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:

```python
#Keep track of inventory:
inventory = {
    "apple": 50,
    "banana": 30,
    "orange": 20
}

print(inventory)

# Output:
# {'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:

```python
#Keep track of inventory:
inventory = {
    "apple": 50,
    "banana": 30,
    "orange": 20
}

inventory.update({"pear": 10})
inventory.update({"pineapple": 5})

print(inventory)

# Output:
# {'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 pair 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:

```python
visited_places = {
"city": "",
"country": "",
"year": "",
}
visited_places["city"] = input("Enter city name: ")
visited_places["country"] = input("Enter country name: ")
visited_places["year"] = input("Enter the year: ")
print(visited_places)

# Output:
# Enter city name: Trollhättan
# Enter country name: Sweden
# Enter the year: 2024
# {'city': 'Trollhättan', 'country': 'Sweden', 'year': '2024'}
```

Task 9:

```python
visited_places = {
    "city": "",
    "country": "",
    "year": "",
}
my_visited_places = []
for i in range(0, 2):
    visited_places["city"] = input("Enter city name: ")
    visited_places["country"] = input("Enter country name: ")
    visited_places["year"] = input("Enter the year: ")
    my_visited_places.append(visited_places)
print(my_visited_places)

# Output:
# Enter city name: Trollhättan
# Enter country name: Sweden
# Enter the year: 2024
# Enter city name: Borås
# Enter country name: Sweden
# Enter the year: 2024
# [{'city': 'Borås', 'country': 'Sweden', 'year': '2024'}, {'city': 'Borås', 'country': 'Sweden', 'year': '2024'}]
```

Task 10:

```python
visited_places = {
    "city": "",
    "country": "",
    "date": "",
}
my_visited_places = []
for i in range(0, 2):
    # Make a copy of the dictionary template:
    places = visited_places.copy()
    places["city"] = input("Enter city name: ")
    places["country"] = input("Enter country name: ")
    places["date"] = input("Enter the year: ")
    my_visited_places.append(places)
print(my_visited_places)

# Output:
# Enter city name: Trollhättan
# Enter country name: Sweden
# Enter the year: 2024
# Enter city name: Borås
# Enter country name: Sweden
# Enter the year: 2024
# [{'city': 'Trollhättan', 'country': 'Sweden', 'date': '2024'}, {'city': 'Borås', 'country': 'Sweden', 'date': '2024'}]
```

Task 11:

```python
visited_places = {
"city": "",
"country": "",
"dates": [],
}
my_visited_places = []
for i in range(0, 2):
    # Make a copy of the dictionary template:
    places = visited_places.copy()
    places["city"] = input("Enter city name: ")
    places["country"] = input("Enter country name: ")
    times = input("How many times you visited?")
    for j in range(0, int(times)):
        year = input(f"Enter the year of the {j+1} time you went there: ")
        places["dates"].append(year)
    my_visited_places.append(places)
print(my_visited_places)

# Output:
"""
Enter city name: Trollhättan
Enter country name: Sweden
How many times you visited?6
Enter the year of the 1 time you went there: 2023
Enter the year of the 2 time you went there: 2023
Enter the year of the 3 time you went there: 2024
Enter the year of the 4 time you went there: 2024
Enter the year of the 5 time you went there: 2024
Enter the year of the 6 time you went there: 2024
Enter city name: Borås
Enter country name: Sweden
How many times you visited?2
Enter the year of the 1 time you went there: 2024
Enter the year of the 2 time you went there: 2024
[
    {'city': 'Trollhättan', 'country': 'Sweden','dates': ['2023', '2023', '2024', '2024', '2024', '2024', '2024', '2024']},
    {'city': 'Borås', 'country': 'Sweden', 'dates': ['2023', '2023', '2024', '2024', '2024', '2024', '2024', '2024']}
]
"""
```

Task 12:

```python
import copy
visited_places = {
"city": "",
"country": "",
"dates": [],
}
my_visited_places = []
for i in range(0, 2):
    # Make a deep copy of the template for this player
    places = copy.deepcopy(visited_places)
    places["city"] = input("Enter city name: ")
    places["country"] = input("Enter country name: ")
    times = input("How many times you visited?")
    for j in range(0, int(times)):
        year = input(f"Enter the year of the {j+1} time you went there: ")
        places["dates"].append(year)
    my_visited_places.append(places)
print(my_visited_places)

"""
Output:

Enter city name: Trollhättan
Enter country name: Sweden
How many times you visited?6
Enter the year of the 1 time you went there: 2023
Enter the year of the 2 time you went there: 2023
Enter the year of the 3 time you went there: 2024
Enter the year of the 4 time you went there: 2024
Enter the year of the 5 time you went there: 2024
Enter the year of the 6 time you went there: 2024
Enter city name: Borås
Enter country name: Sweden
How many times you visited?2
Enter the year of the 1 time you went there: 2024
Enter the year of the 2 time you went there: 2024
[
    {'city': 'Trollhättan', 'country': 'Sweden', 'dates': ['2023', '2023', '2024', '2024', '2024', '2024']},
    {'city': 'Borås', 'country': 'Sweden', 'dates': ['2024', '2024']}
]
"""
```

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