Functional Analysis – EC and BV

Apr/2025 Shmuel Gershon / Michael Stahl

Overview

In this exercise we continue our validation work on the T-Tweak web-based service.

In case you forgot, re-read the opening paragraphs of Exercise 2.

This exercise:

The following URL presents the specification of our service's APIs: http://t-tweak.gershon.info/docs /
We fixed many of the issues you found in Exercise 2 (the ones that matter; we think…)

Your mission:

a) Analyze the "password" API, list the equivalence classes of this API and propose test cases.
b) Analyze the "Substring" API, list the boundary values and propose test cases.

Directions and hints:

General

You don't have to run the tests you suggest, although running them may help you better understand the API. You can call the APIs from the URL directly, or using curl – for example:

```
C:\> curl http://t-tweak.gershon.info/substring/abcdefghij/1/1
```

or Python:

```
import requests

…

# build the url

…

# Call the API

url = ‘http://t-tweak.gershon.info/substring/abcdefghij/1/1’

response = requests.get(url)
```

Password - guide

1) Work with the Rules of the password API to identify the ECs (the rules appear as part of the API documentation).
2) We require two equivalence partitioning:
    a. Based on the input
    b. Based on the output
3) Some simplifying rules:
    a. Once the score value is zero, the code stops checking other rules
    b. The order of rules evaluation is:
        i. Length
        ii. Same-letter or same-digit password string
        iii. Illegal passwords (‘password’, ‘root’, ‘admin’)
        iv. Presence of uppercase and lowercase letters and digits in the password string

4) To give you an idea how many ECs should be in the solution, here is a hint:
    a. Input-based equivalence partition: The correct number of ECs is between 15 and 30
    b. Output-based partition: between 5 and 15

5) Once you defined the ECs, you need to provide one test case for each EC
    a. This would be a representative member of the EC. Example: if one EC is “strings of length between 5 and 8”, then **t#G1b** is a possible test case for this EC. But the example can also be of length 6; or 7; or 8.

6) A correct partition means that any possible password string is a member of only one of the equivalence classes you defined.

Substring - guide

1) BVs are of course dependent on the equivalent partition you use. Which means you need to do equivalent partitioning first.

2) First: do a partition based on each input item by itself and create boundary value tests for this partitioning.
    a. Remember that there are valid and invalid partitions
    b. When testing invalid EC for one input, the other inputs should be valid!
    c. Tests created for this partitioning shall be marked (in the submitted Excel file) as:
       P1: Boundary value tests for **string**
       P2: Boundary value tests for **start**
       P3: Boundary value tests for **end**

3) Second: do partitioning based on the relations between parameters and create boundary value tests for these partitions.
    a. Only cover relations between two parameters at a time. No need to cover relations between all 3 parameters.
    b. Example for relations: **start <= end**
    c. No need to create partitions for relations such as: **start <= end <= string_len**
    d. No need to test the far end of partitions whose far end is +/- MAXINT .
    e. In a test for a certain relation, the 3rd parameter needs to be valid and not introduce another issue.

       Example of a good test for testing **start <= end :** string/start/end = abcdefg/5/6
       Bad example for testing **start <= end :** abcd/5/6

       However…. There are some cases where you must enter an invalid value also to a 2nd parameter. Example: invalid test for **start** value: 21 – you must also have **end=21** or higher, to avoid triggering the start > end error…

    f. You may find that some cases were already covered by previous tests. DO NOT repeat tests – you will lose points for redundant tests. Hint: See the example for “however” in (e). What other boundary test does it cover?

    g. Tests created for this partitioning shall be marked (in the submitted Excel file) as **P4**.

4) No need to test **start** and **end** with non-digit inputs. In real life you’d need to test for it, but for this exercise we remove this requirement.

5) No need to analyze output equivalence partitions – just the inputs

6) For each boundary, give the two test cases that provide the 2-point boundary value coverage.

a. Some test cases are not feasible for trying out via the Swagger UI, since it blocks you from trying values out of range. You should still list these tests. If you want to run them (*you don't have to as part of this exercise!!!*), such tests can be executed by direct call to the server using **curl** or a simple Python code. You can also look at the structure of the "request URL" the Swagger UI shows for a valid call, modify it as needed, and paste it in a new browser window.

How to submit:

Submit the results in three .csv files, with the following headers (examples in purple)

Filename: **Input_ECs_<your ID>.csv**

Contains the tests that cover the *input ECs*

EC description	Test case	Your ID
Len(password) = 5	5gwer	012345678
Etc.		

WARNING: If you build your solution in Excel, and save the file as .csv for the submission, be careful about how Excel will save long numbers. It will most likely turn the number 1000000000 to 1.00E+09 . Our exercise tester will read this as '1.00E+09' and not as you wanted: 1000000000. So check what you got in the csv before submitting. Or prevent Excel from doing this conversion by using '1000000000 (which force Excel to treat this as a string and not a number).

Filename: O**utput_ECs_<your ID>.csv**

Contains the tests that cover the *input ECs*

EC description	Test case	Your ID
<some EC>	<appropriate test case>	012345678

The description of an EC is assumed to mean: "All strings that match the following condition"

Filename: **EX3_substring_<your ID>.csv**

Parameter	Partition	EC	String	Start	End	Order	Validity	Your ID
String	P1	Len(password) = 5	Abcde	2	4	1	Valid	012345678
String	P1	Len(password) = 6	34D353	4	3	2	Invalid	012345678
								…
Start	P2	Start < 6	…	…	…			012345678
…			…	…	…			…
End	P3	End > 3	…	…	…			012345678
…								…
String / Start	P4	…	…	…	…			012345678

Parameter: The BV of what parameter is tested by this test. For tests that cover relation, write the two parameters.

Partition: The partition group defined in the guide the boundary value applies.

EC: The equivalence class whose boundaries are covered by the proposed test case.

String, Start, End: The values to use in the test

Order: The serial number of the test (start at 1 and increment by 1 for each additional test)

Validity: Does this test case covers the valid or invalid side of the boundary?

Your ID: Your ID number (תעודת זהות)

Grading

The exercise weight in the final grade is 6 points. It is not mandatory exercise.

Password EC testing – weight: 60%

You lose points for undertesting (ECs not covered by any test) and for over-testing (more than a single test for each EC).

Substring BV analysis – weight: 40%

You lose points for undertesting (A boundary is not covered by one valid, one invalid tests) and for over-testing (more than a single test to test each side of a boundary).

The deadline for submission is 28/May/25 23:55 .

Late submission penalty:

- One day: -1 point (out of the 6)
- Two days: -2 points (out of the 6)

You can’t submit later than 2 days after the deadline.