Part2-UnitTesting

GitHub URL: https://github.com/SeeAnish/Final-Project-2022FSSW567-A/tree/main/Part2

1. **Requirements:**
    1. **The system shall be able to scan the MRZ of a travel document using a hardware device scanner and get the information in MRZ as two strings (line 1 and line 2 from the above Figure). Note that you do not need to worry about the implementation of the hardware device. But you need to define this method for the software part. This means that you define an empty method for this function.**

```
6  def scan_MRZ():
7      pass
8
```

We define a empty method for scan the MRZ

    2. **The system shall be able to decode the two strings from specification #1 into their respective fields and identify the respective check digits for the fields.**

```python
def encode(string0: str) -> str:
    string1 = string0[45:]
    passport_number = string1[0:9]
    country_code = string1[10:13]
    birth_date = string1[13:19]
    sex = string1[20]
    expiration_date = string1[21:27]
    personal_number = string1[28:37]

    d= {}
    k={}
    d['line2'] =  k
    k['passport_number'] = passport_number
    k['country_code'] = country_code
    k['birth_date'] = birth_date
    k['sex'] = sex
    k['expiration_date'] = expiration_date
    k['personal_number'] = personal_number

    return d
```

Test input

```
85
86
87  if __name__ == '__main__':
88      print(encode('P<CIVLYNN<<NEVEAH<BRAM<<<<<<<<<<<<<<<<<<<<<<;W620126G54CIV5910106F9707302AJ010215I<<<<<<
```

Test output

```
{'line2': {'passport_number': 'W620126G5', 'country_code': 'CIV', 'birth_date': '591010', 'sex': 'F', 'expiration_date': '
970730', 'personal_number': 'AJ010215I'}}
```

3. **The system shall be able to encode travel document information fields queried from a database into the two strings for the MRZ in a travel document. This is the opposite process compared to specification #2. Assume that the database function is not ready. But for testing purposes, you need to define a method for database interaction and leave it empty.**

Our main algorithm to use for get check digit:

```
def algorithm(string: str) -> str:
    printable = digits + ascii_uppercase
    string = string.upper().replace("<", "0")
    weight = [7, 3, 1]
    summation = 0
    for i in range(len(string)):
        c = string[i]
        if c not in printable:
            raise ValueError("%s contains invalid characters" % string, c
        summation += printable.index(c) * weight[i % 3]
    summation %=10
    return summation
```

decoded

```
def decode(dict):
    translator = Translator()
    dict1 = dict["line1"]
    dict2 = dict["line2"]
    issc = dict1["issuing_country"]
    lastname = dict1["last_name"]
    givenname = dict1["given_name"]
    passport = dict2["passport_number"]
    country = dict2["country_code"]
    birth = dict2["birth_date"]
    sex = dict2["sex"]
    exd = dict2["expiration_date"]
    pn = dict2["personal_number"]
    line1decode = "P<" + issc + lastname + "<<" + givenname.replace(' ','<')
    line1decoded1 = line1decode.ljust(44,'<')
    line2decode = passport + str(algorithm(passport)) + country + birth + str(algorithm(birth)) + \
    sex +exd + str(algorithm(exd)) + pn + "<<<<<<" + str(algorithm(pn))
    decoded = line1decoded1 +";"+line2decode
    return decoded
```

With input

```
if __name__ == __main__ :
    print(decode({
            "line1": {
                "issuing_country": "CRI",
                "last_name": "LYNN",
                "given_name": "NEVEAH BRAM"
            },
            "line2": {
                "passport_number": "W620126G5",
                "country_code": "CIV",
                "birth_date": "591010",
                "sex": "F",
                "expiration_date": "970730",
                "personal_number": "AJ010215I"
            }
        }))
```

Output is

```
 & 'C:\Users\byy\AppData\Local\Microsoft\WindowsApps\python3.10.exe' 'c:\Users\byy\.vscode\extensions\ms-pyth
2.20.1\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '61825' '--' 'c:\Users\byy\Desktop\Fina
2FSSW567-A\unit\MRTD.py'
None
P<CRILYNN<<NEVEAH<BRAM<<<<<<<<<<<<<<<<<<<<;W620126G54CIV5910106F9707302AJ010215I<<<<<<6
```

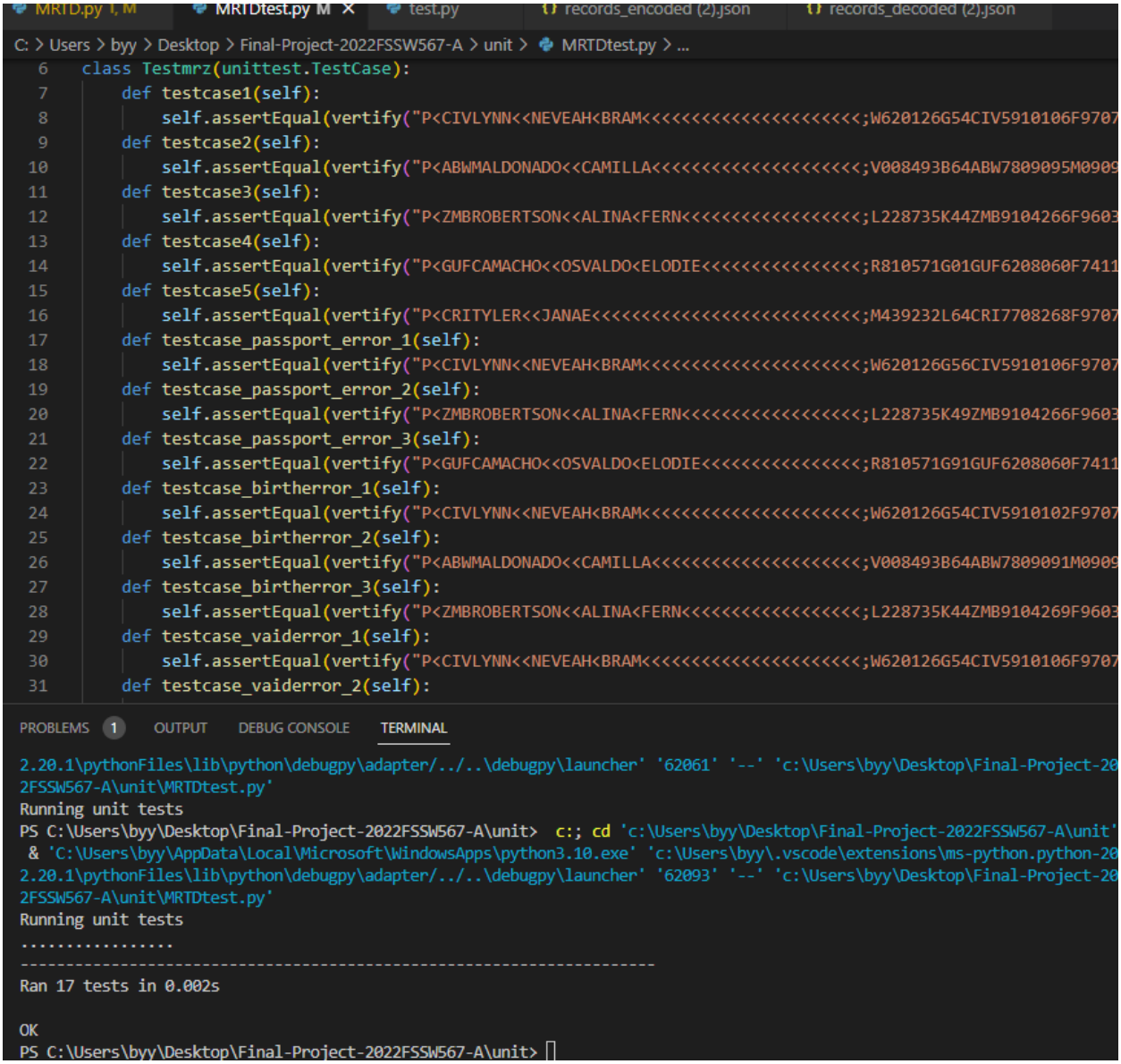
4. **The system shall be able to report a mismatch between certain information fields and the check digit. The system shall report where the miss match happened, i.e. which information field does not match its respective check digit.**

Verify function:

```python
def vertify (string0: str) -> str:
    string1 = string0[45:]
    passport = string1[0:9]
    passport_vertify_code = int(string1[9])
    birth = string1[13:19]
    birth_vertify_code = int(string1[19])
    validity = string1[21:27]
    validity_vertify_code = int(string1[27])
    personal_code = string1[28:43]
    personal_vertify_code = int(string1[43])

    if algorithm(passport) != passport_vertify_code:
        return("passport info error")
    elif algorithm(birth) != birth_vertify_code:
        return("birth date info error")
    elif algorithm(validity) != validity_vertify_code:
        return("validity info error")
    elif algorithm(personal_code) != personal_vertify_code:
        return("personal code error")
    else:
        return("passed")
```

Test case:



Test Case Results:

Show Summary Failed Passed All

Test Group/Test case	Count	Pass	Fail	Error	View
Testmrz	17	17	0	0	Detail
testcase1		PASS			
testcase2		PASS			
testcase3		PASS			
testcase4		PASS			
testcase5		PASS			
testcase_birtherror_1		PASS			
testcase_birtherror_2		PASS			
testcase_birtherror_3		PASS			
testcase_passport_error_1		PASS			
testcase_passport_error_2		PASS			
testcase_passport_error_3		PASS			
testcase_personalcodeerror_1		PASS			
testcase_personalcodeerror_2		PASS			
testcase_personalcodeerror_3		PASS			
testcase_vaiderror_1		PASS			
testcase_vaiderror_2		PASS			
testcase_vaiderror_3		PASS			
Total	17	17	0	0	

2. Coverage test results

```
卷的序列号是 3C28-E145

 C:\Users\byy\Desktop\Final-Project-2022FSSW567-A\unit 的目录

2022/12/10  04:00    <DIR>          .
2022/12/10  04:00    <DIR>          ..
2022/12/10  04:29             1,267 MRTD.py
2022/12/10  04:30             2,502 MRTDtest.py
2022/12/10  03:54    <DIR>          __pycache__
               2 个文件          3,769 字节
               3 个目录 14,915,981,312 可用字节

C:\Users\byy\Desktop\Final-Project-2022FSSW567-A\unit>coverage run MRTDtest.py
Running unit tests
.................
----------------------------------------------------------------------
Ran 17 tests in 0.003s

OK

C:\Users\byy\Desktop\Final-Project-2022FSSW567-A\unit>coverage report -m
Name          Stmts   Miss  Cover   Missing
-------------------------------------------
MRTD.py          32      1    97%   11
MRTDtest.py      40      0   100%
-------------------------------------------
TOTAL            72      1    99%

C:\Users\byy\Desktop\Final-Project-2022FSSW567-A\unit>
```

3.Use MutPy to perform mutation testing

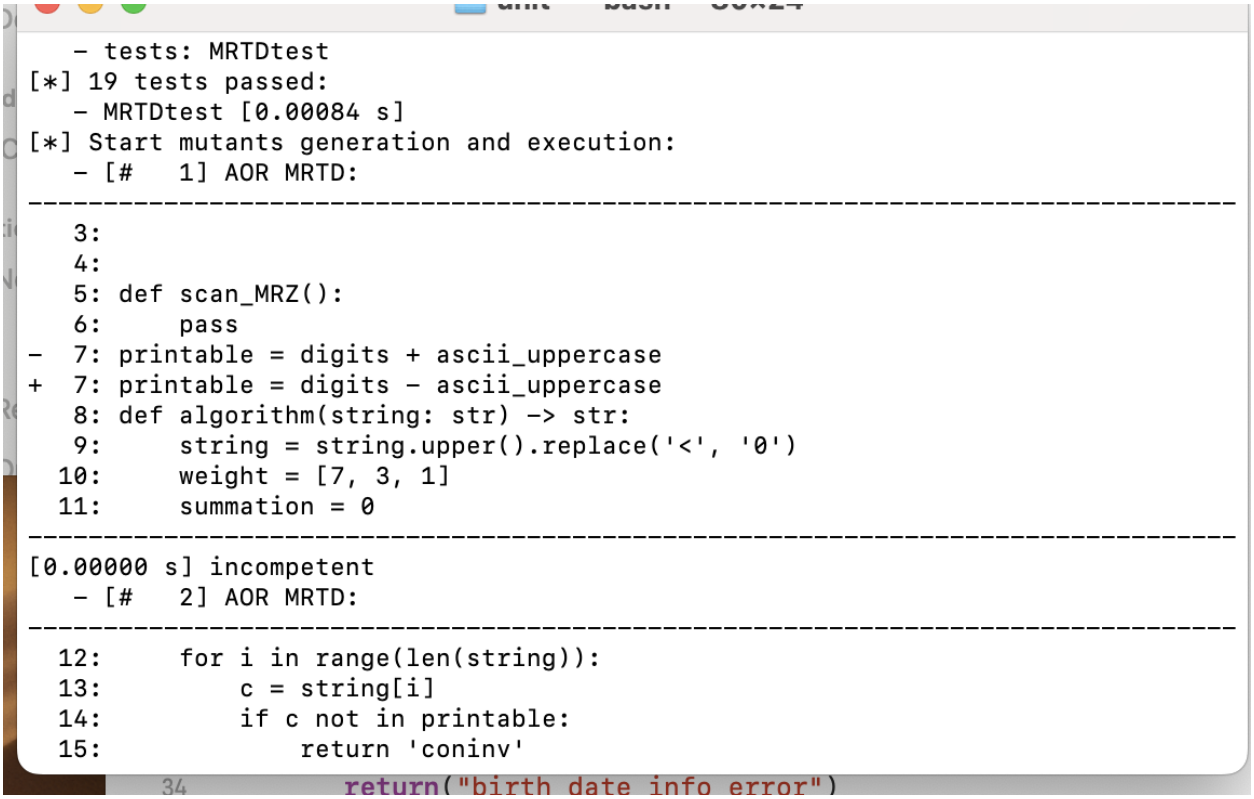
```
  33:         return 'passport info error'
--------------------------------------------------------------------------------
[0.00000 s] incompetent
   - [#  43] SIR MRTD:
--------------------------------------------------------------------------------
  25:     birth = string1[13:19]
  26:     birth_vertify_code = int(string1[19])
  27:     validity = string1[21:27]
  28:     validity_vertify_code = int(string1[27])
- 29:     personal_code = string1[28:43]
+ 29:     personal_code = string1[28:]
  30:     personal_vertify_code = int(string1[43])
  31:
  32:     if algorithm(passport) != passport_vertify_code:
  33:         return 'passport info error'
--------------------------------------------------------------------------------
[0.00000 s] incompetent
[*] Mutation score [9.24660 s]: 0.0%
   - all: 43
   - killed: 0 (0.0%)
   - survived: 34 (79.1%)
   - incompetent: 9 (20.9%)
   - timeout: 0 (0.0%)
kannimne:unit pinwei$
```

1. **How many mutants are generated based on your functions?**
   43
2. **How many mutants are killed by your test cases; how many mutants survived your test cases? Discuss how you could improve your test cases based on results from MutPy.**
   Killed 0 mutants, and 9 incompetent, we can try using a variety of test inputs, including edge cases and invalid inputs. This can help ensure that the test cases are thorough and cover a wide range of possibilities.
3. **Bonus Point: Please create additional test cases to kill the mutants. And list the names of the additional test cases.**

The first mutant was survived

```
unit — -bash — 80×24
   - tests: MRTDtest
[*] 18 tests passed:
   - MRTDtest [0.00080 s]
[*] Start mutants generation and execution:
   - [#   1] AOR MRTD:
----------------------------------------------------------------------------
   5: def scan_MRZ():
   6:     pass
   7:
   8: def algorithm(string: str) -> str:
-  9:     printable = digits + ascii_uppercase
+  9:     printable = digits - ascii_uppercase
  10:     string = string.upper().replace('<', '0')
  11:     weight = [7, 3, 1]
  12:     summation = 0
  13:     for i in range(len(string)):
----------------------------------------------------------------------------
[0.23055 s] survived
   - [#   2] AOR MRTD:
----------------------------------------------------------------------------
  11:     weight = [7, 3, 1]
  12:     summation = 0
  13:     for i in range(len(string)):
  14:         c = string[i]
```

I write the printable to global variable, this mutant became incompetent


- tests: MRTDtest
[*] 19 tests passed:
- MRTDtest [0.00084 s]
[*] Start mutants generation and execution:
- [# 1] AOR MRTD:
3:
4:
5: def scan_MRZ():
6: pass
- 7: printable = digits + ascii_uppercase
+ 7: printable = digits - ascii_uppercase
8: def algorithm(string: str) -> str:
9: string = string.upper().replace('<', '0')
10: weight = [7, 3, 1]
11: summation = 0
[0.00000 s] incompetent
- [# 2] AOR MRTD:
12: for i in range(len(string)):
13: c = string[i]
14: if c not in printable:
15: return 'coninv'
34 return("birth date info error")