Classes are used to provide abstraction, so that they can be used without someone needing to know the specifics of how the class code was written. The docstring should instead focus on the purpose of the class and how it should be used, because that would be a more useful form of documentation for a client.

Save Answer Last saved on **Dec 20 at 11:39 PM**

Q6.2 chi_airport

2 Points

Write code for a client program that would create an instance of the `Airport` class with the `id` "ORD" and the `location` "Chicago" and assign it to the variable `chi_airport`. This code would appear outside of the two classes, in a client program.

```
chi_airport = Airport("ORD", "Chicago")
```

Save Answer Last saved on **Dec 20 at 11:40 PM**

Q6.3 Add marias_plane

2 Points

Write code that will add the airplane `marias_plane` to `chi_airport`. Assume the variable `marias_plane` has already been set to refer to an `Airplane` object. This code would appear outside of the two classes, in a client program, after the code you have written for the problem above.

```
chi_airport.add_airplane(marias_plane)
```

Save Answer Last saved on **Dec 20 at 11:40 PM**

Q6.4 get_fast_airplanes

4 Points

Add a method `get_fast_airplanes(self, airline, min_speed)` to the `Airport` class that will return a list of all of the airplanes for a particular airline that are at least as fast as the provided `min_speed`. If the airline is not present or there are no planes for that airline, return the empty list.

```
def get_fast_airplanes(self, airline, min_speed):
    pln_lst = []
    if airline in self.airplanes:
        for plane in self.airplanes[airline]:
            if plane.get_max_speed() >= min_speed:
                pln_lst.append(plane)
    return pln_lst
```

Save Answer Last saved on **Dec 20 at 11:41 PM**

Q7 Sorting

3 Points

You are given `lst`, a list of strings, where each string represents an integer. For example, the contents might be:

```
lst = ['123', '8', '567', '4', '90']
```

Write code that would sort `lst` in a descending numerical order and put the sorted result in `new_lst`.

For example, given the above contents of `lst`, `new_lst` should be: `['567', '123', '90', '8', '4']`.

Your code should work for any `lst` that is a list of Strings where each string represents an integer. ***You should NOT use a list comprehension for this problem.***

```
new_lst = sorted(lst, key=int, reverse=True)
```

Save Answer Last saved on **Dec 20 at 11:42 PM**

Q8 Data Structures

4 Points

We're designing a class representing recipes, and we want to choose a data structure that could store the recipe ingredients. For each recipe, we want to store both the ingredient and the amount for each ingredient.

One option is to store the ingredients in a dictionary where the keys are the ingredients and the values are the ingredient amount. Another option is to store the ingredients as a list of (ingredient, ingredient_amount) tuples. Give a possible reason to use each approach, by describing what functionality or benefit that data structure could give your class that the other would not.

Benefit of using a dictionary where the keys are the ingredients and the values are the ingredient amount:

With a dictionary, we can easily modify the ingredient amount by changing the value associated with the ingredient key. Tuples can't be modified, so we have to delete the old key value pair and replace it to make any changes.
With a dictionary we can also get the ingredient amount efficiently without having to search through all the other ingredients as we would in a list of tuples to find the ingredient we want.

Benefit of using a list of (ingredient, ingredient_amount) tuples:

With a list of tuples, we can maintain a particular order of our ingredients. Maybe we want to keep our ingredients in the order that they are used in

the recipe. Maybe we want to sort our ingredients alphabetically. Dictionaries are unordered, so we can't use them for this purpose.

Save Answer Last saved on **Dec 20 at 11:43 PM**

Q9 Graphs

4 Points

The dictionary `state_dict` maps states to a list of the states they are adjacent to. **Write code that a) adds appropriate nodes and edges to `state_graph` and then b) displays the graph.** The nodes of the graph are the states that appear in `state_dict` (as keys or values). A node should be added for every state present and an edge should be added whenever two states are adjacent to one another. You can assume "State B" will be in `state_dict["State A"]` if and only if "State A" is in `state_dict["State B"]`. Don't forget to call `draw_networkx`.

For example, given the contents of `state_dict` shown below, there should be an edge between "Washington" and "Oregon" and an edge between "Washington" and "Idaho", because they are adjacent to each other. The full contents of `state_dict` are not shown and your code should work for any contents of `state_dict` (e.g. do not hard-code in nodes and edges for the states shown below).

```
import networkx as nx
import matplotlib.pyplot as plt

# The keys of state_dict are states (strings),
# and the values are lists of states that neighbor the key.
state_dict = {}
state_dict["Washington"] = ["Oregon", "Idaho"]
state_dict["Oregon"] = ["California", "Washington", "Idaho", "Nevada"]
state_dict["California"] = ["Oregon", "Nevada", "Arizona"]
state_dict["Alaska"] = []
… # Full contents of state_dict not shown

state_graph = nx.Graph()

# Your code goes below here!
```

```
for state in state_dict:
    neighbors = state_dict[state]
    if len(neighbors) == 0:
        state_graph.add_node(state)
    for neighbor in neighbors:
        state_graph.add_edge(state, neighbor)
        # The nodes for state and neighbor would be added
        # automatically, but also o.k. to add each node explicitly.

nx.draw_networkx(state_graph)
plt.show()
```

Save Answer Last saved on **Dec 20 at 11:44 PM**

Q10 File I/O

8 Points

You are given the script of a play as a text file. Each line in the file begins with a character's name, followed by a colon (:) and then their corresponding speech. You can assume that the only colon in each line will be the one separating the character name from their speech. Fill in the code to build a `word_counter` dictionary that maps each character name to the total number of words that they say throughout the script. Any punctuation that is next to a word (not separated by a space from the word) should be considered part of that word. For example, if this were the `romeo_juliet_script.txt` file:

```
JULIET: Come hither, nurse. What is yond gentleman?
Nurse: The son and heir of old Tiberio.
JULIET: What's he that now is going out of door?
Nurse: Marry, that, I think, be young Petrucio.
```

The resulting `word_counter` should contain:
`{'JULIET': 16, 'Nurse': 14}`

Starter Code:

```
word_counter = {}
script = "romeo_juliet_script.txt"
```

```
# Write your code here

print(word_counter)
```

```
script_file = open(script)
for line in script_file:
  parts = line.split(":")
  character = parts[0]
  words = parts[1].split()
  count = word_counter.setdefault(character, 0)
  word_counter[character] += len(words)
script_file.close()
```

Save Answer Last saved on **Dec 20 at 11:44 PM**

Q11 You're done!

1 Point

Did you work on this Exam alone or collaborate with others? If you collaborated with others (for either part 1 or Part 2), who did you collaborate with? **Please list full names and UWNetIDs of everyone you collaborated with.**

Enter your answer here

How long did you personally spend on this exam (Part 1 + Part 2)?

Enter your answer here

CONGRATULATIONS! You have now completed CSE 160 21au! Have a wonderful Winter break! Stay Safe!

The CSE 160 Staff

Save Answer

Save All Answers

Submit & View Submission