

Course: Introduction to Data Science (DS2006) - Laboratory 07

Task 1 & 2:

Currently we are using lists to store player information. If we want to be able to store additional information about the player, we would run into a problem where we would be creating more and more lists to store and handle additional player data. This is not a dynamic and scalable solution. A necessary change would be to create a Player class that holds information such as name, wins, roll history and perhaps even win rate. By doing this we can add more methods to the player class and keep things structured rather than having tens of lists.

Task 3: inventory.py file created

Task 4:

```
1  #Keep track of inventory:
2  inventory = {
3      "apple": 50,
4      "banana": 30,
5      "orange": 20
6  }
7
8  print(inventory)
9
10 # Output:
11 # {'apple': 50, 'banana': 30, 'orange': 20}
```

A string representation of the `inventory` dictionary is printed with each pair showing the key and its corresponding value.

Task 5: inventory.py modified

Task 6:

```
1  #Keep track of inventory:
2  inventory = {
3      "apple": 50,
4      "banana": 30,
5      "orange": 20
6  }
7
8  inventory.update({"pear": 10})
9  inventory.update({"pineapple": 5})
10
11 print(inventory)
12
13 # Output:
14 # {'apple': 50, 'banana': 30, 'orange': 20, 'pear': 10, 'pineapple': 5}
```

With the new changes made to the code the output now includes 2 additional pairs (pear and pineapple). The `.update` method adds the new key-value pair to the end of the dictionary, and since we added `pear` and `pineapple` it will print out the dictionary with those two pairs as well as the original pairs.

Task 7: inventory.py modified

Task 8:

```
1  visited_places = {
2      "city": "",
3      "country": "",
4      "year": "",
5  }
6  visited_places["city"] = input("Enter city name: ")
7  visited_places["country"] = input("Enter country name: ")
8  visited_places["year"] = input("Enter the year: ")
9  print(visited_places)
10
11 # Output:
12 # Enter city name: Trollhättan
13 # Enter country name: Sweden
14 # Enter the year: 2024
15 # {'city': 'Trollhättan', 'country': 'Sweden', 'year': '2024'}
```

Task 9:

```
1  visited_places = {
2      "city": "",
3      "country": "",
4      "year": "",
5  }
6  my_visited_places = []
7  for i in range(0, 2):
8      visited_places["city"] = input("Enter city name: ")
9      visited_places["country"] = input("Enter country name: ")
10     visited_places["year"] = input("Enter the year: ")
11     my_visited_places.append(visited_places)
12 print(my_visited_places)
13
14 # Output:
15 # Enter city name: Trollhättan
16 # Enter country name: Sweden
17 # Enter the year: 2024
18 # Enter city name: Borås
19 # Enter country name: Sweden
20 # Enter the year: 2024
21 # [{'city': 'Borås', 'country': 'Sweden', 'year': '2024'}, {'city': 'Borås', 'country': 'Sweden', 'year': '2024'}]
```

Task 10:

```
1 visited_places = {
2     "city": "",
3     "country": "",
4     "date": "",
5 }
6 my_visited_places = []
7 for i in range(0, 2):
8     # Make a copy of the dictionary template:
9     places = visited_places.copy()
10    places["city"] = input("Enter city name: ")
11    places["country"] = input("Enter country name: ")
12    places["date"] = input("Enter the year: ")
13    my_visited_places.append(places)
14 print(my_visited_places)
15
16 # Output:
17 # Enter city name: Trollhättan
18 # Enter country name: Sweden
19 # Enter the year: 2024
20 # Enter city name: Borås
21 # Enter country name: Sweden
22 # Enter the year: 2024
23 # [{'city': 'Trollhättan', 'country': 'Sweden', 'date': '2024'}, {'city': 'Borås', 'country': 'Sweden', 'date': '2024'}]
```

Task 11:

```
1 visited_places = {
2     "city": "",
3     "country": "",
4     "dates": [],
5 }
6 my_visited_places = []
7 for i in range(0, 2):
8     # Make a copy of the dictionary template:
9     places = visited_places.copy()
10    places["city"] = input("Enter city name: ")
11    places["country"] = input("Enter country name: ")
12    times = input("How many times you visited?")
13    for j in range(0, int(times)):
14        year = input(f"Enter the year of the {j+1} time you went there: ")
15        places["dates"].append(year)
16    my_visited_places.append(places)
17 print(my_visited_places)
18
19 # Output:
20 """
21 Enter city name: Trollhättan
22 Enter country name: Sweden
23 How many times you visited?6
24 Enter the year of the 1 time you went there: 2023
25 Enter the year of the 2 time you went there: 2023
26 Enter the year of the 3 time you went there: 2024
27 Enter the year of the 4 time you went there: 2024
28 Enter the year of the 5 time you went there: 2024
29 Enter the year of the 6 time you went there: 2024
30 Enter city name: Borås
31 Enter country name: Sweden
32 How many times you visited?2
33 Enter the year of the 1 time you went there: 2024
34 Enter the year of the 2 time you went there: 2024
35 [
36     {'city': 'Trollhättan', 'country': 'Sweden', 'dates': ['2023', '2023', '2024', '2024', '2024', '2024', '2024', '2024']},
37     {'city': 'Borås', 'country': 'Sweden', 'dates': ['2023', '2023', '2024', '2024', '2024', '2024', '2024', '2024']}
38 ]
39 """
```

Task 12:

```
1 import copy
2 visited_places = {
3     "city": "",
4     "country": "",
5     "dates": [],
6 }
7 my_visited_places = []
8 for i in range(0, 2):
9     # Make a deep copy of the template for this player
10    places = copy.deepcopy(visited_places)
11    places["city"] = input("Enter city name: ")
12    places["country"] = input("Enter country name: ")
13    times = input("How many times you visited?")
14    for j in range(0, int(times)):
15        year = input(f"Enter the year of the {j+1} time you went there: ")
16        places["dates"].append(year)
17    my_visited_places.append(places)
18 print(my_visited_places)
19
20 """
21 Output:
22
23 Enter city name: Trollhättan
24 Enter country name: Sweden
25 How many times you visited?6
26 Enter the year of the 1 time you went there: 2023
27 Enter the year of the 2 time you went there: 2023
28 Enter the year of the 3 time you went there: 2024
29 Enter the year of the 4 time you went there: 2024
30 Enter the year of the 5 time you went there: 2024
31 Enter the year of the 6 time you went there: 2024
32 Enter city name: Borås
33 Enter country name: Sweden
34 How many times you visited?2
35 Enter the year of the 1 time you went there: 2024
36 Enter the year of the 2 time you went there: 2024
37 [
38     {'city': 'Trollhättan', 'country': 'Sweden', 'dates': ['2023', '2023', '2024', '2024', '2024', '2024']},
39     {'city': 'Borås', 'country': 'Sweden', 'dates': ['2024', '2024']}
40 ]
41 """
```

Task 13: better_places.py submitted

Task 14: multiplayer-battle-of-dices-dict.py submitted

Task 15:

When the program is run with 4 players, the first 2 rounds were draws and thus the bug did not show itself. However, in round 3 and forward all 4 players were incorrectly declared winners of the round in each round until the game ended. The built in `dict.copy()` method creates a shallow copy of the dictionary which means mutable objects like the `rolls` list in `player_info` are not copied and remain as references. In the case of `bugged-multiplayer-battle-of-dices.py` whenever a new roll is added to any of the players' `rolls` list it is inserting a new element into the same `rolls` list for all the players. This results in an inaccurate calculation of the round's winner and thus the winners of the current round are always all the players. This demonstrates the limitation and use cases for the built in `dict.copy()` method. It is most effective for copying only the immediate fields of a dictionary and keeping nested dictionaries or lists as references to the original dictionary. By using `copy.deepcopy()`, the dictionary is recursively copied and allows a unique `rolls` list for each player.

Task 16: multiplayer-battle-of-dices.py submitted