

## Exercise 1

- Apply the equivalence partitioning black box technique to obtain the test cases for a software module that classifies individuals according to the following inputs:
  - Code: string of 3 digits not starting with "00"
  - Control character: A character in 'a'..'z' or character '-'
  - Person Type: "hired" or "interim"
- The expected output values are:
  - "S1", if the code represents an even number and person is "hired".
  - "S2", if the code represents an even number and person is "interim".
  - "S3", if the code represents an odd number and person is "hired".
  - "S4", if the code represents an odd number and person is "interim".
  - "S5", if the control character is "-".
  - "S6", otherwise.
- Note: in case of conflict, S5 has more priority.

! LOOK EXPECTED OUTPUT → REDUCED CLASSES

## Equivalence Partitioning

- Identification Heuristics of equivalence classes:
  - If a range of values is specified for the input data, a valid class and two invalid classes will be created. *Check on type of value*
  - If a finite value is specified a valid class and two invalid classes will be created
  - If a condition of type «it must be» or a boolean condition is specified (e.g. "the first character must be a character from the alphabet", a valid class «it is an alphabet character» and another invalid class «it is not an alphabet character» will be created
  - If a set of valid accepted values and the program treats them in a different way, then a valid class for each value and another invalid class must be created
  - If it is suspected that some concrete elements of a class are handled in a different way, the class must be partitioned into reduced classes

EXACT VALUE

Input classes	Valid cases	Invalid cases	Heuristic
Code	Code: $ParseInt() \in [10 \dots 999]$	Code: $ParseInt() \Rightarrow Error$ Code: $ParseInt() < 10$ Code: $ParseInt() > 999$	range <del>boolean</del>
Control-char	Control-char $\in ['a' \dots 'z']$ Control-char = '-'	char < 'a' (except from '-') char > 'z'	range <del>set of accepted values</del>
Person	"hired" "interim"	other	Set of values

Diff approach:

1 valid  
2 invalid

! Cover as many valid cases in 1 test

Valid classes	Input	Output

Invalid Classes	Input	Output

SOLUTION

Input	Valid Classes	Invalid Classes	Heuristic
Code	(1) 3 char, even number, which no starts by '00' (2) 3 char, odd number, which no starts by '00'	(3) No number (4) (less 3 digit) (5) (more 3 digit) (6) Starts by '00'	Boolean Finite Values, minor classes
Control Character	(7) Value in ['a'..'z'] (8) '-'	(9) < 'a' (under lower, except -) (10) > 'z' (upper highest)	Boolean Range Values
Person	(11) 'hired' (12) 'interim'	(13) other	Set of values

Valid Classes	Input	Output	Invalid Classes	Input	Output
(1) 3 char, even number, which no starts by '00'	Code: 222, Control: S1, Person: hired	S1	(3) No number	Code: 222, Control: S1, Person: hired	S6
(2) 3 char, odd number, which no starts by '00'	Code: 111, Control: S1, Person: hired	S5	(4) (less 3 digit)	Code: 222, Control: S1, Person: hired	S6
(6) Starts by '00'	Code: 000, Control: S1, Person: hired	S6	(5) (more 3 digit)	Code: 999, Control: S1, Person: hired	S6
(7) Value in ['a'..'z']	Code: 222, Control: S1, Person: hired	S4	(9) < 'a' (under lower, except -)	Code: 222, Control: S1, Person: hired	S6
(8) '-'	Code: 222, Control: S1, Person: hired	S5	(10) > 'z' (upper highest)	Code: 222, Control: S1, Person: hired	S6
(11) 'hired'	Code: 222, Control: S1, Person: hired	S4			
(12) 'interim'	Code: 222, Control: S1, Person: interim	S4			

## Exercise 4

Apply the equivalence partitioning black box technique to obtain the test cases for a method which generates a report according to the following inputs:

- Student Name: Which contains at least name and one surname
- Group: three characters, the first is a letters A, C or D, and the following two are two digits from 01 until 15
- Theory mark (T): positive number (10 maximum)
- Laboratory mark (L): positive number (10 maximum)
- Deliverable mark (D) : positive number (8 maximum)

In the report, the method adds the final course mark by means of the equation:  $0.6 \cdot T + 0.4 \cdot P + 0.1 \cdot D$

Provide the equivalence partitioning table with the input variable, valid classes, invalid classes and heuristic applied. Also, provide two tables the resulting with test cases, one for valid classes and another for the invalid ones.

Input classes	Valid classes	Invalid classes	Heuristic
Student name	(1) Has at least a name and 1 surname ✓	(2) It only has less than 2 names ✓ OK, but we could put more	Boolean
Group	(3) First letter must be <u>A, C or D</u> and ends with number in [01, ..., 15] * <small>SET condition</small>	(4) The first letter is not A, C or D (5) The last two characters don't belong to [01, ..., 15]	<del>Exact value</del> / Boolean * As the code behaves in the same way for all
Theory mark	(6) Value in [0, ..., 10]	(7) Value < 0 (8) Value > 10	Range
Laboratory mark	(9) Value in [0, ..., 10]	(10) Value < 0 (11) Value > 10	Range
Deliverable mark	(12) Value in [0, ..., 8]	(13) Value < 0 (14) Value > 8	Range

Valid cases	Test	Output

Invalid cases	Test	Output

**Exercise 4** 12 For each property, write a regular expression that matches the value described. If possible, the value must be in the form of a number or a string. If not, the value must be a string.

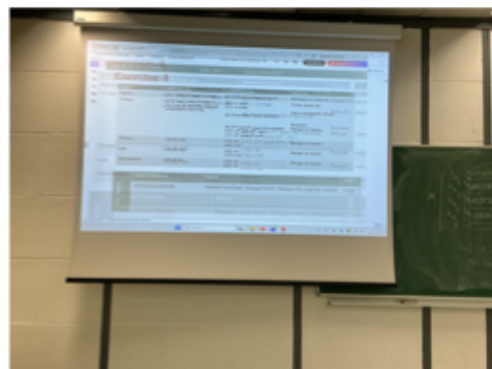
Property	Value	Accepted values	Reason
Name	(7) 1000 at least 2 letters	(2) 1000 at least 2 letters	Boolean: is must be
Group	(1) 3 char, with 1st char A, L or D and the rest are 2 numbers in [0-9]	(5) 3 char (1) 3 char (4) First letter not in [A,L,D]	Boolean: Range of values Set of accepted values
Theory	(12) [0-10]	(12) 0-10 (13) 0-10 (14) 0-10	Boolean: Range of values
Lab	(14) [0-10]	(14) 0-10 (15) 0-10	Boolean: Range of values
Deliverable	(14) [0-10]	(14) 0-10 (15) 0-10	Boolean: Range of values

**Exercise 4** Ingeniería del Software DSC-UPV Curso 2018-2019

Property	Value	Accepted values	Reason
Name	(1) 1000 at least 2 letters	(2) 1000 at least 2 letters	Boolean: is must be
Group	(1) 3 char, with 1st char in {A, L, D} and the rest are 2 numbers in [0-9]	(5) 3 char (1) 3 char (4) First letter not in {A,L,D}	Boolean: Range of values Set of accepted values
Theory	(12) [0-10]	(12) 0-10 (13) 0-10 (14) 0-10	Boolean: Range of values
Lab	(14) [0-10]	(14) 0-10 (15) 0-10	Boolean: Range of values
Deliverable	(14) [0-10]	(14) 0-10 (15) 0-10	Boolean: Range of values

Invalid Class	Reason	Out
(2)(3)(12)(15)(18)	Name: Joe, Group: A10, Theory: 10, Lab: 10, Del: 8	--
(1)(8)(12)(15)(18)	Name: Joe Doe, Group: A1, Theory: 10, Lab: 10, Del: 8	--
(1)(7)(12)(15)(18)	Name: Joe Doe, Group: A01, Theory: 10, Lab: 10, Del: 8	--
(1)(8)(12)(15)(18)	Name: Joe Doe, Group: B01, Theory: 10, Lab: 10, Del: 8	--
(1)(9)(12)(15)(18)	Name: Joe Doe, Group: C0, Theory: 10, Lab: 10, Del: 8	--
(1)(10)(12)(15)(18)	Name: Joe Doe, Group: C00, Theory: 10, Lab: 10, Del: 8	--
(1)(11)(12)(15)(18)	Name: Joe Doe, Group: C10, Theory: 10, Lab: 10, Del: 8	--
(1)(13)(12)(15)(18)	Name: Joe Doe, Group: C10, Theory: 11, Lab: 10, Del: 8	--
(1)(13)(12)(15)(18)	Name: Joe Doe, Group: C10, Theory: 5, Lab: 12, Del: 8	--
(1)(13)(12)(15)(18)	Name: Joe Doe, Group: C10, Theory: 5, Lab: 8, Del: 1	--
(1)(13)(12)(15)(20)	Name: Joe Doe, Group: C10, Theory: 5, Lab: 8, Del: 9	--



# Exercise 1.

Apply the equivalence partitioning black box technique to obtain the test cases for a software module that classifies individuals according to the following inputs:

- Code: string of 3 digits not starting with "00"
- Control character: A character in 'a'..'z' or character '-'
- Person Type: 'hired' or 'interim'


The expected output values are:

- "S1", if the code represents an even number and person is "hired".
- "S2", if the code represents an even number and person is "interim".
- "S3", if the code represents an odd number and person is "hired".
- "S4", if the code represents an odd number and person is "interim".
- "S5", if the control character is "-".
- "S6", otherwise.


Provide the equivalence partitioning table with the input variable, valid classes, invalid classes and heuristic applied. Also, provide two tables the resulting with test cases, one for valid classes and another for the invalid ones.

Input variable	Valid classes	Invalid classes	Heuristic
Code	<p>(1) <u>string of 3 digits, not starting with 00 and representing an even n<sup>e</sup></u></p> <p>(2) <u>string of 3 digits, not starting with 00 and representing an odd n<sup>e</sup></u></p>	<p>(3) <u>String has less than 3 digits</u></p> <p>(4) <u>String has more than 3 digits</u></p> <p>(5) <u>String starts with 00</u></p> <p>(6) <u>Not a number</u></p>	<p><u>Boolean</u></p> <p><u>Reduced classes</u></p> <p><u>Boolean</u></p>
Control character	<p>(7) char belonging to ['a', ..., 'z']</p> <p>(8) char = '-'</p>	<p>(9) char &lt; 'a' (except from '-')</p> <p>(10) char &gt; 'z'</p>	<p>Range of values</p>
Person Type	<p>(11) "hired"</p> <p>(12) "interim"</p>	<p>(13) type ∉ [hired, interim]</p>	<p>Set of valid accepted values</p>

Invalid classes	Input	Output
(3)(7)(11)	Code = "abc" Control character = 'a' Person = 'hned'	S6
(4)(7)(11)	Code = "10" Control character = 'a' Person = 'hned'	S6
(5)(8)(12)	Code = "1000" Control character = '-' Person = 'inturm'	S6
(6)(8)(12)	Code = '001' Control character = '-' Person = inturm	S6
(1)(9)(12)	Code = '100' Control character = '.' Person = inturm	S6
(1)(10)(11)	Person = '100' Control character = 'Z'	S6
(2)(8)(13)	Person = '101' Control char = '-' Person = 'halal'	S6



UNIVERSITAT  
 DE VALÈNCIA



etsinf  
Center for Software Engineering

Introducción a la Ingeniería del Software  
 Chapter 9 Seminar - Black Box

**Exercise 2.**

Applying the black box equivalence partitioning testing technique, obtain the test cases for the following software module that classifies individuals according to the following inputs:

- Creation date: string with format "dd-mm-yyyy" representing a valid date
- Type of person: "student" or "professor"

The expected output values are:

- "S1", if the month is 08 and the person is a student.
- "S2", if the month is not 08 and the person is a professor.
- "S3", if the person is student or professor.
- "S4", in any other correct case.
- "S5", in any other incorrect case.

**Note.** The order of evaluation (priority) of each output is S1..S5.

Provide the equivalence partitioning table with the input variable, valid classes, invalid classes and heuristic applied. Also, provide two tables the resulting with test cases, one for valid classes and another for the invalid ones.

Input	Valid classes	Invalid classes	Heuristic
Creation_date	(1) Has a format dd-mm-yyyy representing a valid date where mm ≠ 08  (2) Has a format dd-08-yyyy representing a valid date	(3) Has a different format (4) It does not represent a valid date	Reduced classes Boolean
Type of person	(5) "student" (6) "professor"	(7) type of person ∉ [student, professor]	Set of accepted values

Valid classes	Input	Output
(1) (6)	date = 09-11-2005 , type = professor	S2
(2) (5)	date = 29-08-2014 , type = student	S1

Invalid classes	Input	Output
(3) (5)	date = 01-error - 1999 , type = student	S5
(4) (6)	date = 42-23-0000 , type = professor	S5
(1) (7)	date = 29-11-2005 , type = hola	S5