program2_2140232

July 29, 2022

0.0.1 Program 2 - mutable and immutable datatypes

Joel Varghese

2140232

29-7-22 Lets say we have a huge dataset describing the information of people who have a term deposit with the bank with each row representing each people. We can extract the information of each people into a list for further modification.

[ ]:
```
N1 = ["Jacob Thomas", "Male","33","self-employed","single",1467]

N1
```

[ ]:
```
['Jacob Thomas', 'Male', '33', 'self-employed', 'single', 1467]
```

0.0.2 using insert and remove functions on list

Now lets say that this person has gotten married within the time he first made a term deposit to now. we need to change the status from "single" to "married". For this we can use the insert and remove function.

[ ]:
```
N1.remove("single")
N1
```

[ ]:
```
['Jacob Thomas', 'Male', '33', 'self-employed', 1467]
```

[ ]:
```
N1.insert(4,"married")
N1
```

[ ]:
```
['Jacob Thomas', 'Male', '33', 'self-employed', 'married', 1467]
```

0.0.3 using pop/append operations to keep track of people waiting for loans

lets say we have a tuple countaining the names of people who are awaiting a loan from the bank. Now we want to access those names one by one but also remove them from the tuple. since a tuple is immutable we need to first convert it into a list to make changes. we can use the pop function.

```
[ ]: loan_queue = ("joel jones", "maria thomas", "jennifer lopez","sean hatheway")

     loan_queue = list(loan_queue)

     print(loan_queue.pop(0))
```

```
joel jones
```

using the pop operator we were able to access and remove the name from the list. if we print the list again it willl no longer be available in the list.

```
[ ]: loan_queue
```

```
[ ]: ['maria thomas', 'jennifer lopez', 'sean hatheway']
```

now if we another person is trying to acces loans we can add their name usinf the append function.

```
[ ]: loan_queue.append("lindsey stirling")
```

```
[ ]: loan_queue
```

```
[ ]: ['maria thomas', 'jennifer lopez', 'sean hatheway', 'lindsey stirling']
```

0.0.4 Using dictionnaries to store information for easy access

Dictionaries can be used to store key-value pairs of data. since these values are stored using hash mapping algorithms they can be accessed very quick.

```
[ ]: data1 = {"joel jones":["Male",33,"working",(9,7,2022),12363],
              "maria thomas":["Female",23,"student",(4,5,2022),42524],
              "jennifer lopez":["Female",29,"working",(4,5,2022),64533],
              "sean hatheway":["Male",27,"working",(20,6,2022),34734],
             "lindsey stirling":["Female",31,"working",(3,6,2022),34839],
              "jack write":["Male",25,"student",(21,5,2022),66724]
             }
```

the dictionary above stores the names as keys and the value as a list containing information such as gender,age,working status,date of openong bank account, and account number. we cab access all these information using the names.

```
[ ]: print(data1["joel jones"])
     print(data1["jennifer lopez"])
```

```
['Male', 33, 'working', (9, 7, 2022), 12363]
['Female', 29, 'working', (4, 5, 2022), 64533]
```

```
[ ]: data1["joel jones"][2]
```

```
[ ]: 'working'
```

0.0.5 Using sets to store individuals with similar charecteristics and deriving further information

we can store the names of female account holders and working account holders in two sets.

[ ]:
```
set_working = set()
set_female = set()

for k in data1:
    if data1[k][0] == "Female":
        set_female.add(k)
    if data1[k][2] == "working":
        set_working.add(k)
```

[ ]:
```
print(set_working)

print(set_female)
```

```
{'lindsey stirling', 'joel jones', 'jennifer lopez', 'sean hatheway'}
{'lindsey stirling', 'maria thomas', 'jennifer lopez'}
```

we can use these sets to find further commonalities.

example1 : names of women who are working can be the intersection of set of people who are working and set of people who are female

[ ]:
```
print(set_working.intersection(set_female))
```

```
{'jennifer lopez', 'lindsey stirling'}
```

example2 : names of men who are students

[ ]:
```
set_male = set()
set_student = set()
for k in data1:
    if data1[k][0] == "Male":
        set_male.add(k)
    if data1[k][2] == "student":
        set_student.add(k)


print("students : ",set_student)

print("males : " ,set_male)

print("males who are students : ", set_male.intersection(set_student))
```

```
students :  {'maria thomas', 'jack write'}
males :  {'joel jones', 'jack write', 'sean hatheway'}
males who are students :  {'jack write'}
```