Faculty of Computers, Informatics and Microelectronics
Technical University of Moldova

Software Testing
Laboratory work #3

Black-box technique

Authors:

Vlad Ganusceac

Supervisor:

Mariana Catruc

2019
Chişinau

Instruction:

1. Enter to the following url: localhost:6880/Home
2. The registration form will be loaded
3. The form will contain three input fields:
    a. For email;
    b. For password;
    c. For password confirmation;
4. In case of successful registration the system will redirect us to Home/Index. Otherwise, the loop will be continued from step [2].
5. Click the button doTwit.
6. The input form will be loaded. Enter the message. All twits are starting with #.
7. Commit your message entering corresponding button.
8. The message has been saved and is stored in Home/Twits directory of Website.
9. Check if it was saved.

Password validation (requirements):

1. Password should be not less than 8 characters.
2. It must contain:
    a. At least one digit;
    b. At least one lower case character;
    c. At least one upper case character;
    d. At list one of symbols from this set:
    specialSet = {‘,’ ,’.’ ,’$’, ‘_’ , ‘!’ ,’@’ , ‘#’ , ‘%’ , ‘&’ , ‘^’ , ‘*’ , ‘<’ , ‘>’};
    e. All other characters are strictly restricted;
3. Its maximum length shouldn’t be greater than 40 characters;
4. The new password must be unique.

The second requirement could be represented in the following manner (a – e):

$$Password \cap \{1, \dots 9\} \neq \emptyset \; and \; Password \cap \{'a', \dots, 'z'\} \neq \emptyset \; and \; Password \cap \{'A', \dots, 'Z'\} \neq \emptyset \; and \; Password \subset \{1, \dots 9\} \cup \{'a', \dots, 'z'\} \cup \{'A', \dots, 'Z'\} \cup \text{specialSet}$$

In order to have the minimum number of test cases for this item we should check the cases:

1. When all three conditions are true;
2. One condition is false (4 tests).

The first requirement and the fourth one could be united into one condition:

$$7 < Length < 41$$

It should be checked in the test where other conditions are valid.

Thus, the decision table would look like this:

Relation / Condition	R1	R2	R3	R4	R5	R6
$7 < Length < 41$	True	**False**	True	True	True	True
$Password \cap \{1, \dots 9\} \neq \emptyset$	True	True	**False**	True	True	True
$Password \cap \{'a', \dots, 'z'\} \neq \emptyset$	True	True	True	**False**	True	True

$Password \cap \{'A', \dots, 'Z'\} \neq \emptyset$	True	True	True	True	**False**	True
$Password \subset \{1, \dots 9\} \cup \{'a', \dots, 'z'\} \cup \{'A', \dots, 'Z'\} \cup specialSet$	True	True	True	True	True	**False**
Success	***True***	False	False	False	False	False

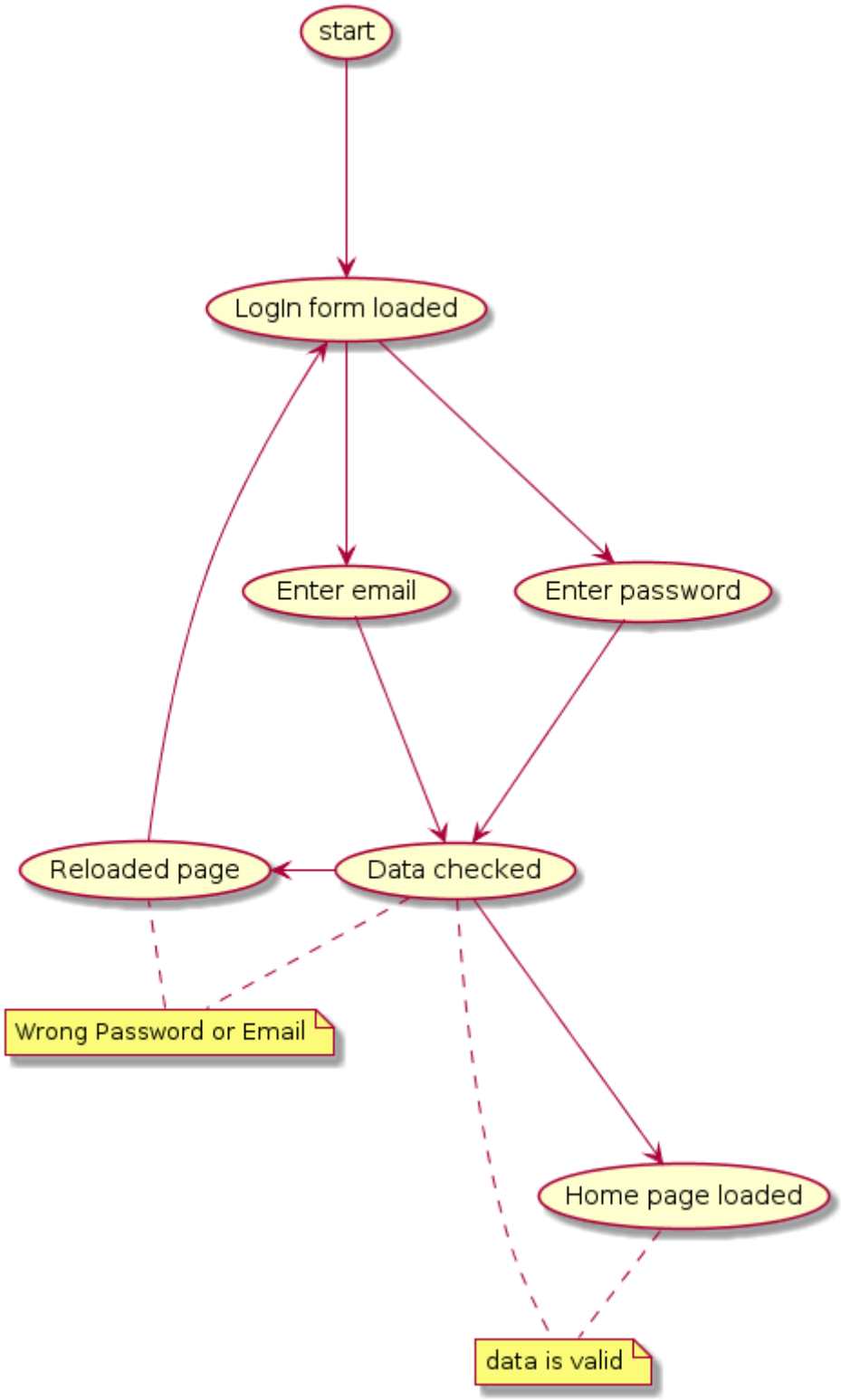
We have got the respective decision table because all the conditions were join i.e. between them where relation 'and'.

The decision table would look like this:

Test	Input	Description	Output
1	1Administrator*	$7 < Length < 41$ and $Password \cap \{0, \dots, 9\} \neq \emptyset$ and $Password \cap \{'a', \dots, 'z'\} \neq \emptyset$ and $Password \cap \{'A', \dots, 'Z'\} \neq \emptyset$ and $Password \subset \{0, \dots 9\} \cup \{'a', \dots, 'z'\} \cup \{'A', \dots, 'Z'\}$	Password corresponds to all norms.
2	1Adm*	$\boldsymbol{Length < 8}$ and $Password \cap \{0, \dots, 9\} \neq \emptyset$ and $Password \cap \{'a', \dots, 'z'\} \neq \emptyset$ and $Password \cap \{'A', \dots, 'Z'\} \neq \emptyset$ and $Password \subset \{0, \dots 9\} \cup \{'a', \dots, 'z'\} \cup \{'A', \dots, 'Z'\} \cup specialSet$	The length of the password should be at least 8 characters.
3	1Administrator*7_fkegoiegkKHFFEKDJKFEPKEFFLKFK&JFKgeggfeeJgbjegueguejfgieghgfufeekfoeghiOegowhgiowhegifygfyegfygeyfgyFuhfugeufg87835hujgf3gf*^&ffdSfleifkdnvovrn,r.vrnknbscubemv	$\boldsymbol{Length > 40}$ and $Password \cap \{0, \dots, 9\} \neq \emptyset$ and $Password \cap \{'a', \dots, 'z'\} \neq \emptyset$ and $Password \cap \{'A', \dots, 'Z'\} \neq \emptyset$ and $Password \subset \{0, \dots 9\} \cup \{'a', \dots, 'z'\} \cup \{'A', \dots, 'Z'\} \cup specialSet$	The length of the password should not exceed 40 characters.
4.	@^Adm1n^	**Boundary test** $\boldsymbol{Length = 8}$ and $Password \cap \{0, \dots, 9\} \neq \emptyset$ and $Password \cap \{'a', \dots, 'z'\} \neq \emptyset$ and $Password \cap \{'A', \dots, 'Z'\} \neq \emptyset$ and $Password \subset \{0, \dots 9\} \cup \{'a', \dots, 'z'\} \cup \{'A', \dots, 'Z'\} \cup specialSet$	Password corresponds to all norms.
5.	0123456789abcdefghklmAJFEKPKGOE*&#^$.,JFi01	**Boundary test** $\boldsymbol{Length = 40}$ and	Password corresponds

		$Password \cap \{0, \dots, 9\} \neq \emptyset$ and $Password \cap \{'a', \dots, 'z'\} \neq \emptyset$ and $Password \cap \{'A', \dots, 'Z'\} \neq \emptyset$ and $Password \subset \{0, \dots 9\} \cup \{'a', \dots, ' z'\}$ $\cup \{'A', \dots, ' Z'\}$ $\cup specialSet$	to all norms.
6	Gefjh&^dTe	$7 < Length < 41$ and $\boldsymbol{Password} \cap \{\mathbf{0}, \dots, \mathbf{9}\} = \emptyset$ and $Password \cap \{'a', \dots, 'z'\} \neq \emptyset$ and $Password \cap \{'A', \dots, 'Z'\} \neq \emptyset$ and $Password \subset \{0, \dots 9\} \cup \{'a', \dots, ' z'\}$ $\cup \{'A', \dots, ' Z'\}$ $\cup specialSet$	The password must contain at least one digit.
7	*AB87%DFE5^	$7 < Length < 41$ and $Password \cap \{0, \dots, 9\} \neq \emptyset$ and $\boldsymbol{Password} \cap \{'\boldsymbol{a}', \dots, '\boldsymbol{z}'\} = \emptyset$ and $Password \cap \{'A', \dots, 'Z'\} \neq \emptyset$ and $Password \subset \{0, \dots 9\} \cup \{'a', \dots, ' z'\}$ $\cup \{'A', \dots, ' Z'\}$ $\cup specialSet$	The password must contain at least one camel case character English character
8	#adowm75^r&k	$7 < Length < 41$ and $Password \cap \{0, \dots, 9\} \neq \emptyset$ and $Password \cap \{'a', \dots, 'z'\} \neq \emptyset$ and $\boldsymbol{Password} \cap \{'\boldsymbol{A}', \dots, '\boldsymbol{Z}'\} = \emptyset$ and $Password \subset \{0, \dots 9\} \cup \{'a', \dots, ' z'\}$ $\cup \{'A', \dots, ' Z'\}$ $\cup specialSet$	The password must contan at least one Pascal case English character.
9	238Agre90*_//';:”	$7 < Length < 41$ and $Password \cap \{0, \dots, 9\} \neq \emptyset$ and $Password \cap \{'a', \dots, 'z'\} \neq \emptyset$ and $Password \cap \{'A', \dots, 'Z'\} \neq \emptyset$ and $\boldsymbol{Password} \not\subset \{\mathbf{0}, \dots \mathbf{9}\} \cup \{'\boldsymbol{a}', \dots, ' \boldsymbol{z}'\}$ $\cup \{'\boldsymbol{A}', \dots, ' \boldsymbol{Z}'\}$ $\cup \boldsymbol{specialSet}$	The password contains some non-permissive characters.

State transition diagram “Authentication process”:



Do twit:

1. Each twit starts with a specific character: ‘#’;
2. This character must appear no more than 1 time;
3. Message shouldn’t be empty (i.e. ‘#’ can’t be symbol terminus);
4. The maximum length of the twit is 100 characters.

Analyzing all the requirements, we may achieve the following conclusion:

1. Check input data where:

a. Exists only one ‘#’;
    i. Was set in the beginning;
    ii. Any other place in twit.
b. There are no ‘#’;
c. There are more than one ‘#’ characters (lets it be 2).

2. The length of the twit:
    a. Less than 100 characters;
    b. Empty twit;
    c. Greater than 100 characters;
    d. Boundary check, i.e. 100 characters.

Relation / Condition	R1	R2	R3	R4
Number of ‘#’	True	**False**	True	True
In the beginning of twit	True	True	**False**	True
0 < Length < 101	True	True	True	**False**
Success	***True***	False	False	False

Decision table:

Test	Input	Description	Output
1	#Hello	Valid test	OK
2	#	Empty twit	Fail
3	Hel#lo	Wrong place of ‘#’	Fail
4	##Hello	More than one ‘#’	Fail
5	#jfefoeogjeogjekjgiejgoegj Wfljfegjoejgiweg93ut8tkh Fkoegj8hhkefp3juep083kf Jejifhe;wjeogowegjoe[geo Wejgi;v-rg94lmbveovpo0 Kgokeogkorjgorjgpojbnevr Fjeigibhrpk9rbh80g49gu4	More than 100 characters	Fail
6	***Here should be twit with 100 characters exactly***	Boundary check	OK

Conclusion:

During this laboratory work we have showed our knowledge accumulated during lectures and from personal experience. The black-box testing technic is very useful in the initial stage when all the requirements of the project are discussed between members of the teem and during in-go testing. The undiscussable strong point of this method is that we shouldn’t take care how does the code is/was/will be written, we have input data and a concrete result which we tend to achieve.

Thus, with help of State-transition diagram we are analyzing possible paths in the application and define the minimum number of classes (entities – depending on context) in order to make our code more reusable. After that, when we have established modules and their functionality, we are starting to create unit tests which will cover all possible outcomes. The aim is to check all possible errors and unexpected behavior in the project. It is considered a good practice to do this number of tests as few as possible.