CSIT110 / CSIT810

Python

Lecture 11

Dr. Joseph Tonien

School of Computing and Information Technology
University of Wollongong

Objectives

Understanding of:

- Dictionary

Dictionary

used to store key-value pairs

```
dict = {}  # this is an empty dictionary

amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
} # information about a person

state_abb = {
  "NSW": "New South Wales",
  "ACT": "Australian Capital Territory",
  "NT": "Northern Territory",
  "QLD": "Queensland",
  "SA": "South Australia",
  "TAS": "Tasmania",
  "VIC": "Victoria",
  "WA": "Western Australia"
} # Australian state abbreviations
```

Dictionary

```
dict = {}

amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
}
```

using function `len` to find out how many key-value pairs:

```
dict_length = len(dict)        # → 0

amanda_length = len(amanda)    # → 3
```

Dictionary

```
dict = {}

amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
}
```

using function `print` to print out the whole dictionary

```
print(dict)

print(amanda)
```

Dictionary

```
amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
}
```

Values can be accessed using the corresponding keys

```
fname = amanda["first_name"]   # → "Amanda"

lname = amanda["last_name"]    # → "Smith"

age = amanda["age"]            # → 20


email = amanda["email"]        # → KeyError: "email"
```

Dictionary

```
amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
}
```

Another way to access the values using function `get`:

```
fname = amanda.get("first_name")   # → "Amanda"

lname = amanda.get("last_name")    # → "Smith"

age = amanda.get("age")            # → 20


email = amanda.get("email")        # → None
```

Dictionary

```
amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
}
```

Sometimes using function `get` is better because:

NO ERROR

```
email = amanda.get("email")          # → None
```

ERROR

```
email = amanda["email"]              # → KeyError: "email"
```

None

```
amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
}
```

In Python, `None` is equivalent to null in other programming languages:

```
email = amanda.get("email")          # → None
if (email is None):
  print("User has no email")
else:
  print("User email is " + email)
```

Dictionary

```
amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
}
```

We can specify a **default value** in the function `get` if the key-value pair is not found:

```
std_type = amanda.get("student_type", "N/A")   # → "N/A"

credit_point = amanda.get("credit_point", 0)   # → 0
```

Dictionary

```
amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
}
```

We can get the list of all keys:

```
all_keys = amanda.keys()
# all_keys = ["last_name", "age", "first_name"]
```

Note that the returned list of keys is in **arbitrary order**

Dictionary

```
amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
}
```

```
all_keys = amanda.keys()

for k in all_keys:
  v = amanda.get(k)
  print("key={0}, value={1}".format(k,v))
```

```
key=first_name, value=Amanda
key=age, value=20
key=last_name, value=Smith
```

Dictionary

```
amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
}
```

We can get the list of all values:

```
all_values = amanda.values()
# all_values = ["Smith", 20, "Amanda"]
```

Note that the returned list of values is also in **arbitrary order**

Dictionary

```
amanda = {
  "first_name": "Amanda",
  "last_name": "Smith",
  "age": 20
}
```

we can change the existing values:

```
amanda["first_name"] = "Mandy"
amanda["last_name"] = "Jones"
amanda["age"] = 24
```

we can add new key-value pair:

```
amanda["email"] = "Mandy.Jones@gmail.com"
```

Dictionary

```
amanda = {
  "first_name": "Mandy",
  "last_name": "Jones",
  "age": 24,
  "email": "Mandy.Jones@gmail.com"
}
```

we can delete a key-value pair:

```
del amanda["email"]
```

we can delete **all** key-value pairs, the dictionary becomes empty:

```
amanda.clear()
```

Capital cities

```
capital_city = {
"Australia": "Canberra",
"Denmark": "Copenhagen",
"Ireland": "Dublin",
"Hong Kong": "Hong Kong",
"Nepal": "Kathmandu"
}

country = input("Enter country: ")
capital = capital_city[country]
print("Capital city of {0} is {1}".format(country, capital))
```

```
Enter country: Australia
Capital city of Australia is Canberra
```

State abbreviation

```
state_abb = {
  "NSW": "New South Wales",
  "ACT": "Australian Capital Territory",
  "NT": "Northern Territory",
  "QLD": "Queensland",
  "SA": "South Australia",
  "TAS": "Tasmania",
  "VIC": "Victoria",
  "WA": "Western Australia"
}

scode = input("Enter state NSW/ACT/NT/QLD/SA/TAS/VIC/WA: ")
state = state_abb[scode]
print("The state you entered is " + state)
```

```
Enter state NSW/ACT/NT/QLD/SA/TAS/VIC/WA: NT
The state you entered is Northern Territory
```

User record maintenance

User record maintenance

```
users = {
  "matt": {"phone": "1234", "email": "ma77@code.org"},
  "joe": {"phone": "9823", "email": "mjoe@python.com"},
  "lee": {"phone": "3463", "email": "lee01@gmail.com"}
}
```

We want to build an application to maintain user information, which, for simplicity, consists of **username**, **phone** and **email**.

All user information will be stored in a dictionary called `users` where the key is the **username**, for example:

```
"matt": matt_info
"joe": joe_info
"lee": lee_info
```

User record maintenance

```
users = {
  "matt": {"phone": "1234", "email": "ma77@code.org"},
  "joe": {"phone": "9823", "email": "mjoe@python.com"},
  "lee": {"phone": "3463", "email": "lee01@gmail.com"}
}
```

```
"matt": matt_info
"joe": joe_info
"lee": lee_info
```

Each user info itself is also a dictionary with two keys: **phone** and **email**, for example:

```
matt_info = {"phone": "1234", "email": "ma77@code.org"}
joe_info = {"phone": "9823", "email": "mjoe@python.com"}
lee_info = {"phone": "3463", "email": "lee01@gmail.com"}
```

User record maintenance

```
users = {}   # initially, no users

# this function prints the menu,
# asks user for an option,
# and returns the option
def get_option():
  print()
  print("User record maintenance")
  print("V > View")
  print("A > Add")
  print("U > Update")
  print("D > Delete")
  print("Q > Quit")

  option = input("Enter option: ")
  return option
```

User record maintenance

```
# the main program
while True:
  option = get_option()

  if (option == "V"):
    view()
  elif (option == "A"):
    add()
  elif (option == "U"):
    update()
  elif (option == "D"):
    delete()
  elif (option == "Q"):
    break
  else:
    print("Invalid input")
```

User record maintenance

```
def add():
  username = input("Enter username: ")
  phone = input("Enter phone: ")
  email = input("Enter email: ")

  # create a user detail using dictionary
  user_info = {"phone": phone, "email": email}

  # add this user info to the user dictionary
  # using username as the key
  users[username] = user_info
```

```
Enter username: matt
Enter phone: 1234
Enter email: ma77@code.org

user_info = {"phone": "1234", "email": "ma77@code.org"}
users["matt"] = user_info
users = {
  "matt": {"phone": "1234", "email": "ma77@code.org"}
}
```

User record maintenance

```
def view():
  username = input("Enter username: ")

  # retrieve the user info
  user_info = users.get(username)
  if (user_info is None):
    print("Record not found")
  else:
    print("Username: " + username)
    print("Phone: " + user_info["phone"])
    print("Email: " + user_info["email"])
```

```
Enter username: matt

Username: matt
Phone: 1234
Email: ma77@code.org
```

User record maintenance

```
def update():
  username = input("Enter username: ")
  phone = input("Enter phone: ")
  email = input("Enter email: ")

  # retrieve the user info
  user_info = users.get(username)
  if (user_info is None):
    print("Record not found")
  else:
    # update the record
    user_info["phone"] = phone
    user_info["email"] = email
```

User record maintenance

```
def delete():
  username = input("Enter username: ")

  # retrieve the user info
  user_info = users.get(username)
  if (user_info is None):
    print("Record not found")
  else:
    # delete the record
    del users[number]
```