SCIT, University of Wollongong

CSIT110/CSIT810

SIM Session 2 2021

Assignment 2 (10%) due on Saturday 8 May 2021 at 00:00AM

Objectives

- Able to write clear code with comments and follow coding convention
- Able to use variables with meaningful names and correct data types
- Able to define functions and class objects

Marking criteria:

- Total marks: 10. 1 mark deducted for each day late.
- More than 3 days late will result in zero marks.
- Code must be able to run with no errors: 0 marks for the whole assignment if an error is thrown.
- Correct file format (.py extension): 0 marks for the whole assignment if file submission is not in correct format.
- Use submission template for file submission.

Question 1	Correctness, completeness and consistency with the assignment specification	2 mark
Question 2	Correctness, completeness and consistency with the assignment specification	2 marks
Question 3	Correctness, completeness and consistency with the assignment specification	2 marks
Question 4	Correctness, completeness and consistency with the assignment specification	2 marks
Question 5	Correctness, completeness and consistency with the assignment specification	2 marks
Question 6	Correctness, completeness and consistency with the assignment specification	0 marks (Challenge yourself)
Overall	Script must include name, student number, subject code; clear code and follow coding convention. Use variables with meaningful names and correct data types	Deduct up to 1 mark

Submission Instruction: Assignment 2 submission is on Moodle. Put all your Python code into a single Python file (file extension .py) and submit it.

Assignment questions: there are 6 assignment questions.

Write clear code with **comments** and follow **coding conventions**. Script should include **your name**, **student number** and **subject code** on top of your code. Please also add this information to the variables as stated in the template Your code must work **exactly** like the provided examples given the input in the examples.

```
name = "John Snow"
student_num = "1234567"  # Student number
subject_code = "CSIT110"  # CSIT110 or SP121
```

Question 1. Write a function that satisfies the following specifications.

Function name	`map_to_dict`
Parameters	1. `list1: list` 2. `list2: list`
Return Value	1. a `dict`
Detailed description	Create and return a dictionary object with the elements in the first list, `list1`, as keys and the elements of the same indices in the list, `list2`, as the values. You may assume the lengths of both list arguments are the same and not equal to zero.
Example	`map_to_dict(["one", "two", "three"], [1, 2, 3])`
Example's console output	`{"one":1, "two":2, "three":3}`

Question 2. Write a function that satisfies the following specifications.

Function name	`report_cases`
Parameter	1. `an int` `2.a str`
Return Value	-
Detailed description	Display the following text, substitute N with the int argument, and disease with the str argument `There are N disease case(s) today.` The int parameter is an optional parameter with a default value of 0. The str parameter is an optional parameter with a default value of "heat stroke"
Example	`report_cases()` `report_cases(199,"heart attack")`
Example's console output	`There are 0 heat stroke case(s) today.` `There are 199 heart attack case(s) today.`

Question 3. Write a function that satisfies the following specifications.

Function name	`get_subscription`
Parameters	-
Return Value	-
Detailed description	With the help of a dictionary, the function must display the six healthy food items (6 Apples, Mixed Vegetables Pack 1kg, Yakult 5 in a pack, Mixed nuts 500g, Milk Powder 200g, Roasted Chicken Breast 1kg)available for subscription. Take subscription orders from user inputs. Your program should work exactly as the following examples. The text in **bold** indicates user input. There is a space after the colon symbols. The first column, containing the names of the food items, is left-aligned and 41 characters in width. The second column is right-aligned and 9 characters in width. The bullets in the list of selection is a space, a hyphen and a space. i.e. " – " Finally display the total cost. Take note of and implement the newlines between each section of text as shown in the examples. You may assume the inputs are of letters n and y, of lower and uppercase.

Examples are on the next page.

Example 1: The user subscribes to 2 items

Console output:

```
Food items available for subscription (price/week)
6 Apples                                    $5.40
Mixed Vegetables Pack 1kg                   $9.32
Yakult 5 in a pack                          $3.20
Mixed nuts 500g                            $16.98
Milk Powder 200g                            $9.47
Roasted Chicken Breast 1kg                  $8.56

Add "6 Apples" to subscription? (Y/N): Y
Add "Mixed Vegetables Pack 1kg" to subscription? (Y/N): N
Add "Yakult 5 in a pack" to subscription? (Y/N): N
Add "Mixed Nuts 500g" to subscription? (Y/N): N
Add "Milk Powder 200g" to subscription? (Y/N): Y
Add "Roasted Chicken Breast 1kg" to subscription? (Y/N): N

Your selection:
 - 6 Apples ($5.40)
 - Milk Powder 200g ($9.47)

Total cost $14.87
```

Example 2: User did not subscribe to any items

Console output

```
Food items available for subscription (price/week)
6 Apples                                    $5.40
Mixed Vegetables Pack 1kg                   $9.32
Yakult 5 in a pack                          $3.20
Mixed nuts 500g                            $16.98
Milk Powder 200g                            $9.47
Roasted Chicken Breast 1kg                  $8.56

Add "6 Apples" to subscription? (Y/N): n
Add "Mixed Vegetables Pack 1kg" to subscription? (Y/N): N
Add "Yakult 5 in a pack" to subscription? (Y/N): N
Add "Mixed Nuts 500g" to subscription? (Y/N): N
Add "Milk Powder 200g" to subscription? (Y/N): N
Add "Roasted Chicken Breast 1kg" to subscription? (Y/N): n

Your selection:
 - None

Total cost $0.00
```

Question 4. Write a function that satisfies the following specifications.

Function	`generate_qns_from_list`
Parameters	1. A list of list objects.
Return Value	1. A list of dictionary objects
Detail Information	The first item in each element in the list argument is a string, while the rest of the items are integers. i.e. `[[str,int,int], [str,int,int,int, ...], [str, int, ...], ...]` The str item contains an arithmetic operator without spaces, i.e. `"+", "-", "x", "/".` This function should convert the argument into a list of dictionaries. Each dictionary will have two keys – `"qns"` and `"ans"`. The value for the key `"qns"` will be a str-typed object containing integers taken from each list element. In the str-typed object, the integers are separated with the characters space, operator, space. The operator is given in the first item of each list element, e.g. `" + ", " - ", " x ", " / ".` The value for the `"ans"` key will be the sum/product/quotient of all the numbers in the list, depending on the arithmetic operator given You should skip the lists that contain fewer than 2 integers. There is no need to print anything.

An example is given on the next page.

Example:

```
input_list = [["+",1,3,3], ["-",2,5,-1],
              ["x",3,2],["/",12,3,2],["x",0,23],["+",1,2,3,4]]

generate_qns_from_list(input_list)
```

```
[{"qns": "1 + 3 + 3", "ans": 7},
 {"qns": "2 - 5 - -1", "ans": -2},
 {"qns": "3 x 2", "ans": 6},
 {"qns": "12 / 3 / 2", "ans": 2},
 {"qns": "0 x 23", "ans": 0},
 {"qns": "1 + 2 + 3 + 4", "ans": 10}]
```

Question 5a.
Create a class that satisfies the following specifications.

Class name	`Backpack`
Class attribute	`style: str`
Instance attributes	`account_id: str` `inventory: dict`
Parameters	`account_id: str`
Detailed information	Assign the str `"basic"` to the class attribute, `style`. The value of the instance attribute, account_id, should be assigned with the argument of the same name in the constructor. The instance attribute, `inventory`, is an empty dictionary when a `Backpack` object is instantiated. The keys in the dictionary are the items' names, of str type, while the values, of int type, mark the quantity of the items in the cart.

Question 5b
Write an instance method for the BackPack class that satisfies the following specifications.

Method name	`add_item_to_bag`
Parameters	1. `item_name: str` 2. `quantity: int`
Return values	-
Detailed description	The function should check if the item exists in the keys of the instance attribute, `inventory`, and update the number of the items in the inventory accordingly. That is, if the item does not exist in the inventory, it should create a key-value pair for the inventory. Otherwise, it should add the quantity in the argument to the existing value. You may assume that the quantity in the argument will be more than 0.

Question 5c
Write an instance method for the ShoppingCart class that satisfies the following specifications.

Method name	`remove_item_from_bag`
Parameter	1. `item_name: str` 2. `quantity: int`
Return value	-
Detailed description	The function should update the `inventory` attribute with the correct number of items. If the updated quantity is 0, the key value pair should be removed from the inventory. You can assume that the item name in the argument is in the inventory and the quantity to be removed is less than or equal to the quantity in the inventory.

Question 5d

Write an instance method for the BackPack class that satisfies the following specifications.

Method name	`empty_bag`
Parameter	–
Return value	–
Detailed description	This method empties the instance attribute, `inventory`

Question 5e

Write an instance method for the BackPack class that satisfies the following specifications.

Method name	`count_items`
Parameter	–
Return value	1. `int`
Detailed description	This method returns the total number of items based on the existing items in the inventory. If there is nothing in the inventory, the return value should be 0.

An example of how your code should work given the following parameters.

```
bag = BackPack("394eds7A")
bag.add_item_to_bag("dagger", 1)
bag.add_item_to_bag("wooden sword", 2)
bag.add_item_to_bag ("red potion", 3)
bag.add_item_to_bag ("red potion", 2)
# bag.inventory is now {"dagger": 1, "wooden sword": 2, "red
potion": 5}
bag.remove_item_from_bag("wooden sword", 1)
bag.remove_item_from_bag("red potion", 5)
# bag.inventory is now {"dagger": 1, "wooden sword": 1}
bag.count_items()  # returns 2
bag.empty_bag()
bag.count_items()  # returns 0
```

Question 6. Challenge Yourself
Given that the `student` dictionary object is one with keys `name` and `results`, an example of the `student` dictionary object is as follows:

```
{
      "name": "Fus Ro Dah",
      "results": {
            "assignment_1": 10,
            "assignment_2": 10,
            "examination_1": 10,
       },
}
```

Question 6a.
Define a class `Student` that satisfies the following specifications.

Class name	`Student`
Instance attributes	1. `name: str` 2. `results: Dict[str,int]`
Parameter	1. `student dictionary` object
Detailed information	The class constructor should accept a `student` **dictionary** object, as seen in the example above, and instantiate the instance attributes accordingly

Question 6b

Write a class method for the Student class that satisfies the following specifications.

Method name	`dict_to_class_obj`
Parameter	1. a `list: list[dict]`
Return value	1. a `list: list[Student]`
Detailed description	The function should return a list of `Student` class objects, constructed from the list of student dictionaries and assigning the instance attributes with the dictionaries' keys of the same name.

Here is an example of a list argument

```
[{
   "name": "Fus Ro Dah",
   "results": {
           "assignment_1": 10,
           "assignment_2": 10,
           "examination_1": 10,
           },
 },{
   "name": "Foo Barry",
   "results": {
          "assignment_1": 1,
          "assignment_2": 2,
          "examination_1": 3,
          },
 }]
```

Question 6c

Write an instance method for the Student class that satisfies the following specifications.

Method name	`get_weighted_result`
Parameter	1. `weights: Dict[str, float]`
Return value	1. `results: float`
Detailed description	A weighted result is the product of the value in the `weights` dictionary and the value in the `results` dictionary that has the same key. You may assume that the keys in the `weights` dictionary is a subset of keys from the `results` dictionary. This method should return a single value – the weighted sum of the results based on the keys from the `weights` dictionary.

Here is an example of the input `weights` dictionary.

```
weights  = {"assignment_1": 1.0, "examination_1": 9.0}
```

Using the example above and that in question 4b, the weighted result for the student named Foo Barry is 28.

assignment_1: 1.0 * 1 = 1
examination_1: 9.0 * 3 = 27
weighted result: 28

Please note that not all key-value pairs present in the `results` dictionary may be included in the weights dictionary. In the example above, `assignment_2` was excluded. You can assume that the keys in the weights are all present in the results dictionary.