Exercise 8

1 Deducing representative test cases from a specification

Assumptions (just for our understanding)

The program has multiple arrays of length 6:
- An array storing the values of the dice.
- A boolean array tracking which dice are fixed from a previous throw.
- A boolean array tracking which dice are fixed from the current throw.
- Implicit: An array tracking unfixed dice.

Parameters

Value of dice (identified by diceIndex)	
1	[if]
5	[if]
2	[if]
3	[single]
4	[single]
6	[single]
< 1	[error]
> 6	[error]

Environment

Fixation of dice	
fixed before rolling dice again	[error]
fixed after rolling dice	[if]
unfixed	[property unfixed]
Base score (from 6 fixed scoring dice)	
Base score = 0	[if]
Base score >= 700	[single]
Three of a kind (Triple Double)	
Triple double not possible: Neither two nor five 1s or 5s are (already) fixed	[if unfixed]
Triple double possible: Two dice with value 1 are (already) fixed	[if unfixed] [single]
Triple double possible: Five dice with value 1 are (already) fixed	[if unfixed] [single]
Triple double possible: Two dice with value 5 are (already) fixed	[if unfixed] [single]
Triple double possible: Five dice with value 5 are (already) fixed	[if unfixed] [single]

Number of test cases:
#if + #if_unfixed + #single = 3x1x1 + 3x1x1x1 + 5+1+1+4 = 3 + 3 + 11 = 17

Implemented test case:
https://gitlab.uni-marburg.de/tadjikys/SQ24_Hamed_Langbein_Semenovykh_Tadjiky/-/blob/ueb08/ueb08/app/src/test/java/org/example/PlayerTest.java?ref_type=heads#L53-78

2 Data-driven testing in JabRef

Parameters

Strings		
length = 0	[single]	
length = 1	[single]	
length > 1	[if]	
Java: array is null	[error]	
Java: at least 1 element of array is null	[error]	
Separator		
length = 0	[single]	
length >= 1	[if]	
Java: String is null	[error]	
From & To		**Comments**
		with len we refer to the length of the strings array
1 <= from + 1 = to <= len	[single]	to is 1 greater than from; both are within bounds
1 <= from + 1 < to <= len	[if]	to is at least 2 greater than from; both are within bounds
from < 0 <= to	[single]	from left of bounds → start with first element
len = from & to >= 0	[error]	behavior if from=len is unspecified → we decided to expect an exception
to < 0 & from != len	[error]	to left of bounds → expect an exception
len < to	[single]	to right of bounds → end with last element
to <= from	[single]	range [from, to] is empty → expect empty string
len < from < to	[single]	expect last element

(in green: 5 “From-To bullet points” of specification)