GOVERNMENT OF WESTERN AUSTRALIA

TAFE WA South Metropolitan

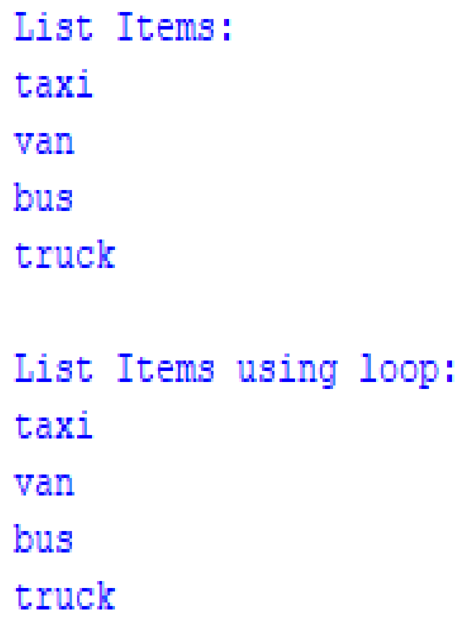
Data Structures – Tuples and Dictionaries

(Session 9)

Review - Lists

```
# intializing list and print all the index numbers
vehicle_list = ["taxi", "van", "bus", "truck"]
print("\nList Items: ")
print(vehicle_list[0])
print(vehicle_list[1])
print(vehicle_list[2])
print(vehicle_list[3])

print("\nList Items using loop: ")
for i in range(len(vehicle_list)):
    print(vehicle_list[i])
```



```
List Items:
taxi
van
bus
truck

List Items using loop:
taxi
van
bus
truck
```

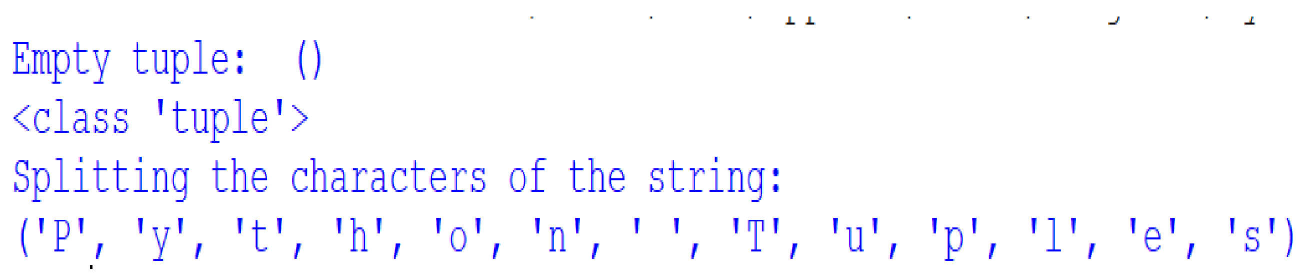
Overview

- Introduction to Python Tuples
- How to Access Tuple Items
- How to Loop Through a Tuple
- Tuple methods
- Introduction to Python Dictionaries
- Adding and Removing Items
- Looping Dictionaries
- Dictionary Methods

Python Tuples

- Tuples are ordered and immutable
- Tuples use round brackets **( )**
- Example:

```
my_tuple = ()  # empty tuple
print("Empty tuple: ", my_tuple)
print(type(my_tuple))
my_string = ("Python Tuples")
# converts string to tuple
my_tuple = tuple(my_string)
print("Splitting the characters of the string:")
print(my_tuple)
```



```
Empty tuple:  ()
<class 'tuple'>
Splitting the characters of the string:
('P', 'y', 't', 'h', 'o', 'n', ' ', 'T', 'u', 'p', 'l', 'e', 's')
```

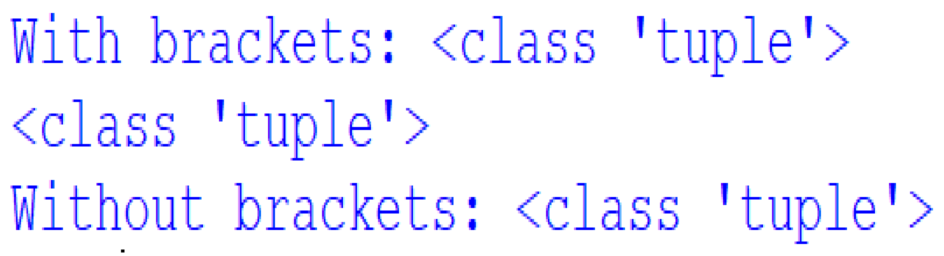
Creating a Tuple

```
# this is not a tuple as parathesis define operator precedence
my_tuple = (5)
print("Not a tuple, but just:", type(my_tuple))
# not a tuple, but string
my_tuple = ("5")
print("Not a tuple, but just:", type(my_tuple))
```

```
Not a tuple, but just: <class 'int'>
Not a tuple, but just: <class 'str'>
```

Creating a Tuple (cont.)

```
# to write a single value (tuple) you need to include a comma
my_tuple = (5,)
print(“With brackets:", type(my_tuple))
my_tuple = ("5",)
print(type(my_tuple))
# creating a tuple without round brackets
my_tuple = 5,
print("Without brackets:", type(my_tuple))
```



```
With brackets: <class 'tuple'>
<class 'tuple'>
Without brackets: <class 'tuple'>
```

Accessing Tuple Items

- Tuple items are indexed
- The first item has index [0]

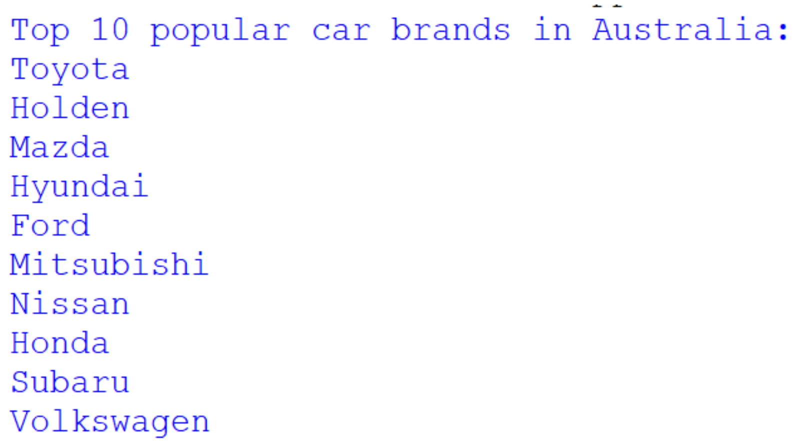
```
# Top Japanese car brands in Australia
tup_brand = ("Toyota","Mazda","Honda", "Mitsubishi")
tup_sales = (1, 2, 3, 4)
# print all brands
print("Top four Japanese brands in Australia:", tup_brand[:])
print("Brand that sells the most:",tup_sales[0], tup_brand[0])
```

```
Top four Japanese brands in Australia: ('Toyota', 'Mazda', 'Honda', 'Mitsubishi')
Brand that sells the most: 1 Toyota
```

Looping Through a Tuple using Index

- By using for loop and index

```
tup1 = ("Toyota","Holden","Mazda", "Hyundai", "Ford")
tup2 = ("Mitsubishi", "Nissan", "Honda", "Subaru", "Volkswagen")
tup_carbrands = tup1 + tup2   # concatenating tuples
print("Top 10 popular car brands in Australia:")
for i in range(len(tup_carbrands)):
  print(tup_carbrands[i])
```



```
Top 10 popular car brands in Australia:
Toyota
Holden
Mazda
Hyundai
Ford
Mitsubishi
Nissan
Honda
Subaru
Volkswagen
```

Activity 1: Looping Through a Tuple via Index

- By using while loop and index

```
tup1 = ("Toyota","Holden","Mazda", "Hyundai", "Ford")
tup2 = ("Mitsubishi", "Nissan", "Honda", "Subaru", "Volkswagen")
tup_carbrands = tup1 + tup2   # concatenating tuples
print("Top 10 popular car brands in Australia:")
# using while loop
i = 0
while i < len(?):   # your code instead of ?
  print(i+1, tup_carbrands[i])
  i = i + 1
```

```
Top 10 popular car brands in Australia:
1 Toyota
2 Holden
3 Mazda
4 Hyundai
5 Ford
6 Mitsubishi
7 Nissan
8 Honda
9 Subaru
10 Volkswagen
```

Activity 2: Iterating over items of a Tuple via for loop

```
# in the following program
# we initialize a tuple
tup1 = (1, 2, 3, 4)
# and then concatenate the tuples.
tup2 = tup1 + (5, 6)
# Finally, iterating over its elements using for Loop
for ? in tup2 :     # replace ? With your code
    print(i)
```

```
1
2
3
4
5
6
```

Tuple Methods

- index()- returns the index of the specified element (the first occurrences)
- count() – returns the occurrences of an element in the tuple

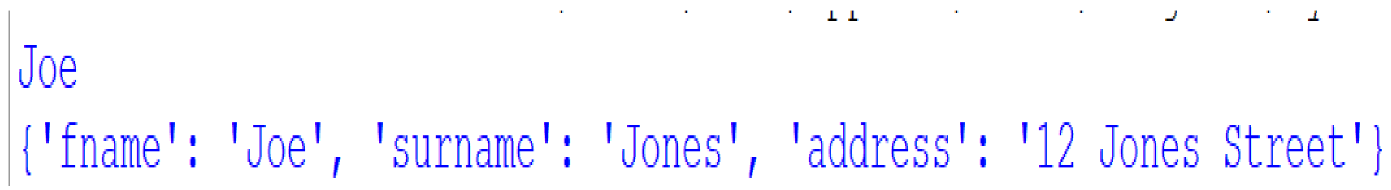
```
# initialize a tuple, note that tuples allow duplicates
tup_fruits = ("apples", "bananas", "cherries", "oranges", "bananas")
# index method
fruit = tup_fruits.index("bananas")
print("Index:", fruit)
# count method
occurrences = tup_fruits.count("bananas")
print("Occurrences:", occurrences)
```

```
Index: 1
Occurrences: 2
```

Python Dictionaries

- Dictionaries are ordered, mutable and do not allow duplicate
- Dictionaries use curly brackets { }

```
# initialize a dictionary
my_dict = {"fname": "Joe", "surname": "Jones", "address": "12 Jones Street"}
# print the first-item value
print(my_dict["fname"])
# print dictionary
print(my_dict)
```

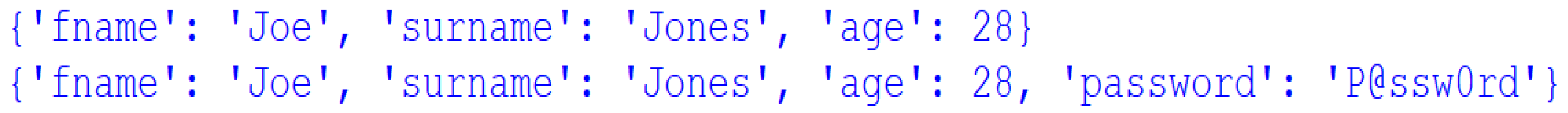


```
Joe
{'fname': 'Joe', 'surname': 'Jones', 'address': '12 Jones Street'}
```

Adding and Removing Items

- Adding a new index key and assigning a value to it
- Using the update() method

```
# initialize a dictionary
my_dict = {"fname": "Joe", "surname": "Jones"}
# adding a new item via index
my_dict["age"] = 28
print(my_dict)
# adding a new item with key: value pars
my_dict.update({"password": "P@ssw0rd"})
print(my_dict)
```



```
{'fname': 'Joe', 'surname': 'Jones', 'age': 28}
{'fname': 'Joe', 'surname': 'Jones', 'age': 28, 'password': 'P@ssw0rd'}
```

Looping Dictionaries

- Using the **key()** method to obtain the keys of a dictionary
- Using the **values()** method to obtain the values of a dictionary

```
# initialize a dictionary
my_dict = {"fname": "Joe", "surname": "Jones", "address": "12 Jones Street"}

# print all values using the keys() method
print("Keys are: ")
for i in my_dict.keys():
    print(i)
# print all values using the values() method
print("\nValues are: ")
for i in my_dict.values():
    print(i)
```

```
Keys are:
fname
surname
address

Values are:
Joe
Jones
12 Jones Street
```

Dictionary Methods

- All build-in methods (see https://www.w3schools.com/python/python_dictionaries_methods.asp)
- clear() – removes all the elements form the dictionary
- copy() – returns a copy of dictionary
- get() – returns a value of the specified key
- pop() – remove the element with the specified key

Activity 3: Python Dictionary Methods

```
my_dict = {"fname": "Joe", "surname": "Jones", "address": "12 Jones Street"}
new_dict = my_dict.copy()
print(new_dict)
# get the value of "surname" item
value = new_dict.get("surname")
print(value)
# remove the "address" item
new_dict.pop("address")
print(new_dict)
# remove all items
new_dict.clear()
print(new_dict)
```

```
{'fname': 'Joe', 'surname': 'Jones', 'address': '12 Jones Street'}
Jones
{'fname': 'Joe', 'surname': 'Jones'}
{}
```

South Metropolitan

Questions?