API and Python training

Session 5

This session agenda

- If-else, if – elif -else
- While
- For
- Using control structures for handling API responses
- Demo
- Handling exceptions
- Demo

---- Next session – part 2 ------

- Python functions
- Demo

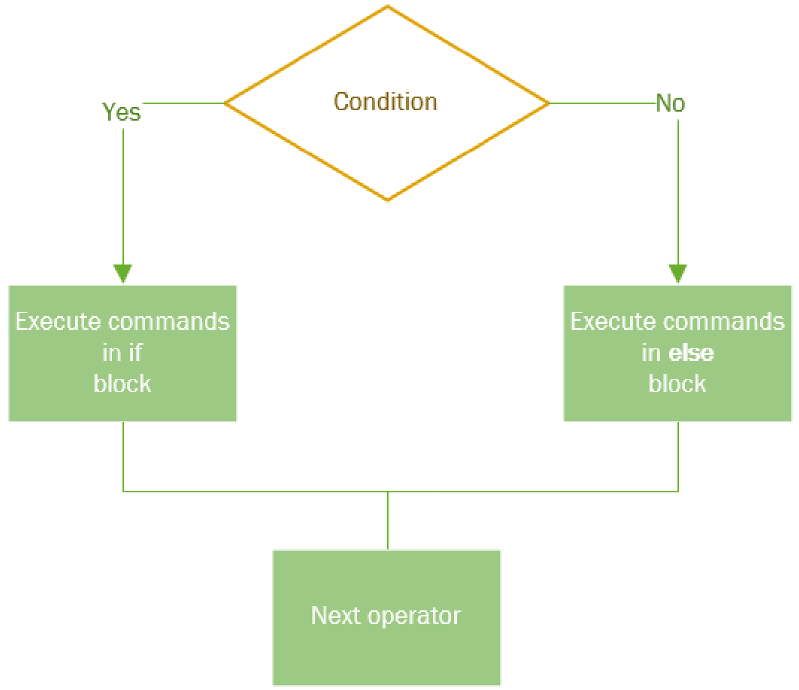
If … else or if … elif .. else

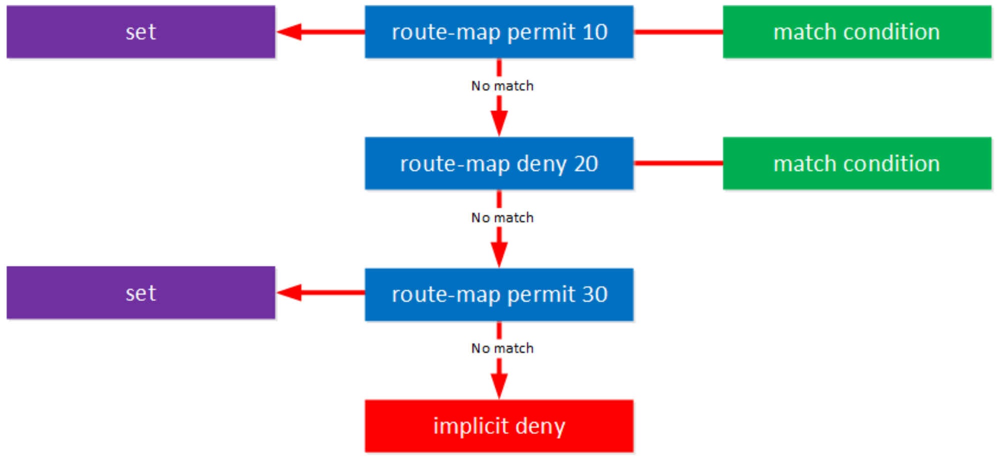
- Note the colon at the end of the string
- **Else can be omitted**, so may have a single **if**

```
if <condition>:
    <block of code to run if the condition is True>
else:
    <block of code to run otherwise>
```

- **If-elif** if top condition matches, no further conditions will be processed
- Similar logic to route-map or access-lists ------------------->

```
if <condition-1>:
    <block of code to run if the condition-1 is True>
elif <condition-2>:
     <block of code to run if the condition-2 is True>
else:
    <block to run if no previous conditions matched>
```


Condition
Yes
No
Execute commands in if block
Execute commands in **else** block
Next operator



https://networklessons.com/cisco/ccnp-route/introduction-to-route-maps

Conditions

- Can be different, most common are comparison - always answers – **True of False**
- Conditions **depends on data type**, examples:

- Integers

  if a == 10.8:

  if response_code != 200:

  if my_int_variable > 0:

- Strings:

  if my_string == test_string: exact match

  if my_string in test_sting: substring match example: *if 'e' in 'hello'*

- Lists:

  if <element> in <list> example: *if 'red' in ['white', 'red', 'blue']:*

- Can be multiple conditions:

  if response_code == 200 ***and*** *my_string == (test_string* ***or*** *another_string):*

- Not necessary to compare something with something, but **use 'shortcuts'**

  if my_string: - means if it's not empty (string or list) or not 0 (integer)

  if got_response: - the same as *got_response == True:*

- You can use **not**:

  if not a: *- means a is not empty (string) or nor 0 (integer)*

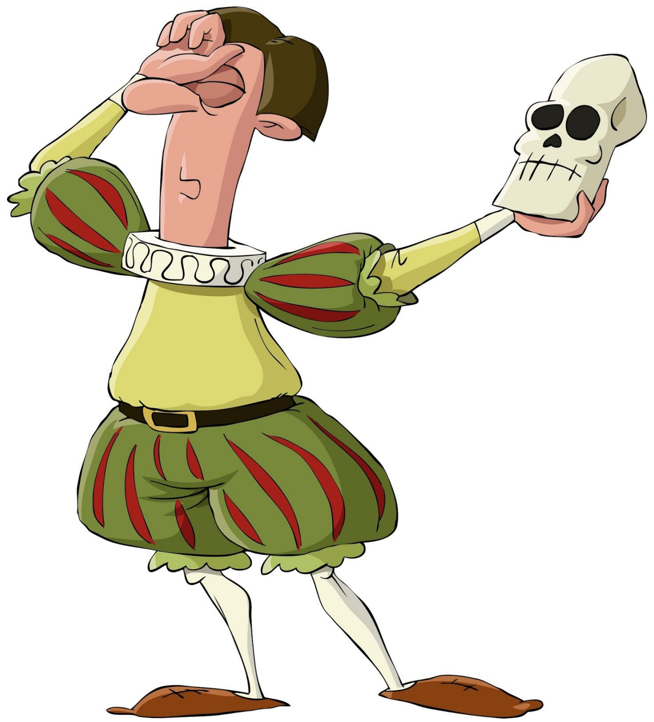
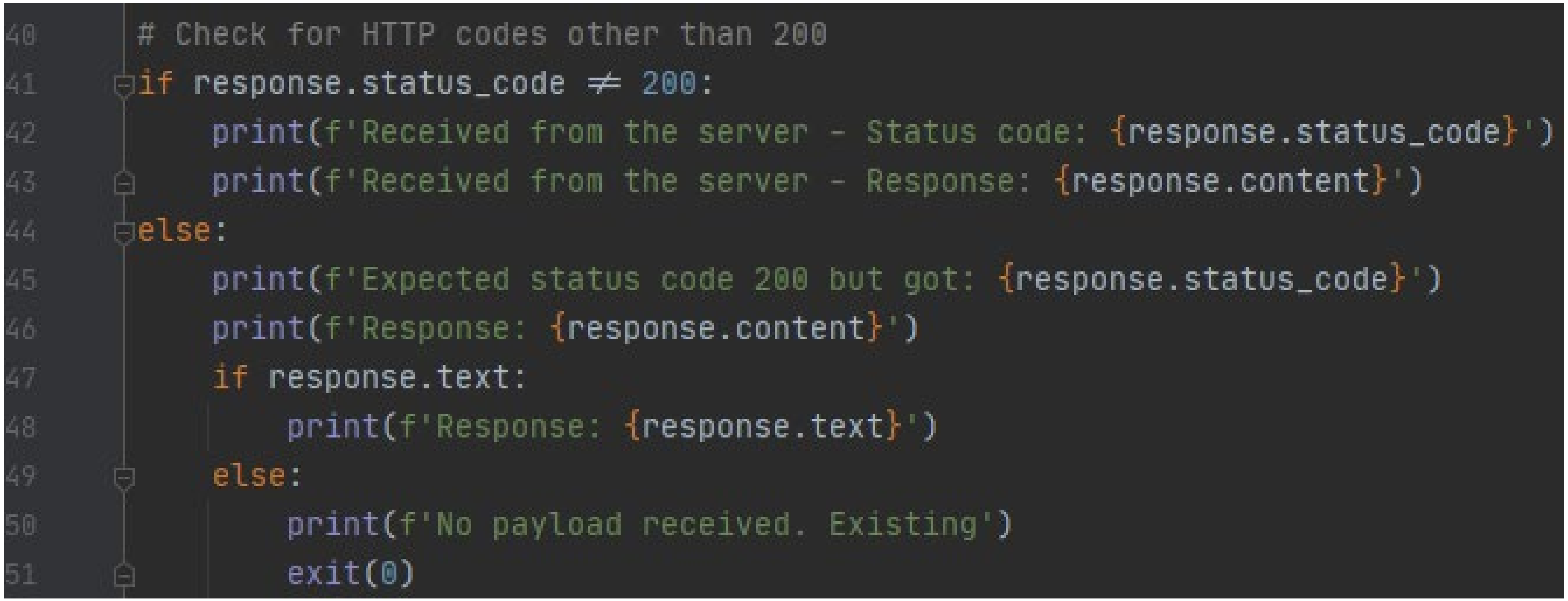


Image :http://queenelizabethsfantasy.blogspot.com/2014/11/to-be-or-not-to-be.html

Indentation

- Indicates where the block of code begins and ends. In C and Java-like languages you may see { }
- You need to use the same number of spaces in the same block of code (Line 44, 45 and 47)
- Any number of spaces as long as you follow previous rule, but recommended is 4
- Don’t mix Tab and Spaces if you use a text editor or CLI, but IDE allows to do so and will convert Tab into spaces



```
40  # Check for HTTP codes other than 200
41  if response.status_code ≠ 200:
42      print(f'Received from the server - Status code: {response.status_code}')
43      print(f'Received from the server - Response: {response.content}')
44  else:
45      print(f'Expected status code 200 but got: {response.status_code}')
46      print(f'Response: {response.content}')
47      if response.text:
48          print(f'Response: {response.text}')
49      else:
50          print(f'No payload received. Existing')
51          exit(0)
```

https://www.geeksforgeeks.org/indentation-in-python/

https://www.w3schools.com/python/gloss_python_indentation.asp

For loop

- Syntax:

  for <iterator-variable> in <list of another variables>:

      <block of code>

- For-loops in Lists

  list_of_incidents = ['INC34325', 'INC4545', 'INC4543534']

  for incident in list_of_incidents:

      print(incident)

- For-loops in Dictionaries:

  host_properties = {'hostname': 'linux1', 'ipv4':'10.2.2.3', 'OS':'Ubuntu'}

  for property in host_properties:

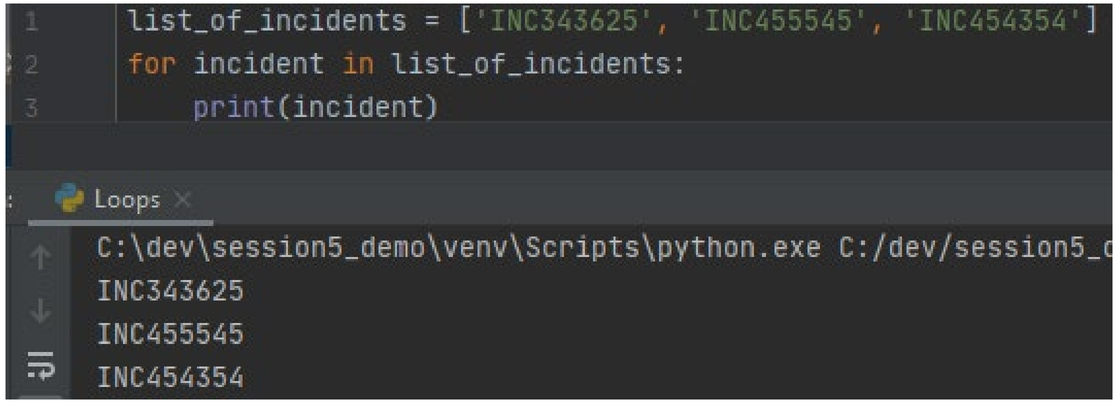
      print(f' Key:{property}, Value: {host_properties[property]}')

- Note you don't have to define variable *<iterator-variable>* in advance but you need to have list or dictionary already defined

- List comprehension – another powerful Python 'shortcut'

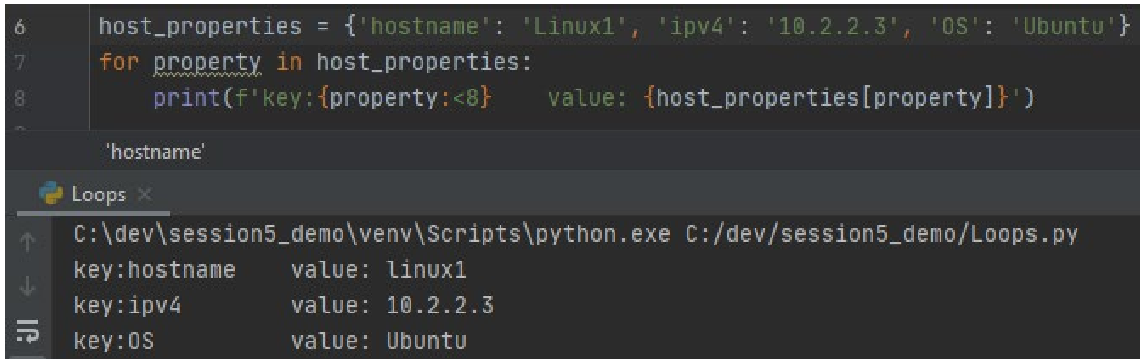
https://www.programiz.com/python-programming/list-comprehension

Not necessary a list but any interable object, see https://realpython.com/python-for-loop/



```
list_of_incidents = ['INC343625', 'INC455545', 'INC454354']
for incident in list_of_incidents:
    print(incident)
```

```
C:\dev\session5_demo\venv\Scripts\python.exe C:/dev/session5_d
INC343625
INC455545
INC454354
```



```
host_properties = {'hostname': 'Linux1', 'ipv4': '10.2.2.3', 'OS': 'Ubuntu'}
for property in host_properties:
    print(f'key:{property:<8}    value: {host_properties[property]}')
```

```
C:\dev\session5_demo\venv\Scripts\python.exe C:/dev/session5_demo/Loops.py
key:hostname    value: linux1
key:ipv4        value: 10.2.2.3
key:OS          value: Ubuntu
```

While loop

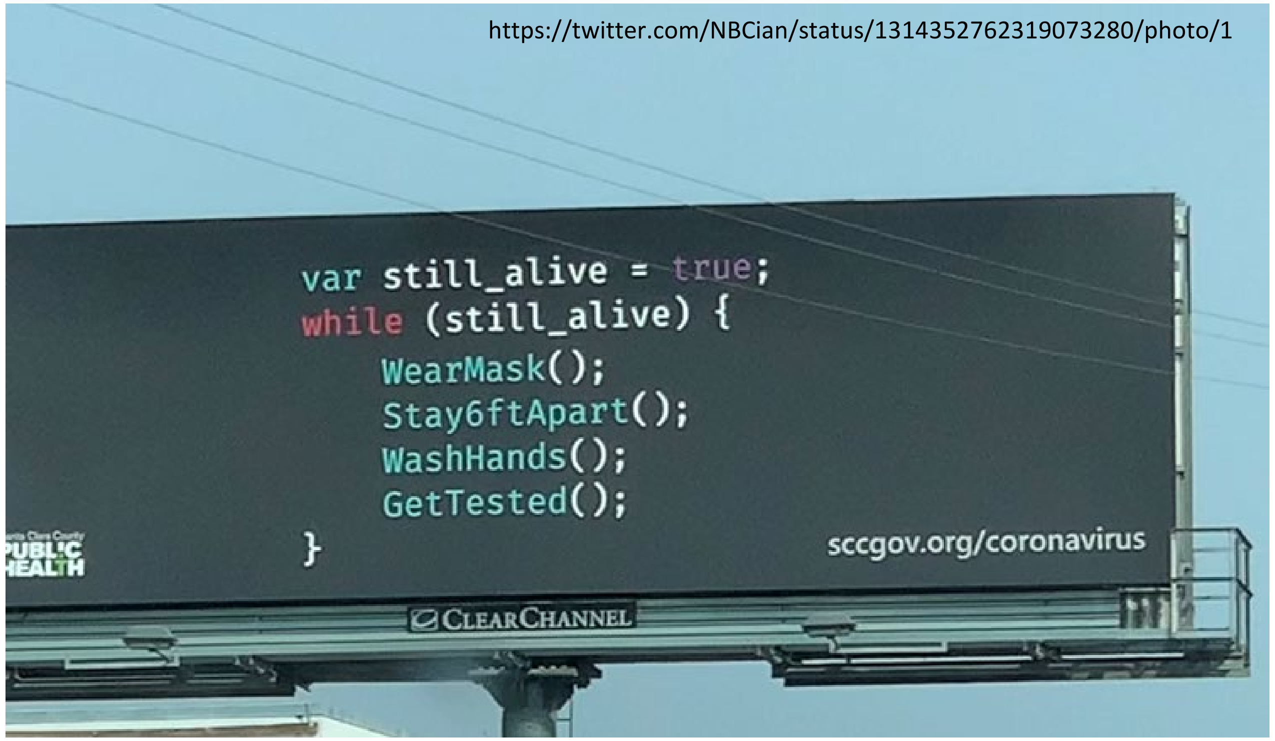
- Syntax:
  variable = <something>
  while <condition is True>:
  <block of code>

- The <block of code> will run as many times and as long as <condition> remains True

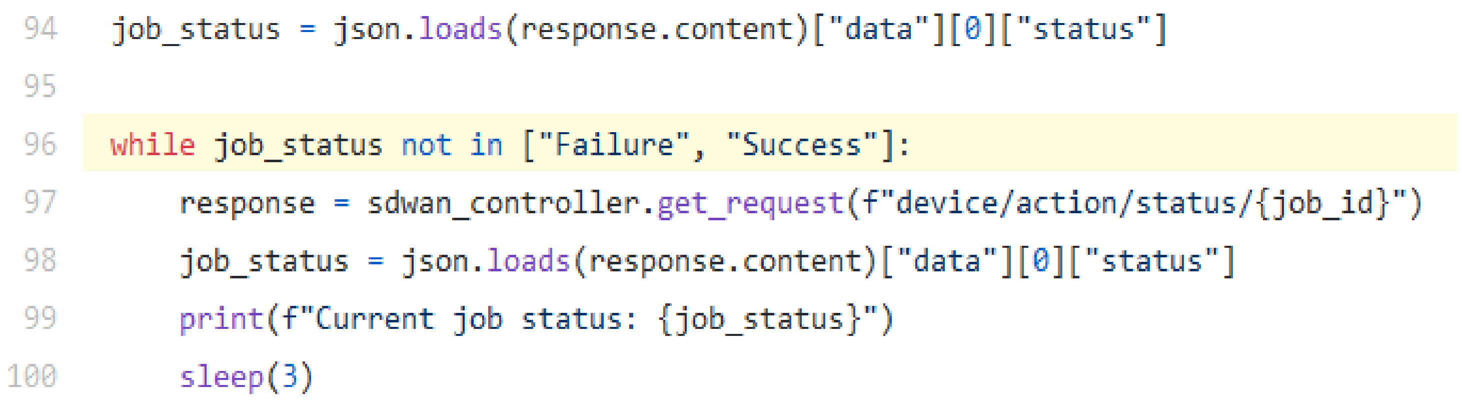
- Make sure <condition> can change within the loop to avoid indefinitely loop, or you can use *break* statement inside the loop

- Condition – the same as in if-else
  while a > 10:
  while <string> in <List>:
  while response:

- Define the <variable> first to check in the condition


https://twitter.com/NBCian/status/1314352762319073280/photo/1

```
var still_alive = true;
while (still_alive) {
    WearMask();
    Stay6ftApart();
    WashHands();
    GetTested();
}
```



```
job_status = json.loads(response.content)["data"][0]["status"]

while job_status not in ["Failure", "Success"]:
    response = sdwan_controller.get_request(f"device/action/status/{job_id}")
    job_status = json.loads(response.content)["data"][0]["status"]
    print(f"Current job status: {job_status}")
    sleep(3)
```

https://github.com/supro200/sdwan-auto-upgrade/blob/master/main.py

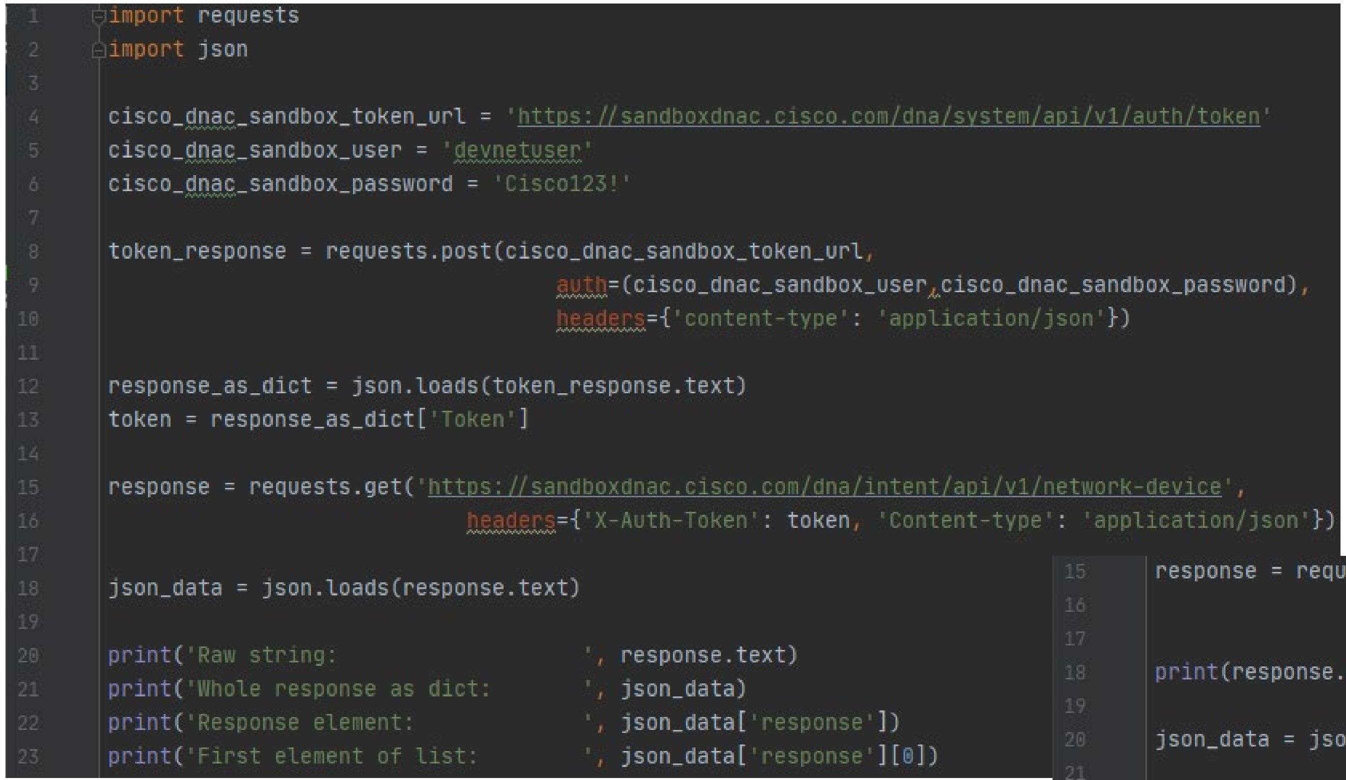
Using control structures while handling API responses

Typical use cases

- Check status code
- Check if response contains payload
- Check if a key returned in the response (such as ‘result’)
- Check if some keyword exist in the response
- Iterate though the list returned from the server
- While loops can used to poll server for a status

Demo

Demo 1 – adding checking response code and iterating through a list

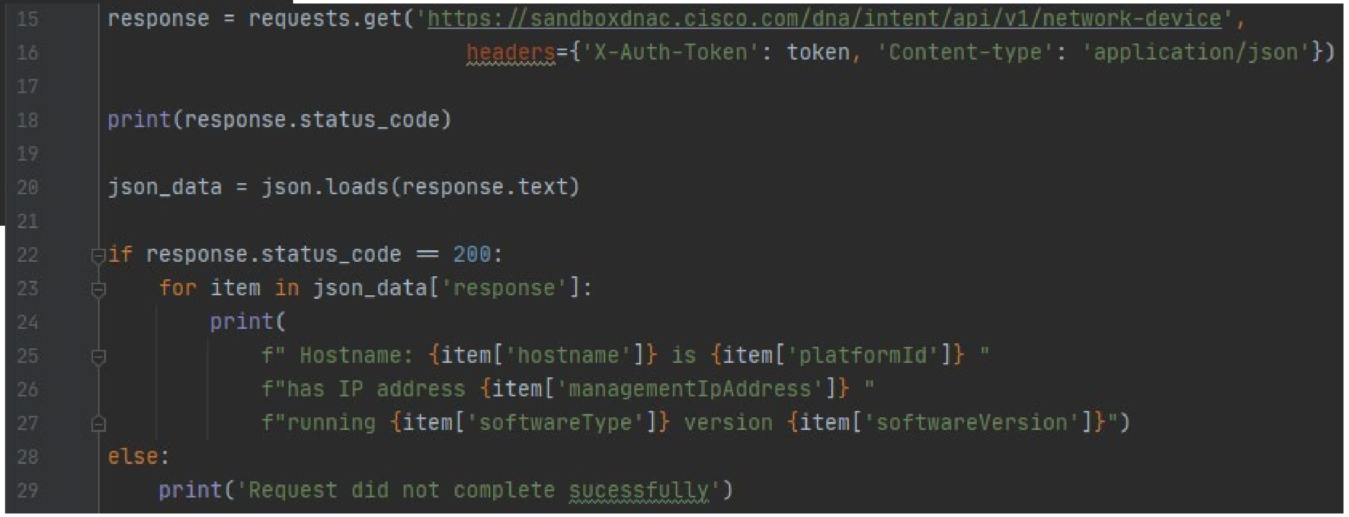


```
1   import requests
2   import json
3
4   cisco_dnac_sandbox_token_url = 'https://sandboxdnac.cisco.com/dna/system/api/v1/auth/token'
5   cisco_dnac_sandbox_user = 'devnetuser'
6   cisco_dnac_sandbox_password = 'Cisco123!'
7
8   token_response = requests.post(cisco_dnac_sandbox_token_url,
9                                  auth=(cisco_dnac_sandbox_user,cisco_dnac_sandbox_password),
10                                 headers={'content-type': 'application/json'})
11
12  response_as_dict = json.loads(token_response.text)
13  token = response_as_dict['Token']
14
15  response = requests.get('https://sandboxdnac.cisco.com/dna/intent/api/v1/network-device',
16                          headers={'X-Auth-Token': token, 'Content-type': 'application/json'})
17
18  json_data = json.loads(response.text)
19
20  print('Raw string:                 ', response.text)
21  print('Whole response as dict:     ', json_data)
22  print('Response element:           ', json_data['response'])
23  print('First element of list:      ', json_data['response'][0])
```

<---- Code from Session 4

Changed code below

- Added line 22 – checking response code
- If response code is 200 – iterate through the list
- Otherwise print error message



```
15  response = requests.get('https://sandboxdnac.cisco.com/dna/intent/api/v1/network-device',
16                          headers={'X-Auth-Token': token, 'Content-type': 'application/json'})
17
18  print(response.status_code)
19
20  json_data = json.loads(response.text)
21
22  if response.status_code == 200:
23      for item in json_data['response']:
24          print(
25              f" Hostname: {item['hostname']} is {item['platformId']} "
26              f"has IP address {item['managementIpAddress']} "
27              f"running {item['softwareType']} version {item['softwareVersion']}")
28  else:
29      print('Request did not complete sucessfully')
```

Exceptions

- If there are errors during command execution there will be an exception
- Python (and other languages) will give a detailed dump and terminates the module
- If you anticipate there could be errors, put the fragment of the code in

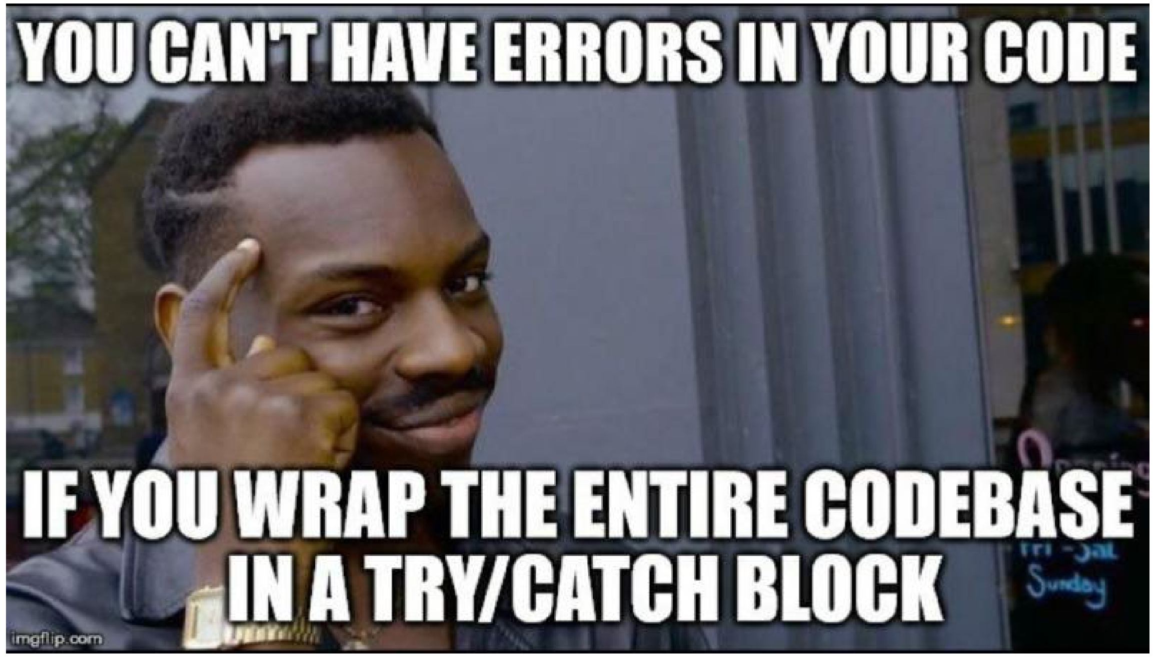
```
try:
   <dangerous fragment of the code>
except <error1>:
    <how you handle this error>
except <error2>:
    <how you handle this error>
else:
     <no error – normal execution>
```

- else is optional
- You can handle different errors in different except statements
- Try to avoid putting too much code into try-block and or catching all errors, unless you provide your users with a message 'Something went wrong'

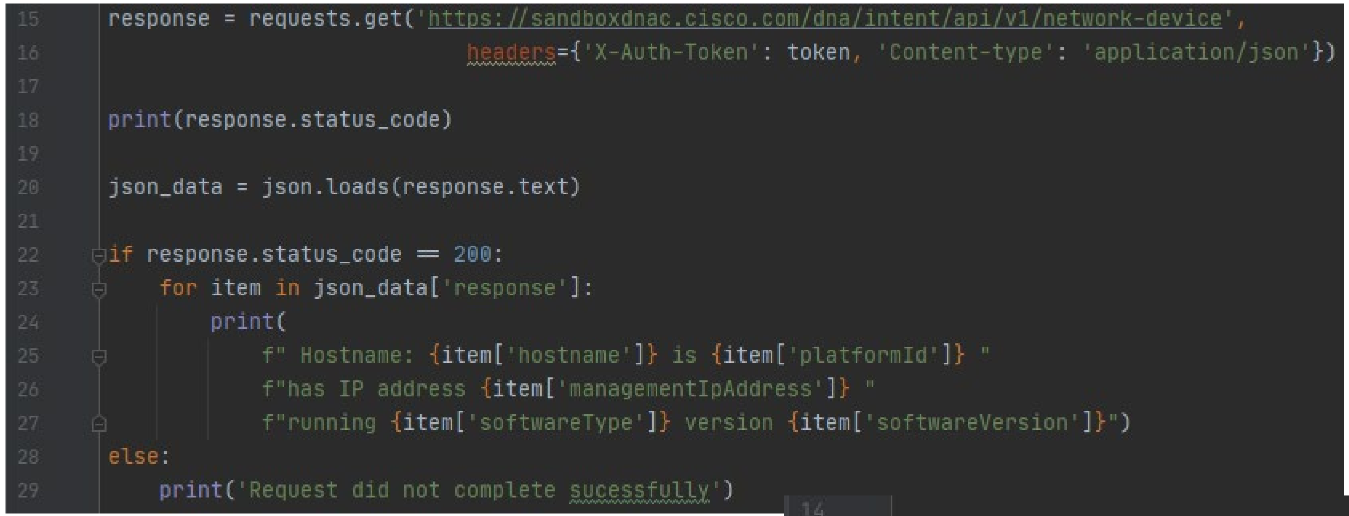
This is a joke ---------->

- Good reference: https://stackoverflow.com/questions/16511337/correct-way-to-try-except-using-python-requests-module
- Example on handling ssh errors: https://github.com/supro200/fw-rule-helper/blob/master/fwhelper/common/network_helpers.py#L55

Demo


YOU CAN'T HAVE ERRORS IN YOUR CODE
IF YOU WRAP THE ENTIRE CODEBASE IN A TRY/CATCH BLOCK
imgflip.com

Demo 2 – adding exception handling – exit

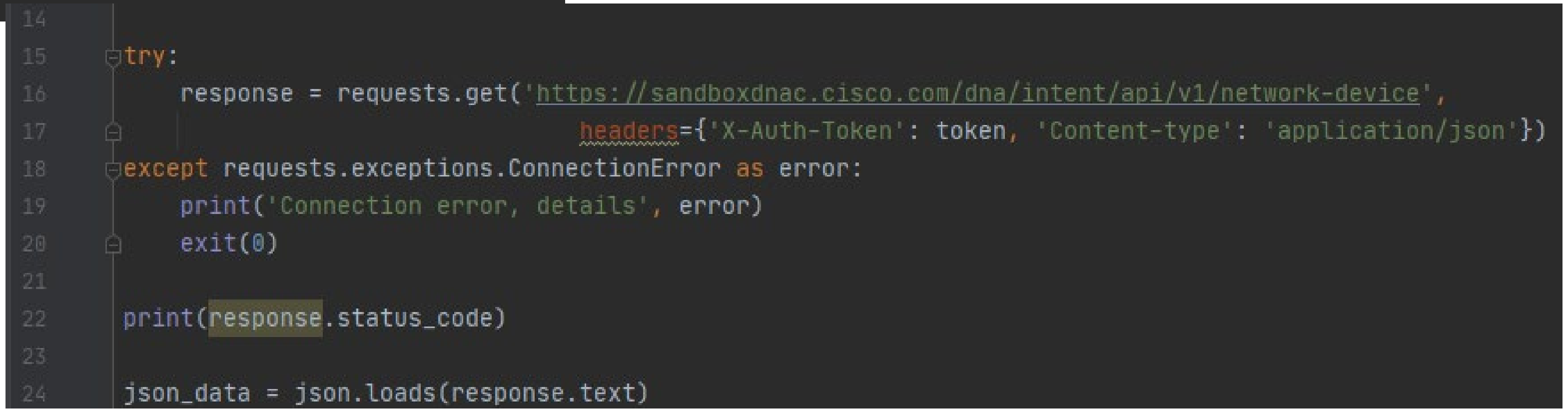


```
15  response = requests.get('https://sandboxdnac.cisco.com/dna/intent/api/v1/network-device',
16                          headers={'X-Auth-Token': token, 'Content-type': 'application/json'})
17
18  print(response.status_code)
19
20  json_data = json.loads(response.text)
21
22  if response.status_code == 200:
23      for item in json_data['response']:
24          print(
25              f" Hostname: {item['hostname']} is {item['platformId']} "
26              f"has IP address {item['managementIpAddress']} "
27              f"running {item['softwareType']} version {item['softwareVersion']}")
28  else:
29      print('Request did not complete sucessfully')
```

<---- Code from previous demo

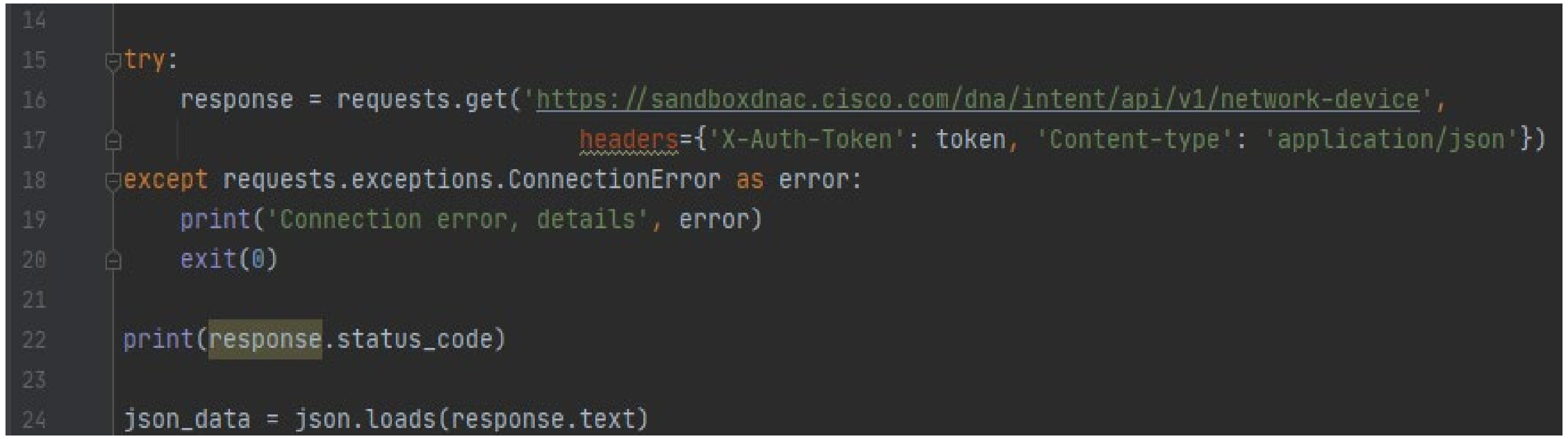
Changed code below

- Original line 15 is in try – except block
- If connection times out – exit the program



```
14
15  try:
16      response = requests.get('https://sandboxdnac.cisco.com/dna/intent/api/v1/network-device',
17                              headers={'X-Auth-Token': token, 'Content-type': 'application/json'})
18  except requests.exceptions.ConnectionError as error:
19      print('Connection error, details', error)
20      exit(0)
21
22  print(response.status_code)
23
24  json_data = json.loads(response.text)
```

Demo 3 – adding exception handling – try-except-else

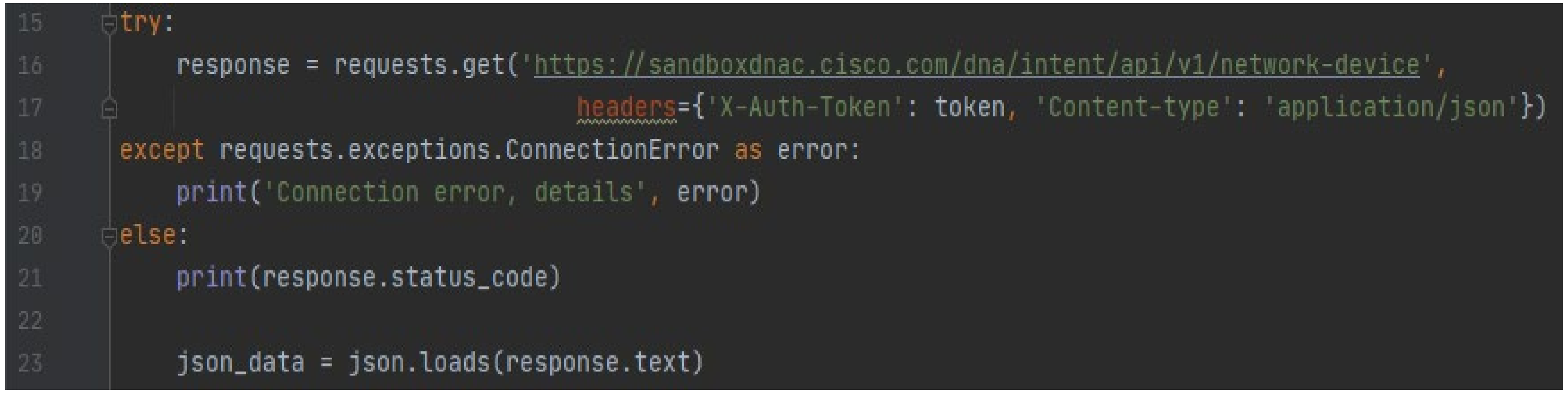


```
try:
    response = requests.get('https://sandboxdnac.cisco.com/dna/intent/api/v1/network-device',
                            headers={'X-Auth-Token': token, 'Content-type': 'application/json'})
except requests.exceptions.ConnectionError as error:
    print('Connection error, details', error)
    exit(0)

print(response.status_code)

json_data = json.loads(response.text)
```

<---- Code from previous demo

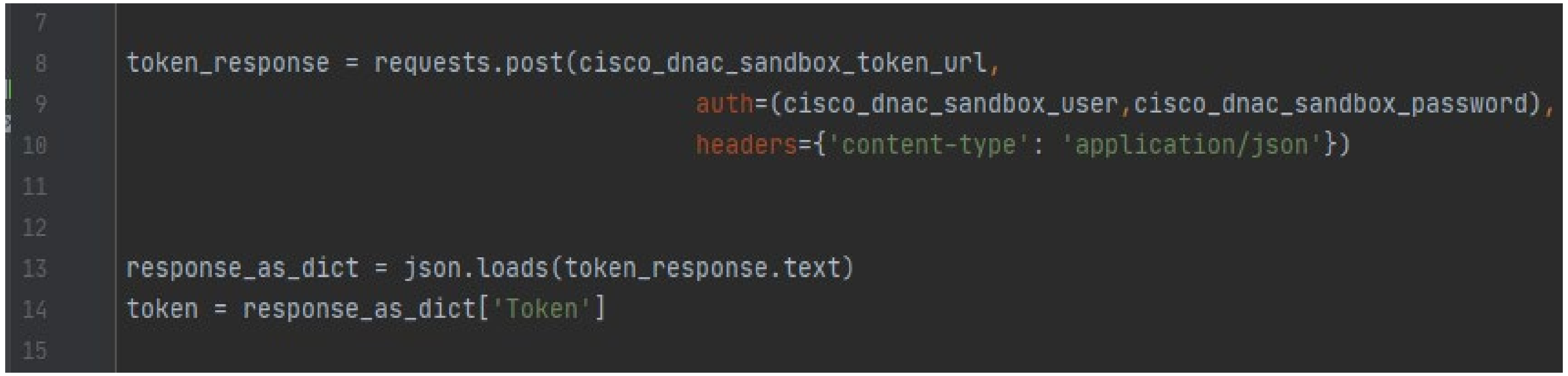
Changed code below

- Instead of exit, print error
- Added else statement – run block of code if there were no exceptions



```
try:
    response = requests.get('https://sandboxdnac.cisco.com/dna/intent/api/v1/network-device',
                            headers={'X-Auth-Token': token, 'Content-type': 'application/json'})
except requests.exceptions.ConnectionError as error:
    print('Connection error, details', error)
else:
    print(response.status_code)

    json_data = json.loads(response.text)
```

Demo 4 – getting token - add exception handling and status code checking

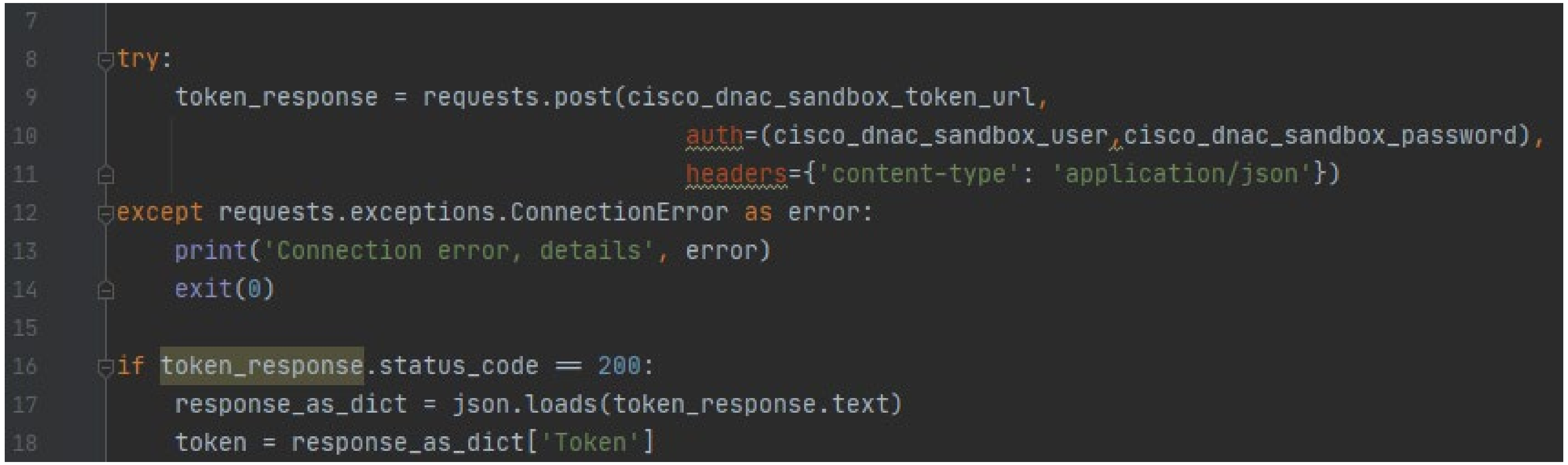


```
 7
 8  token_response = requests.post(cisco_dnac_sandbox_token_url,
 9                                 auth=(cisco_dnac_sandbox_user,cisco_dnac_sandbox_password),
10                                 headers={'content-type': 'application/json'})
11
12
13  response_as_dict = json.loads(token_response.text)
14  token = response_as_dict['Token']
15
```

<---- Original code

Changed code below

- Line 8 – put getting token into try-except statement
- Line 16 – checking return code



```
 7
 8  try:
 9      token_response = requests.post(cisco_dnac_sandbox_token_url,
10                                     auth=(cisco_dnac_sandbox_user,cisco_dnac_sandbox_password),
11                                     headers={'content-type': 'application/json'})
12  except requests.exceptions.ConnectionError as error:
13      print('Connection error, details', error)
14      exit(0)
15
16  if token_response.status_code == 200:
17      response_as_dict = json.loads(token_response.text)
18      token = response_as_dict['Token']
```

Summary and next steps

- **Summary**

Python – control structures – if, for, while

Handling exceptions

- **Next time this session will continue**

Python functions