

# SWEN90006: Assignment 1

Name: Yangzhe Xie

Student number: 1029787

Email: yangzhe.xie@student.unimelb.edu.au

August 31, 2019

## 1 Task 1

### 1.1 Test template trees

Figure 1 - 4 shows the test template trees for the API addUser, loginUser, updateDetails, and retrieveDetails respectively.

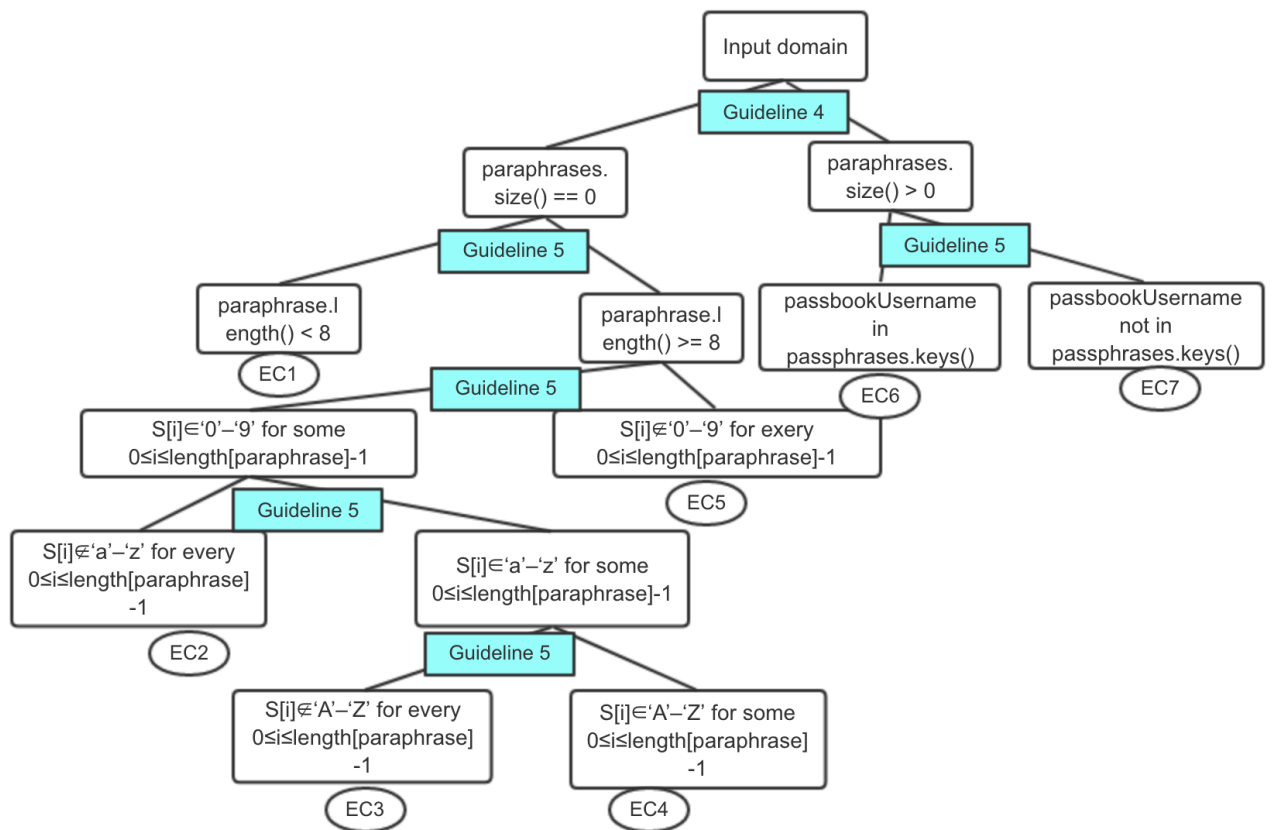


Figure 1: Test template tree for `addUser()`

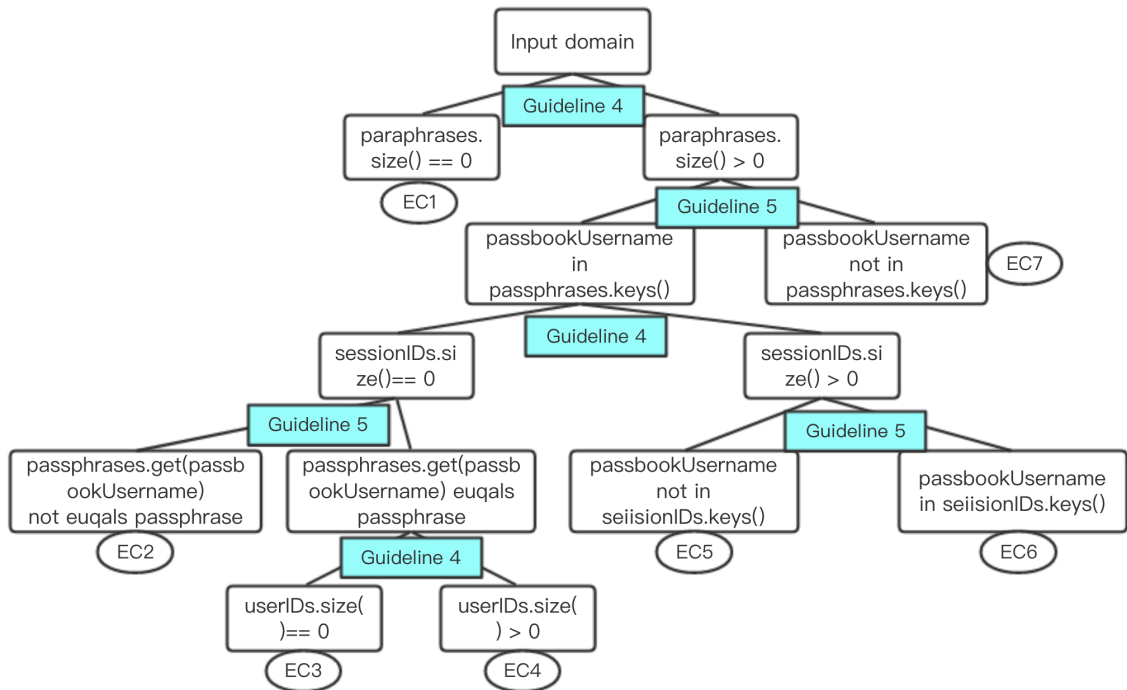


Figure 2: Test template tree for loginUser()

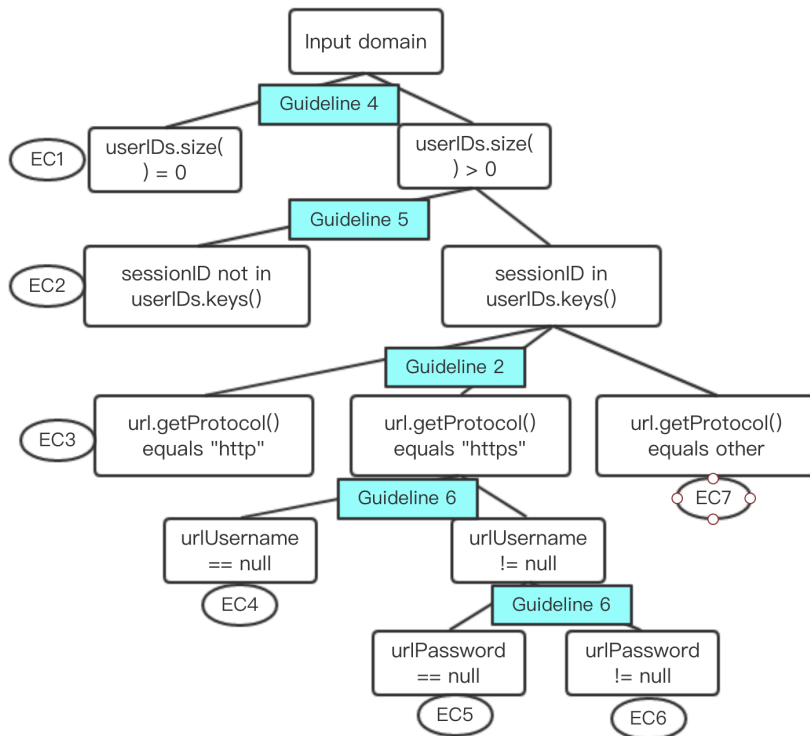


Figure 3: Test template tree for updateDetails()

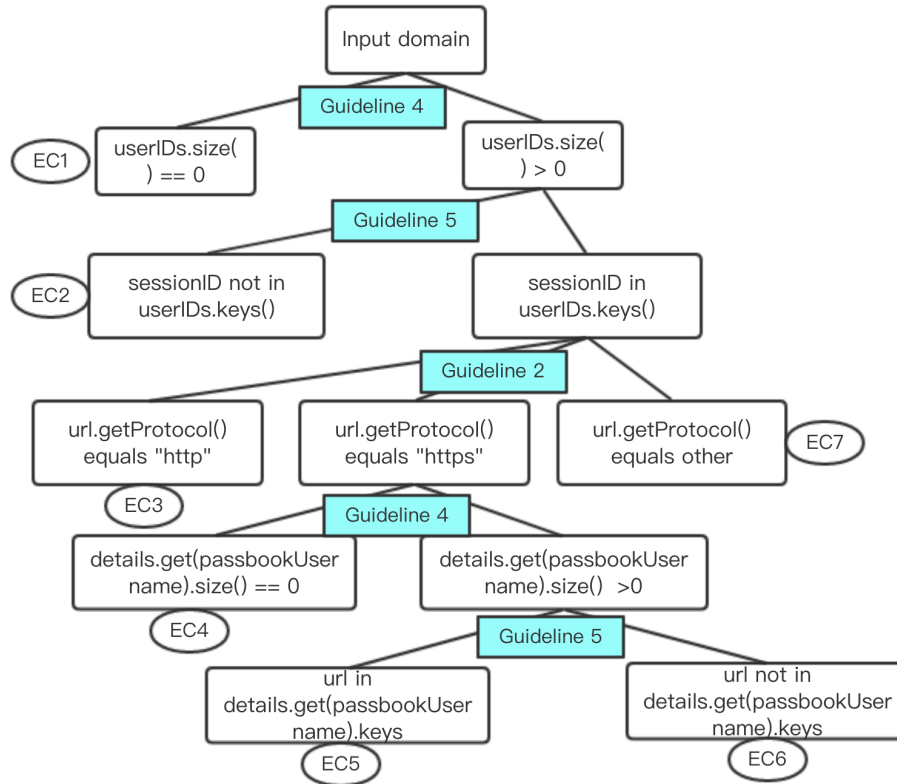


Figure 4: Test template tree for retrieveDetails()

## 1.2 Do your set of equivalence classes cover the input space?

My set of equivalence classes cover the input space. The reasons are as follows:

- 1) All leaf nodes are divided strictly and carefully, so that they do not overlap with other leaf.
- 2) The collection of the set of each sibling node covers all the cases of their parent node.
- 3) If two variables are independent of each other, then the subtree of one variable can be added to a leaf node of the other variable. In this case, all the nodes add up to cover all situations.
- 4) As part of your input domain, the instance variables should also be considered. Note that all of these variables are collections, so according to guideline 4, we should follow the zero-one-many rule. But in this particular case, we just care about whether the collection contains some values. So I combined the two cases (number of elements equals 1 and greater than 1) into one (greater than 0), which does not affect the results of the tests.

## 2 Test cases associated with equivalence classes

### 2.1 addUser

Table 1: Test cases for addUser

ID	Test case	Expected output
----	-----------	-----------------

EC1	paraphrases = {}, passbookUsername = "abc", paraphrase = "12345aA"	WeakPassphraseException
EC2	paraphrases = {}, passbookUsername = "abc", paraphrase = "1234567A"	WeakPassphraseException
EC3	paraphrases = {}, passbookUsername = "abc", paraphrase = "1234567a"	WeakPassphraseException
EC4	paraphrases = {}, passbookUsername = "abc", paraphrase = "123456aA"	-
EC5	paraphrases = {}, passbookUsername = "abc", paraphrase = "abcdABCD"	WeakPassphraseException
EC6	paraphrases = {"abcd": "123456aA"}, passbookUsername = "abcd", paraphrase = "123456aA"	DuplicateUserException
EC7	paraphrases = {"abcd": "123456aA"}, passbookUsername = "abc", paraphrase = "123456aA"	-

## 2.2 loginUser

Table 2: Test cases for loginUser

ID	Test case	Expected output
EC1	paraphrases = {}, sessionIDs = {}, userIDs = {}, passbookUsername = "abc", paraphrase = "123456aA"	NoSuchUserException
EC2	paraphrases = {"abc": "123456aA"}, sessionIDs = {}, userIDs = {} passbookUsername = "abc", paraphrase = "123456aB"	IncorrectPassphraseException
EC3	paraphrases = {"abc": "123456aA"}, sessionIDs = {}, userIDs = {} passbookUsername = "abc", paraphrase = "123456aA"	...
EC4	paraphrases = {"abc": "123456aA"}, sessionIDs = {}, userIDs = {123: "def"} passbookUsername = "abc", paraphrase = "123456aA"	...
EC5	paraphrases = {"abc": "123456aA"}, sessionIDs = {"def": 123}, userIDs = {} passbookUsername = "abc", paraphrase = "123456aA"	...
EC6	paraphrases = {"abc": "123456aA"}, sessionIDs = {"abc": 123}, userIDs = {} passbookUsername = "abc", paraphrase = "123456aA"	AlreadyLoggedInException
EC7	paraphrases = {"abc": "123456aA"}, sessionIDs = {}, userIDs = {} passbookUsername = "abcd", paraphrase = "123456aA"	NoSuchUserException

## 2.3 updateDetails

Table 3: Test cases for updateUserDetails

ID	Test case	Expected output
EC1	userIDs = {}, sessionID = 123, url = "http://test.com", urlUsername = "123", urlPassword = "123"	InvalidSessionIDException
EC2	userIDs = {123:"abc"}, sessionID = 456, url = "http://test.com", urlUsername = "123", urlPassword = "123"	InvalidSessionIDException
EC3	userIDs = {123:"abc"}, sessionID = 123, url = "http://test.com", urlUsername = "123", urlPassword = "123"	-
EC4	userIDs = {123:"abc"}, sessionID = 123, url = "https://test.com", urlUsername = null, urlPassword = "123"	-
EC5	userIDs = {123:"abc"}, sessionID = 123, url = "https://test.com", urlUsername = "123", urlPassword = null	-
EC6	userIDs = {123:"abc"}, sessionID = 123, url = "https://test.com", urlUsername = "123", urlPassword = "123"	-
EC7	userIDs = {123:"abc"}, sessionID = 123, url = "ftp://test.com", urlUsername = "123", urlPassword = "123"	MalformedURLException

## 2.4 retrieveDetails

# 3 Boundary-value analysis

## 3.1 addUser

Table 4: Test cases for addUser

EC	Test case	Remark
EC1	paraphrases = {}, passbookUsername = "abc", paraphrase = "12345aA"	off point for paraphrase.length() < 8
EC2	paraphrases = {}, passbookUsername = "abc", paraphrase = "123456`A"	off point for S[i] not in 'a' - 'z', for every 0 ≤ i ≤ length[paraphrase]-1
EC2	paraphrases = {}, passbookUsername = "abc", paraphrase = "123456{A"	off point for S[i] not in 'a' - 'z', for every 0 ≤ i ≤ length[paraphrase]-1
EC3	paraphrases = {}, passbookUsername = "abc", paraphrase = "123456@a"	off point for S[i] not in 'A' - 'Z', for every 0 ≤ i ≤ length[paraphrase]-1
EC3	paraphrases = {}, passbookUsername = "abc", paraphrase = "123456[a"	off point for S[i] not in 'A' - 'Z', for every 0 ≤ i ≤ length[paraphrase]-1
EC4	paraphrases = {}, passbookUsername = "abc", paraphrase = "234567nA"	on point for S[i] in 'A' - 'Z', for some 0 ≤ i ≤ length[paraphrase]-1

EC4	paraphrases = {}, passbookUsername = "abc", paraphrase = "234567nZ"	on point for S[i] in 'A' - 'Z', for some $0 \leq i \leq \text{length}[\text{paraphrase}] - 1$
EC4	paraphrases = {}, passbookUsername = "abc", paraphrase = "234567Na"	on point for S[i] in 'a' - 'z', for some $0 \leq i \leq \text{length}[\text{paraphrase}] - 1$
EC4	paraphrases = {}, passbookUsername = "abc", paraphrase = "234567Nz"	on point for S[i] in 'a' - 'z', for some $0 \leq i \leq \text{length}[\text{paraphrase}] - 1$
EC4	paraphrases = {}, passbookUsername = "abc", paraphrase = "abcdABC0"	on point for S[i] in '0' - '9', for some $0 \leq i \leq \text{length}[\text{paraphrase}] - 1$
EC4	paraphrases = {}, passbookUsername = "abc", paraphrase = "abcdABC9"	on point for S[i] in '0' - '9', for some $0 \leq i \leq \text{length}[\text{paraphrase}] - 1$
EC5	paraphrases = {}, passbookUsername = "abc", paraphrase = "abcdABC/"	off point for S[i] not in '0' - '9', for every $0 \leq i \leq \text{length}[\text{paraphrase}] - 1$
EC5	paraphrases = {}, passbookUsername = "abc", paraphrase = "abcdABC:"	off point for S[i] not in '0' - '9', for every $0 \leq i \leq \text{length}[\text{paraphrase}] - 1$
EC6	paraphrases = {"abcd": "123456aA"}, passbookUsername = "abcd", paraphrase = "123456aA"	on point for passbookUsername in passphrases.keys()
EC7	paraphrases = {"abcd": "123456aA"}, passbookUsername = "abc", paraphrase = "123456aA"	on point for passbookUsername not in passphrases.keys()

### 3.2 loginUser

Table 5: Test cases for loginUser

ID	Test case	Expected output
EC1	paraphrases = {}, sessionIDs = {}, userIDs = {}, passbookUsername = "abc", paraphrase = "123456aA"	NoSuchUserException
EC2	paraphrases = {"abc": "123456aA"}, sessionIDs = {}, userIDs = {} passbookUsername = "abc", paraphrase = "123456aB"	IncorrectPassphraseException
EC3	paraphrases = {"abc": "123456aA"}, sessionIDs = {}, userIDs = {} passbookUsername = "abc", paraphrase = "123456aA"	...
EC4	paraphrases = {"abc": "123456aA"}, sessionIDs = {}, userIDs = {123: "def"} passbookUsername = "abc", paraphrase = "123456aA"	...
EC5	paraphrases = {"abc": "123456aA"}, sessionIDs = {"def": 123}, userIDs = {} passbookUsername = "abc", paraphrase = "123456aA"	...

EC6	paraphrases = {"abc":"123456aA"}, sessionIDs = {"abc":123}, userIDs = {} passbookUsername = "abc", paraphrase = "123456aA"	AlreadyLoggedInException
EC7	paraphrases = {"abc":"123456aA"}, sessionIDs = {}, userIDs = {} passbookUsername = "abcd", paraphrase = "123456aA"	NoSuchUserException

### 3.3 updateDetails

Table 6: Test cases for updateUserDetails

ID	Test case	Expected output
EC1	userIDs = {}, sessionID = 123, url = "http://test.com", urlUsername = "123", urlPassword = "123"	InvalidSessionIDException
EC2	userIDs = {123:"abc"}, sessionID = 456, url = "http://test.com", urlUsername = "123", urlPassword = "123"	InvalidSessionIDException
EC3	userIDs = {123:"abc"}, sessionID = 123, url = "http://test.com", urlUsername = "123", urlPassword = "123"	-
EC4	userIDs = {123:"abc"}, sessionID = 123, url = "https://test.com", urlUsername = null, urlPassword = "123"	-
EC5	userIDs = {123:"abc"}, sessionID = 123, url = "https://test.com", urlUsername = "123", urlPassword = null	-
EC6	userIDs = {123:"abc"}, sessionID = 123, url = "https://test.com", urlUsername = "123", urlPassword = "123"	-
EC7	userIDs = {123:"abc"}, sessionID = 123, url = "ftp://test.com", urlUsername = "123", urlPassword = "123"	MalformedURLException

### 3.4 retrieveDetails