Black Box Testing

Validation and Verification of Software

Cristian Gustavo Castro Xum
Cristina Martín Bris

The program allows the following functionalities:

- help
- enter_patient
- next_patient <ARG1>
- finalize_treatment <ARG1>
- get_historical <ARG1>
- search_patient<ARG1>
- change_priority <ARG1> <ARG2>
- patients_by_day <ARG1>
- patients_by_area <ARG1>
- number_patients_area <ARG1>
- patients_priority_area <ARG1> <ARG2>

STEP 1: start of the program

The program can be started without arguments or with 1 argument. In the case of passing an argument, it must be a text file.(Because only to create / modify information of a patient is necessary the information that exists in the file).
The file must contain the following fields in three different rows: name, priority and area.

Input Condition	Valid Class	Invalid Class
I. File	1. Empty 2. File that exists and its name is an alphanumeric string.	3. File does not exist 4. The file is not a text file 5. Empty file 6.The file does not contains 3 rows. 7. The file has not an alphanumeric name.
II. Name*	8. The field name inside the file is a string of letters.	9. The file contains 3 rows but the field name is not a string of letters.
III. Priority	10. The field priority inside the file has a correct value (CRITICAL, MODERATE, GENERAL).	11. The field priority has a wrong value(different to CRITICAL, MODERATE, GENERAL).
IV. Area	12. The field area inside the file is a string of letters.	13.The field area is not a string of letters.

*The name of the file is not checked very depthed because there may be different identity card formats.

Eq. Class	Valid Class					Invalid Class							
Input Condit	I		II	III	IV	I					II	III	IV
Eq. Class	1	2	8	10	12	3	4	5	6	7	9	11	13
TC1	X												
TC2		X	X	X	X								
TC3						X							
TC4							X						
TC5								X					
TC6									X				
TC7										X			
TC8		X		X	X						X		
TC9		X	X		X							X	
TC10		X	X	X									X

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC1	Start the program without arguments.	-	Ok message	Unable to retrieve data	Fail
TC2	Start the program with one argument.	Pass as an argument a file containing three rows: the first row a string of letters, the second row with CRITICAL value, and the third with a string of letters.	Ok message	Unable to retrieve data	Fail
TC3	Start the program with one argument, and this argument is a file that does not exist.	A file that does not exists	Error message	Unable to retrieve data	Fail
TC4	Start the program with one argument, and this argument is a file that is not a text file.	A file that is not a text file ‘prueba.xlsx’	Error message	Unable to retrieve data	Fail
TC5	Start the program with one argument, and this argument is a file that is empty	An empty file	Error message	Unable to retrieve data	Fail
TC6	Start the program with one argument, and this argument is a text file that do not have 3 rows.	A file with 2 rows	Error message	Unable to retrieve data	Fail
TC7	The name of the file is not an alphanumeric value	A file that has not an alphanumeric value: 6&!ª1.txt	Error message	Unable to retrieve data	Fail

TC8	Start the program with one argument, this argument is a text file and the field name is not a string of letters.	A file that contains ‘carm3n’ in the name field	Error message	Unable to retrieve data	Fail
TC9	Start the program with one argument, this argument is a text file and the field priority has a wrong value	A file that contains ‘HOLA’ in the field priority	Error message	Unable to retrieve data	Fail
TC10	Start the program with one argument, this argument is a text file and the field area is not a string of letters.	A file that contains ‘PEDR1ATRIA’ in the area field.	Error message	Unable to retrieve data	Fail

STEP 2: program functionalities

1. help

Context: for this function it is assumed that the program has been started with or without a file (and the file is a valid file).
Also, we check that each function works correctly with the start option with file and without file.

Input Condition	Valid Class	Invalid Class
I. File	1. The program starts without a file. 2. The program starts with a file.	
II. Arguments	3. Empty	4. A non empty string.

Eq. Class	Valid Class			Invalid Class
Input Condition	I		II	II
Eq. Class	1	2	3	4
TC11	X		X	
TC12		X	X	
TC13		X		X

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC11	Test the help option having started the program without a file.	empty	A help menu	enter_patient - Enter new patient with the file passed as program argument enter_patient <FilePath> - Enter new patient from the given file next_patient <Area> - Retrieve next patient in the area mentioned finalize_treatment <PatientId> - Discharge patient get_historical <PatientId> - Retrieve history of the patient search_patient <PatientId> - Retrieve patient who is not discharged change_priority <PatientId> <newPriority> - Change priority of patient to a new priority patients_by_day	Pass

				<Date> - Retrieve all patients admitted on the date given in dd-mm-yyyy format. patients_by_area <Area> - Retrieve patients from a given area number_patients _area <Area> - Retrieve number of patients in specific area patients_priority_ area <Area> <Priority> - Retrieve patients by area and priority Exit - Exit the program	
TC12	Test the help option having started the program with a file.	empty	A help menu	enter_patient - Enter new patient with the file passed as program argument enter_patient <FilePath> - Enter new patient from the given file next_patient <Area> - Retrieve next patient in the area mentioned finalize_treatment <PatientId> - Discharge patient get_historical <PatientId> - Retrieve history of the patient search_patient <PatientId> - Retrieve patient who is not discharged	Pass

				change_priority <PatientId> <newPriority> - Change priority of patient to a new priority patients_by_day <Date> - Retrieve all patients admitted on the date given in dd-mm-yyyy format. patients_by_area <Area> - Retrieve patients from a given area number_patients _area <Area> - Retrieve number of patients in specific area patients_priority_ area <Area> <Priority> - Retrieve patients by area and priority Exit - Exit the program	
TC13	Test the help option with an argument	‘prueba’	Error message	enter_patient - Enter new patient with the file passed as program argument enter_patient <FilePath> - Enter new patient from the given file next_patient <Area> - Retrieve next patient in the area mentioned finalize_treatment <PatientId> - Discharge patient get_historical <PatientId> -	Fail

				Retrieve history of the patient search_patient <PatientId> - Retrieve patient who is not discharged change_priority <PatientId> <newPriority> - Change priority of patient to a new priority patients_by_day <Date> - Retrieve all patients admitted on the date given in dd-mm-yyyy format. patients_by_area <Area> - Retrieve patients from a given area number_patients_area <Area> - Retrieve number of patients in specific area patients_priority_area <Area> <Priority> - Retrieve patients by area and priority Exit - Exit the program	

2. Enter Patient

Context: It is assumed that the file that is loaded has a correct format.

Input Condition	Valid Class	Invalid Class
I. File	1. The program starts with a file	2. The program starts without a file
II. Arguments	3. Empty	4. A non empty string. 5. A user that already exists.

Eq. Class	Valid Class		Invalid Class		
Input Condition	I	II	I	II	
Eq. Class	1	3	2	4	5
TC14	X	X			
TC15		X	X		
TC16	X			X	
TC17	X				X

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC14	Test the enter_patient option having started the program with a file and an empty argument	empty	empty/ok message	empty	Pass
TC15	Test the enter_patient option having started the program without a file.	empty	Error message	null	Fails
TC16	Test the neter_patient option with a file and an argument	‘’prueba’	Error message	Invalid Command !	Pass

TC17	Test the enter_patient option with a user that already exists in the system.	51155953C.txt	Error message	empty	Fail

3. Next Patient

Context: It is assumed that the file that is loaded has a correct format.

Input Condition	Valid Class	Invalid Class
I. File	1. The program starts without a file. 2. The program starts with a file.	
II. Arguments (AREA)	3. One argument that is a string of letters.	4. An empty string. 5. More than one argument.

Eq. Class	Valid Class			Invalid Class	
Input Condition	I		II	I	II
Eq. Class	1	2	3	4	5
TC18	X		X		
TC19		X		X	
TC20		X			X
TC21		X	X		

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC18	Test the next_patient option having started the program without a file and and one correct argument	‘Pediatría’	Information about the next patient.	No patients left !	Fail*
TC19	Test the next_patient option with a file and without arguments.	empty	Error message	Invalid Command !	Pass
TC20	Test the next_patient option having started the program with a file and and more than one argument	‘Pediatría’ ‘hola’	Error message	No patients left !	Fail
TC21	Test the next_patient option having started the program with a file and and one correct argument	‘Pediatría’	Information about the next patient.	No patients left !	Fail

*The program does not keep a history of the operations performed.

4. Finalize Treatment

Context: It is assumed that the file that is loaded has a correct format.

Input Condition	Valid Class	Invalid Class
I. File	1. he program starts with a file.	2. The program starts without a file.
II. Arguments (dni)	3. One argument that is and alphanumeric string(dni) os a user and it exists.	4. One argument that is not al alphanumeric string 5. An empty argument 6. An argument that is an alphanumeric string(dni), but this user does not exist in the system. 7. An argument that is a user that was discharged.

Eq. Class	Valid Class		Invalid Class				
Input Condition	I	II	I	II			
Eq. Class	1	3	2	4	5	6	7
TC22	X	X					
TC23			X				
TC24	X			X			
TC25	X				X		
TC26	X					X	
TC27	X						X

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC22	Test finalize_option without a file and a correct argument	51155953C	OK message	Patient not found !	Fail*
TC23	Test finalize_option without a file	51155955C	Error message	Patient not found !	Fail
TC24	Test finalize_option with a wrong value of the argument	51??8Z	Error message	Patient not found !	Pass
TC25	Test finalize_option with an empty value of the argument	Empty	Error message	Invalid Command !	Pass
TC26	Trying to finalize the treatment of a patient that does not exist.	99999999D	Error message	Patient not found !	Pass

TC27	Trying to finalize the treatment of a patient that was already discharged.	51155953C	Error message	Patient not found !	Fail

*The program does not keep a history of the operations performed.

5. Get Historical

Context It is assumed that the file that is loaded has a correct format.

Input Condition	Valid Class	Invalid Class
I. File	1. The program starts with a file. 2. The program starts without a file.	
II. Arguments (dni)	3. One argument that is and alphanumeric string(dni) os a user and it exists.	4. One argument that is not al alphanumeric string 5. An empty argument 6. An argument that is the identity number of a person that is not in the system

Eq. Class	Valid Class			Invalid Class		
Input Condition	I		II	II		
Eq. Class	1	2	3	4	5	6
TC28	X		X			
TC29		X	X			
TC30	X			X		
TC31	X				X	
TC32	X					X

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC28	Test get_historical option with a file and a correct argument	51155953C	Info about the patient/Error message if not exists.	Patient not found !	Fail
TC29	Test get_historical option without a file and a correct argument	51155953C	Info about the patient/Error message if not exists	Patient not found !	Fail
TC30	Test get_historical option with a file and a wrong argument	????	Error message	Patient not found !	Fail
TC31	Test get_historical option with a file and a wrong argument	Empty	Error message	Invalid Command !	Pass
TC32	Test the gest_historical option with a file and with an argument that is the identity number of a person that is not in the system	‘09876543Q’	Error message	Patient not found !	Pass

6. Search Patient

Context It is assumed that the file that is loaded has a correct format.

Input Condition	**Valid Equivalence Class**	**Invalid Equivalence Class**
I. File	1. The program starts with a file. 2. The program starts without a file.	
II. Argument1	3. Any string composed by alpha numerical characters.	4. Any string which contains at least one non alphanumeric character. 5. Empty String

Eq. Class type	Valid Eq. classes			Invalid Eq. classes	
Input Condition	II	I		II	
Eq. Class	3	1	2	4	5
TC33	X	X	-	-	-
TC34	X	-	X		
TC35	-	-	-	X	-
TC36	-	-	-	-	X

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC33	Test search_patient with a valid string of alphanumeric characters with a file	search_patient 12345678Y	Ok message	ID : 12345678Y Name : Cristian Castro Priority : General Area : PEDIATRIA DateEntry : Thu Dec 20 11:28:37 CET 2018 DateDischarge d : --	pass
TC34	Test search_patient with a valid string of alphanumeric characters without a valid a file	search_patient 4242424C	Ok message	Patient not found	fail*
TC35	Test search_patient with a string containing non alphanumeric characters	search_patient 42423%&!	Error message	Patient not found	fail
TC36	Test search_patient with an empty string	search_patient	Error message	Invalid command	pass

***The program does not keep an historical of the operations performed..

7. Change priority

Context It is assumed that the file that is loaded has a correct format.

Input Condition	**Valid Equivalence Class**	**Invalid Equivalence Class**
I. File	1. The program starts with a file	2. The program starts without a file
II . Argument1	3. Any string of alphanumeric characters that matches with an already registered user that has not finalized his treatment yet	4. Any string which contains at least one non alphanumeric character. 5. Any string which does not match with an already registered user. 6. Any string that match with an already registered user but with finalized treatment 7. Empty String
III. Argument2	8. Any state from the set of {CRITICAL, MODERATE, GENERAL} different from the current state of the patient.	9. Any string which not belongs to the set of words {CRITICAL, MODERATE, GENERAL} 10. Any state which is the same with the current state of the patient 11. Empty String

Eq. Class type	Valid Eq. classes			Invalid Eq. classes							
Input Condition	I	II	III	I	II				III		
Eq. Class	1	3	8	2	4	5	6	7	9	10	11
TC37	X	X	X	-	-	-	-	-	-	-	-
TC38	X	-	X	X	-	-	-	-	-	-	-
TC39	X	-	X	-	X	-	-	-	-	-	-
TC40	X	-	X	-	-	X	-	-	-	-	-
TC41	X	-	X	-	-	-	X	-	-	-	-
TC42	X	-	X	-	-	-	-	X	-	-	-
TC43	X	X	-	-	-	-	-	-	X	-	-
TC44	X	X	-	-	-	-	-	-	-	X	-
TC45	X	X	-	-	-	-	-	-	-	-	X

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC37	Test to change to a valid priority state using a valid user (string of alphanumeric) already registered who has not finished his treatment yet and the program starts with a valid file	change_priority 51155953C CRITICAL	Ok message	Invalid value for new priority	fail
TC38	Test to change to a valid priority case using a valid user (string of	change_priority 51155953C CRITICAL	Error Message	Invalid value for new priority	fail

	alphanumeric) already registered who has not finished his treatment yet and the program starts without a file				
TC39	Test case to a different valid priority state using an user (string with non alphanumeric characters) and the program starts with a valid file	change_priority 51155953C%% CRITICAL	Error message	Patient not found	fail
TC40	Test change priority to a different valid state using a non registered user (string of alphanumeric) and the program starts with a valid file	search_patient 5115595311C MODERATE	Error message	Patient not found	Pass
TC41	Test change priority to a different valid priority state using an user who already finalized his treatment and the program starts with a valid file	search_patient 5115595311C MODERATE	Error message	Patient not found	fail
TC42	Test change priority to a different valid priority state using an empty string user and the program starts with a valid file	change_priority MODERATE	Error message	Invalid command	pass
TC43	Test to change to a an invalid priority case using a valid user (string of	change_priority 42342424C MODERADO	Error Message	Patient not found	fail

	alphanumeric) already registered who has not finished his treatment yet and the program starts with a valid file				
TC44	Test to change to a valid priority state but with the same state, using a valid user (string of alphanumeric) already registered who has not finished his treatment yet and the program starts with a valid file	change_priority 5115595311C MODERATE	Error Message	Patient not found	fail
TC45	Test to change to state with empty string, using a valid user (string of alphanumeric) already registered who has not finished his treatment yet and the program starts with a valid file	change_priority 5115595311C	Error Message	Invalid Command	pass

8. Patients by day

Context It is assumed that the file that is loaded has a correct format.

Input Condition	**Valid Equivalence Class**	**Invalid Equivalence Class**
I. File	1.The program starts without a file. 2. The program starts with a valid file.	
II. Argument1	3. Any string with the format dd-mm-yyyy where that date corresponds to a valid calendar date and is before to the current date.	4. Any String with a format different to dd-mm-yyyy 5. Any String with the format dd-mm-yyyy but is not a valid calendar date. 6. Any valid calendar date with the format dd-mm-yyyy but is a future date 7. Empty string

Eq. Class type	Valid Eq. classes			Invalid Eq. classes			
Input Condition	II	I		II			
Eq. Class	3	1	2	4	5	6	7
TC46	X	X	-	-	-	-	-
TC47	X	-	X	-	-	-	-
TC48	-	-	X	X	-	-	-
TC49	-	-	X	-	X	-	-
TC50	-	-	X	-	-	X	-
TC51	-	-	X	-	-	-	X

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC46	Test patients by day with a valid calendar date using a valid format, before the current date and program starts without a valid file	patients_by_day 01-12-2018	Ok message	No patients found	pass
TC47	Test patients by day with a valid calendar date using a valid format, before the current date and program starts with a file	patients_by_day 01-12-2018	Ok message	No patients found	pass
TC48	Test patients by day with a invalid format	patients_by_day 01 of december, 2018	Error message	No patients found	fail
TC49	Test patients by day with a valid format but using an invalid calendar date	patients_by_day 32-04-2018	Error message	No patients found	fail
TC50	Test patients by day with a valid calendar date using a valid format date but is an upcoming date	patients_by_day 01-04-2019	Error message	No patients found	fail
TC51	Test patients by day using an empty string	patients_by_day	Error message	Invalid command	pass

9. Patients by area

Context It is assumed that the file that is loaded has a correct format.

Input Condition	**Valid Equivalence Class**	**Invalid Equivalence Class**
I. File	1. The program starts without a file. 2. The program starts with a valid file.	
II. Argument1	3. Any string with alphabetical characters.	4. Any string which contains at least one non alphabetical character. 5. Empty String

Eq. Class type	Valid Eq. classes			Invalid Eq. classes	
Input Condition	II	I		II	
Eq. Class	3	1	2	4	5
TC52	X	X	-	-	-
TC53	X	-	X	-	-
TC54	-	-	X	X	-
TC55	-	-	X	-	X

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC52	Test patients by area with a string of alphabetical characters and program starts	patients_by_area PEDIATRIA	Ok message	patient not found	fail*

	without a file				
TC53	Test patients by area with a string of alphabetical characters and the program starts with a file	patients_by_area PEDIATRIA	Ok message	ID : 12345678Y Name : Cristian Castro Priority : General Area : PEDIATRIA DateEntry : Thu Dec 20 11:28:37 CET 2018 DateDischarged : --	pass
TC54	Test patients by area with a string with at least non alphabetical characters and the program starts with a file	patients_by_area PEDIATRIA3	Error message	No patients found	fail
TC55	Test patients by area with an empty string and the program starts with a file	patients_by_area	Error message	Invalid command	pass

***The program does not keep an historical of the operations performed..

10. Number Patients by area

Context It is assumed that the file that is loaded has a correct format.

Input Condition	**Valid Equivalence Class**	**Invalid Equivalence Class**
I. File	1. The program starts without a file. 2. The program starts with a valid file.	
II. Argument1	3. Any string with alphabetical characters.	4. Any string which contains at least one non alphabetical character. 5. Empty String

Eq. Class type	Valid Eq. classes			Invalid Eq. classes	
Input Condition	II	I		II	
Eq. Class	3	1	2	4	5
TC56	X	X	-	-	-
TC57	X	-	X	-	-
TC58	-	-	X	X	-
TC59	-	-	X	-	X

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC56	Test number of patients by area with a string of alphabetical characters and program starts without a file	number_patients_area PEDIATRIA	Ok message	No. of patients in PEDIATRIA area : 1	pass
TC57	Test number of patients by area with a string of alphabetical characters and the program starts with a file	number_patients_area PEDIATRIA	Ok message	No patients found	pass
TC58	Test number of patients by area with a string with at least non alphabetical characters and the program starts with a file	number_patients_area PEDIATRIA3	Error message	No patients found	pass
TC59	Test number of patients by area with an empty string and the program starts with a file	number_patients_area	Error message	Invalid command	pass

11. Patients by priority area

Context It is assumed that the file that is loaded has a correct format.

Input Condition	**Valid Equivalence Class**	**Invalid Equivalence Class**
II. File	1. The program starts with a file 2. The program starts without a file	
II . Argument1	3. Any string of alphabetical characters	4. Any string which contains at least one non alphabetical character. 5. Empty String
III. Argument2	6. Any state from the set of {CRITICAL, MODERATE, GENERAL} different from the current state of the patient.	7. Any string which not belongs to the set of words {CRITICAL, MODERATE, GENERAL} 8. Empty String

Eq. Class type	Valid Eq. classes				Invalid Eq. classes			
Input Condition	I		II	III	II		III	
Eq. Class	1	2	3	6	4	5	7	8
TC60	-	X	X	X	-	-	-	-
TC61	X	-	X	X	-	-	--	-
TC62	X	-	-	X	X	-	-	-
TC63	X	-	-	X	-	X	-	-
TC64	X	-	X	-	-	-	X	-
TC65	X	-	X	-	-	-	-	X

ID	Goal	Input	Expected Output	Real Output	Pass/Not pass
TC60	Test patients priority area with a valid area (string of alphabetical) and valid state and the program starts without a valid file	patients_priority_area pediatria CRITICAL	Ok message	No patients found !	fail*
TC61	Test patients priority area with a valid area and a valid state and the program starts with a file	patients_priority_area pediatria CRITICAL	Ok Message	No patients found !	fail
TC62	Test patients priority area with an invalid area (non alphabetical characters) and a valid state and the program starts with a file	patients_priority_area pediatria%& CRITICAL	Error message	No patients found	pass
TC63	Test patients prioirty area with an empty area and a valid state and the program starts with a file	patients_priority_area CRITICAL	Error message	Invalid command	pass
TC64	Test patients priority area with a valid area and an invalid state and the program starts with a file	patients_priority_area pediatria CRITICOX	Error Message	Invalid priority	pass
TC65	Test patients priority area with an valid area and a empty state and the program starts with a file	patients_priority_area pediatria	Error Message	Invalid command	pass

***The program does not keep an historical of the operations performed.

Test cases that fails:

TC1
TC2
TC3
TC4
TC5
TC6
TC7
TC8
TC9
TC10
TC13
TC15
TC17
TC18
TC20
TC21
TC22
TC23
TC27
TC28
TC29
TC30
TC34
TC35
TC37
TC38
TC39
TC41
TC43
TC44
TC48
TC49
TC50
TC52
TC54
TC60
TC61