

BÁO CÁO BÀI TẬP

Môn học: NT521.O11.ANTN

LAB1: AUTOMATING EVERYTHING AS CODE

Git & Testing

GVHD: Đỗ Thị Thu Hiền

1.

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT521.O11.ANTN

THÔNG TIN CHUNG:

STT	Họ và tên	MSSV	Email
1	Nguyễn Văn Khang Kim	21520314	21520314@gm.uit.edu.vn
2	Lưu Gia Huy	21520916	21520916@gm.uit.edu.vn
3	Nguyễn Vũ Anh Duy	21520211	21520211@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:¹

STT	Công việc	Kết quả tự đánh giá
1		100%
2		100%
3		100%
4		100%
5		100%

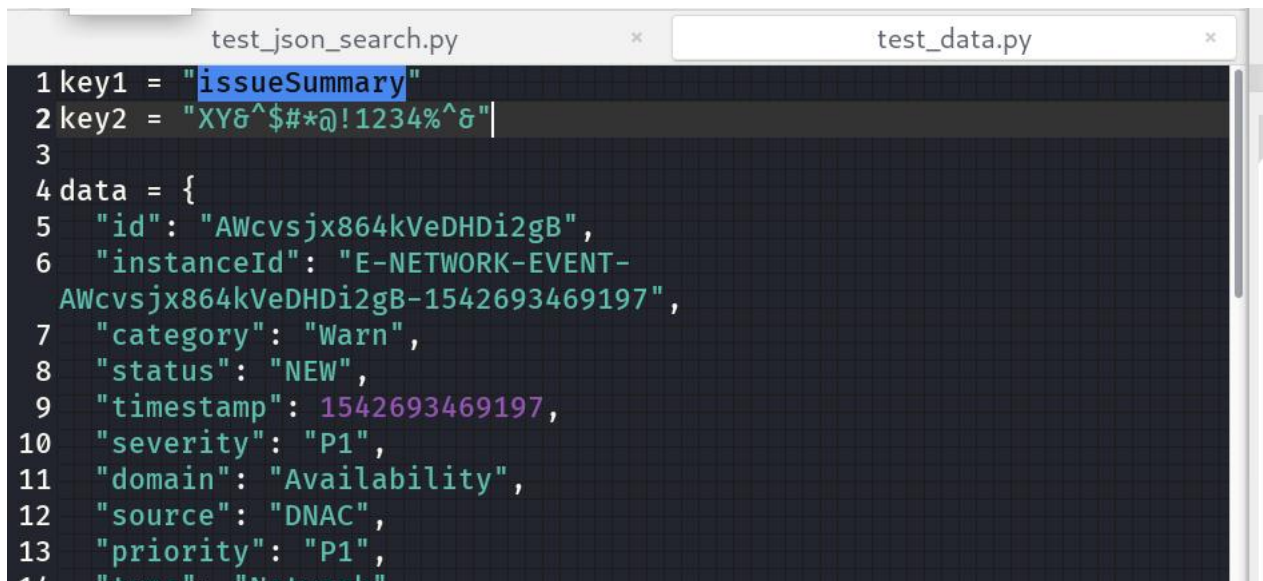
Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

¹ Ghi nội dung công việc, các kịch bản trong bài Thực hành

BÁO CÁO CHI TIẾT

B.2 Test Python Function với unittest

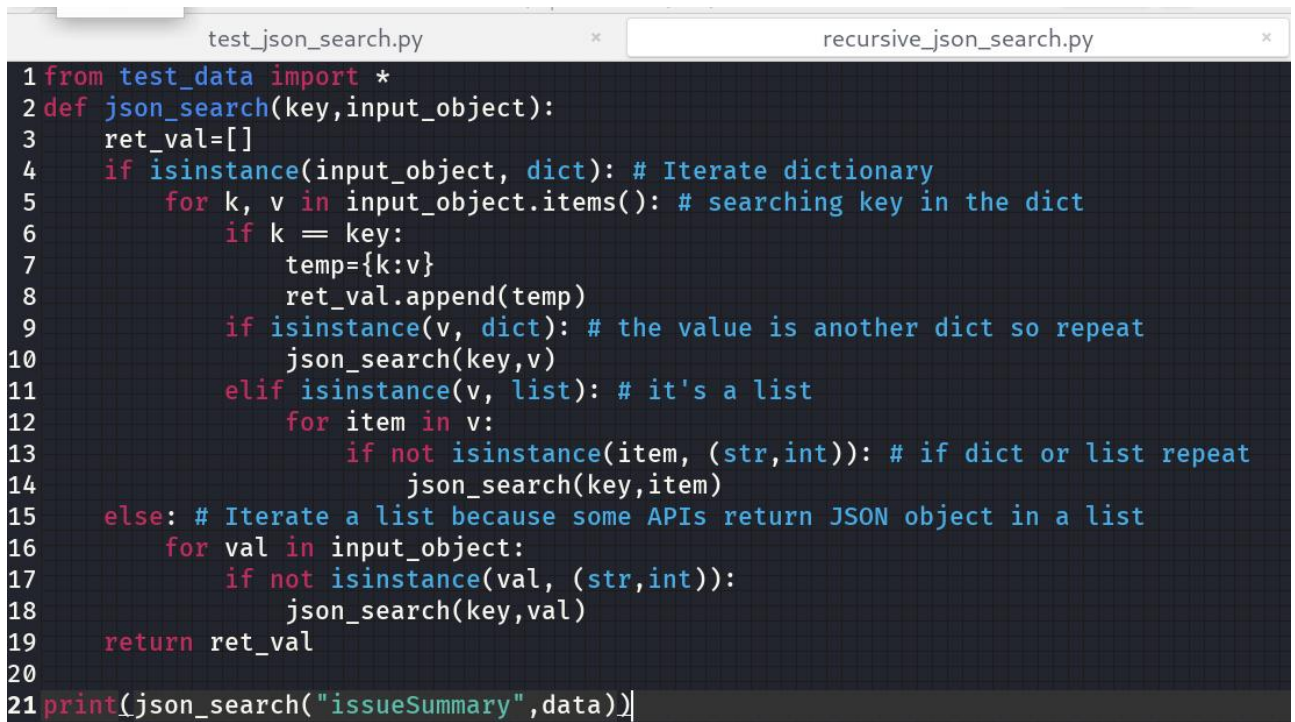
Bước 1. Xem tập tin test_data.py



```

test_json_search.py × test_data.py ×
1 key1 = "issueSummary"
2 key2 = "XY&^$#@!1234%^&|"
3
4 data = {
5     "id": "AWcvsjx864kVeDHDi2gB",
6     "instanceId": "E-NETWORK-EVENT-
    AWcvsjx864kVeDHDi2gB-1542693469197",
7     "category": "Warn",
8     "status": "NEW",
9     "timestamp": 1542693469197,
10    "severity": "P1",
11    "domain": "Availability",
12    "source": "DNAC",
13    "priority": "P1",
14    "type": "Network"
15 }
  
```

Bước 2. Tạo hàm json_search() mà ta sẽ kiểm tra.



```

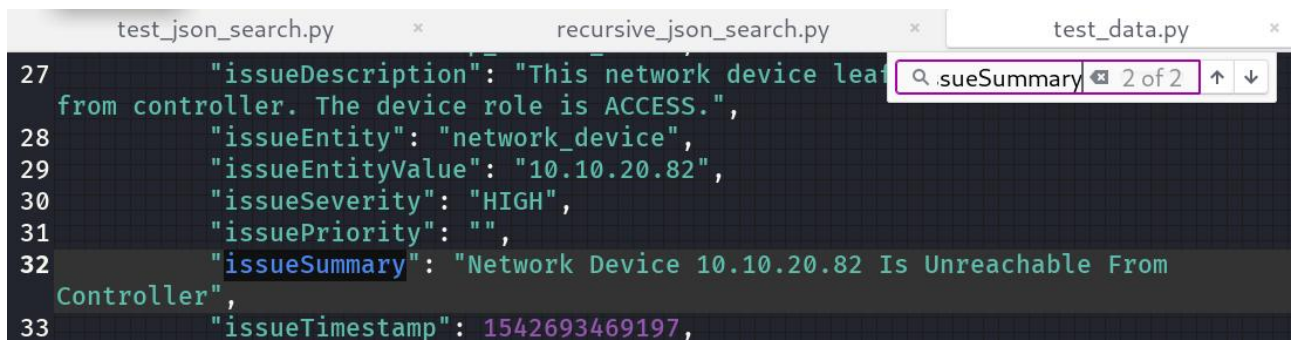
test_json_search.py × recursive_json_search.py ×
1 from test_data import *
2 def json_search(key,input_object):
3     ret_val=[]
4     if isinstance(input_object, dict): # Iterate dictionary
5         for k, v in input_object.items(): # searching key in the dict
6             if k == key:
7                 temp={k:v}
8                 ret_val.append(temp)
9             if isinstance(v, dict): # the value is another dict so repeat
10                json_search(key,v)
11            elif isinstance(v, list): # it's a list
12                for item in v:
13                    if not isinstance(item, (str,int)): # if dict or list repeat
14                        json_search(key,item)
15            else: # Iterate a list because some APIs return JSON object in a list
16                for val in input_object:
17                    if not isinstance(val, (str,int)):
18                        json_search(key,val)
19            return ret_val
20
21 print(json_search("issueSummary",data))
  
```

Bước 3. Lưu và chạy code trên.

```
(kali㉿kali)-[~/laptrinhantoan/lab1/unittest]
$ python3 recursive_json_search.py
[]

(kali㉿kali)-[~/laptrinhantoan/lab1/unittest]
$
```

Bước 4. Kiểm tra lại kết quả của hàm json_search() bằng cách mở tập tin test_data.py và tìm kiếm “issueSummary”. Dù key này có tồn tại trong data nhưng hàm không tìm thấy.



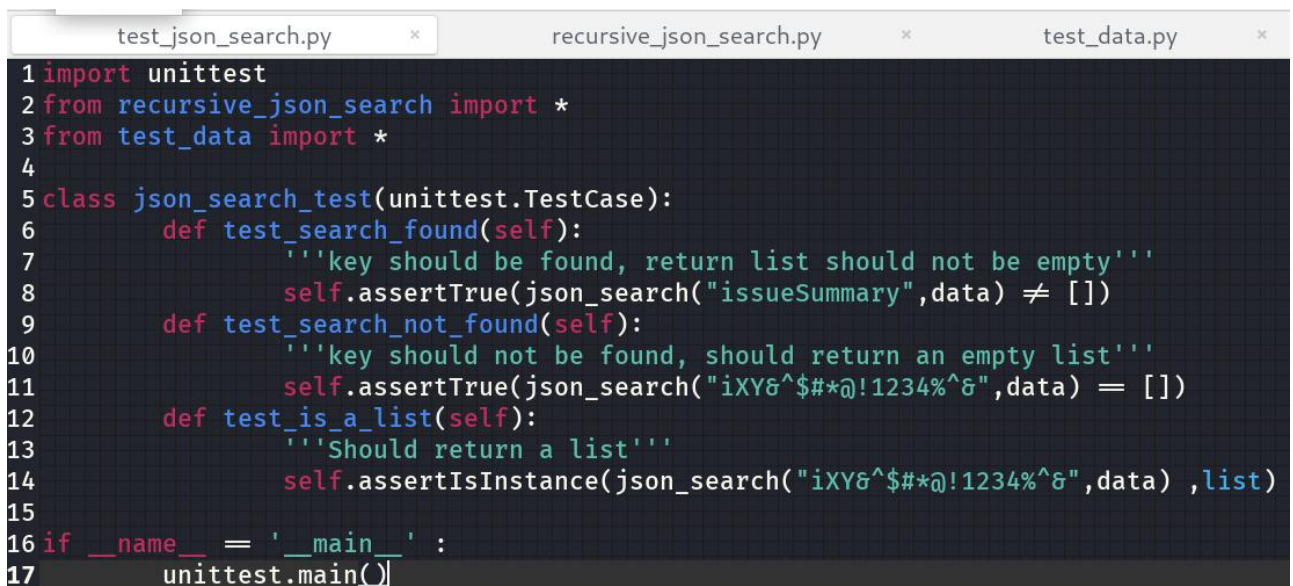
```
test_json_search.py x recursive_json_search.py x test_data.py x
27 "issueDescription": "This network device lea
from controller. The device role is ACCESS.",
28 "issueEntity": "network_device",
29 "issueEntityValue": "10.10.20.82",
30 "issueSeverity": "HIGH",
31 "issuePriority": "",
32 "issueSummary": "Network Device 10.10.20.82 Is Unreachable From
Controller",
33 "issueTimestamp": 1542693469197,
```

==> code lỗi.

Bước 5. Tạo một unit test để kiểm tra function có hoạt động đúng như dự kiến.

- Hàm json_search sẽ được kiểm tra qua 3 test case.

1. Dùng key “issueSummary” có sẵn trong JSON object, xem đoạn code có thể tìm thấy key như vậy không (đúng nếu như list trả về khác rỗng).
2. Dùng key "XY&^\$#@!1234%^&" không có sẵn trong JSON object, xem đoạn code có thể tìm thấy key như vậy không (đúng nếu như list trả về rỗng).
3. Kiểm tra xem hàm có trả về 1 list hay không (đúng nếu như kết quả trả về là 1 list).



```
test_json_search.py x recursive_json_search.py x test_data.py x
1 import unittest
2 from recursive_json_search import *
3 from test_data import *
4
5 class json_search_test(unittest.TestCase):
6     def test_search_found(self):
7         '''key should be found, return list should not be empty'''
8         self.assertTrue(json_search("issueSummary",data) != [])
9     def test_search_not_found(self):
10        '''key should not be found, should return an empty list'''
11        self.assertTrue(json_search("iXY&^$#@!1234%^&",data) == [])
12    def test_is_a_list(self):
13        '''Should return a list'''
14        self.assertIsInstance(json_search("iXY&^$#@!1234%^&",data) ,list)
15
16 if __name__ == '__main__':
17     unittest.main()
```


Bước 6. Chạy test để xem kết quả.

- Chạy code test.

- Quan sát kết quả: Đầu tiên ta thấy list trống, thứ 2 ta thấy **.F.** . Dấu chấm có nghĩa test passed và điểm F có nghĩa là test failed. Do đó, phép thử thứ nhất đã passed, phép thử thứ hai failed, và phép thử thứ ba cũng passed.

```
(kali㉿kali)-[~/laptrinhantoan/lab1/unittest]
$ python3 test_json_search.py
[]
.F.

FAIL: test_search_found (__main__.json_search_test.test_search_found)
key should be found, return list should not be empty

Traceback (most recent call last):
  File "/home/kali/laptrinhantoan/lab1/unittest/test_json_search.py", line 8, in test_search_found
    self.assertTrue(json_search("issueSummary",data) != [])
AssertionError: False is not true

Ran 3 tests in 0.001s

FAILED (failures=1)

(kali㉿kali)-[~/laptrinhantoan/lab1/unittest]
```

- Để liệt kê từng bài test và kết quả của nó, thêm tùy chọn -v trong lệnh unittest.

```
(kali㉿kali)-[~/laptrinhantoan/lab1/unittest]
$ python3 -m unittest -v test_json_search.py
[]
test_is_a_list (test_json_search.json_search_test.test_is_a_list)
Should return a list ... ok
test_search_found (test_json_search.json_search_test.test_search_found)
key should be found, return list should not be empty ... FAIL
test_search_not_found (test_json_search.json_search_test.test_search_not_found)
key should not be found, should return an empty list ... ok

FAIL: test_search_found (test_json_search.json_search_test.test_search_found)
key should be found, return list should not be empty

Traceback (most recent call last):
  File "/home/kali/laptrinhantoan/lab1/unittest/test_json_search.py", line 8, in test_search_found
    self.assertTrue(json_search("issueSummary",data) != [])
AssertionError: False is not true
```

Bước 7. Điều tra và sửa lỗi trong đoạn mã recursive_json_search.py

- Thêm điều kiện **if ret_val is None** thì **ret_val = []** vào trong hàm **json_search()** ta giải quyết được **ret_val** được thiết lập lại bằng rỗng khi đệ quy và khi bắt đầu gọi hàm **json_search()** lần đầu tiên thì **ret_val** luôn rỗng.

- Thêm tham số là **ret_val** cho hàm **json_search()** để **ret_val** không rỗng khi đệ quy.

```
1 from test_data import *
2 def json_search(key,input_object,ret_val):
3     if ret_val is None:
4         ret_val=[]
5     if isinstance(input_object, dict): # Iterate dictionary
6         for k, v in input_object.items(): # searching key in the dict
7             if k == key:
8                 temp={k:v}
9                 ret_val.append(temp)
10            if isinstance(v, dict): # the value is another dict so repeat
11                json_search(key,v,ret_val)
12            elif isinstance(v, list): # it's a list
13                for item in v:
14                    if not isinstance(item, (str,int)): # if dict or list repeat
15                        json_search(key,item,ret_val)
16        else: # Iterate a list because some APIs return JSON object in a list
17            for val in input_object:
18                if not isinstance(val, (str,int)):
19                    json_search(key,val,ret_val)
20        return ret_val
21
22 print(json_search("issueSummary",data,None))
```

- Chạy code test và pass 3 phép thử.

```
(kali㉿kali)-[~/laptrinhantoan/lab1/unittest]
$ python3 test_json_search.py
[{'issueSummary': 'Network Device 10.10.20.82 Is Unreachable From Controller'}]
...
Ran 3 tests in 0.000s
OK
(kali㉿kali)-[~/laptrinhantoan/lab1/unittest]
$
```

Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này

YÊU CẦU CHUNG

- Sinh viên tìm hiểu và thực hiện bài tập theo yêu cầu, hướng dẫn.
- Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
- Sinh viên báo cáo kết quả thực hiện và nộp bài.

Báo cáo:

- File **.PDF**. Tập trung vào nội dung, không mô tả lý thuyết.
- Nội dung trình bày bằng **Font chữ Times New Romans/ hoặc font chữ của mẫu báo cáo này (UTM Neo Sans Intel/UTM Viet Sach)– cỡ chữ 13. Canh đều (Justify) cho văn bản. Canh giữa (Center) cho ảnh chụp.**
- Đặt tên theo định dạng: [Mã lớp]-ExeX_GroupY. (trong đó X là Thứ tự Bài tập, Y là mã số thứ tự nhóm trong danh sách mà GV phụ trách công bố).

Ví dụ: [NT101.K11.ANTT]-Exe01_Group03.

- Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
- **Không đặt tên đúng định dạng – yêu cầu, sẽ KHÔNG chấm điểm bài nộp.**
- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Đánh giá:

- Hoàn thành tốt yêu cầu được giao.
- Có nội dung mở rộng, ứng dụng.

Bài sao chép, trộm, ... sẽ được xử lý tùy mức độ vi phạm.

HẾT