		Test Setup		Identifiers					Assertion			Test data					Test Structure				Legibility		
EvoSuite		Separate Test Setup	Instantiate in the same method	Clear (sensable) identifier	Vague Identifier	Short naming (identifier)	Long naming	Clear Test name	not simple (effective) assertion	weak assertion	Specific Assertion	Clear (Meaningful) Test data	Unclear test data	Clear (Concrete) Logic	Order of calls	Better Variable type	Concise Test	Simpler	Shorter (line lengths)	Code duplication	Annotation	Enough Seperation	No Seperation
Everything seems to be descriptive, the name of the test case helps to see that this is a no argument test case.	+			+															-		+		
No line break make it difficult to read; but the identifier names with index are acceptable; The test data of empty string is not meaningful				+									-										
It was understandable	+																						
The names person0 and person1 and boolean0 are not understandable, and the method name describes what the test does rather than what it tests.					-																		
The intent is clear from the name, but the method itself is hard to follow														-									
The test is slightly long so it isn't as easy to follow.																		-					
It is not clear at all what behaviour this test tests. There are way too many variables which are unnecessary and should be inlined (e.g. bigDecimal0). Identifiers are not descrptive,					-								-	-									
It is easy to make mistakes with such non human readable names. I might have separated the test of checking if the last name was correct unless if assumption was rhat creating two people will change the name of the other one but then I would have checked for person 0 not being overwritten. So not clear why the equals has been written.													-	-									
Unless this is a child of Elon Musk, I do not think that the names makes sense. Because of the structure of the test case it is somewhat inferrable what is being tested. However, it is not very clear what the purpose is.					-									-									
The test is simple, but assertion condition and actual assertion are split. This is not great for readability.										-													
The long lines and bad identifiers make it really hard to understand what is being asserted. The casting makes it a bit easier to understand what nulls are.					-														-				
The test is quite short and simple to understand. The variable names vet0/list1 are not descriptive and confusing. The different "paragraphs" make the test easier to understand.					-														+				
Variable naming is very basic and hard to understand. There are calls to all sorts of different objects/methods that further hinder the understanding of the test.					-										-								
The names of the variables are not at all descriptive, and the meaning of them is hard to figure out. Also, the data are hard to understand. It's hard to see what is part of the strings and what is actual java code, making it harder to read. The name of the test does not need to describe the fact that it creates two person instances, but rather that it compares two different people. (also this should be a separate test from the getLastName check)					-								-	-									
Splitting the assertion condition and actual assertion makes the thing that is asserted harder to relate. The test is short and simple however, and the name hints towards what is tested, although still slightly unclear. At this point I found out I accidentally flipped all the answers in this section. Specifically, all answers that say 'Somewhat disagree' are 'Somewhat agree' and all answers that say 'Strongly disagree' are 'Strongly agree' and vice versa. Please flip all these answers for this specific survey.										-													
Naming should be meaningful					-								-										
Easy to understand.	+																						
The test is quite simple overall but the inputs to the Person() call are long/confusing. boolean0/person0 are not descriptive variable names. The string in assertEquals should be made a constant to prevent the constant redefinition of strings					-								-										
It's clear that this test tests that two people with no arguments are equal. Identifiers, especially boolean0, are not clear. It would be better to do assertTrue(person0.equals(person1)).										-													
The choice of strings is very unintuitive, unsure why use strings like "Kt(tyZA._U*Ce4A". Also the name of the variable "boolean0" is very non-descriptive					-								-										
It's isn't clear why the naming of the test and the actual functions in the test makes sense. The test checks for that the two persons are equal but this doesn't necessarily show for example that person's created without arguments are valid.													-	-									
The identifiers are hard to understand. The test data is not really present a lot so I'm not sure. The test does make sense in the domain. The intent is easy to understand, but, I think this is only because of the limited amount of variables.					-								-	+									
the last assertion I have no idea what it does, that could benefit from a 'message' or knowing the getValue() method. Otherwise quite straight forward to see what should go on! Yeah the identifiers are not so descriptive, but clear in the sense of "there is a language 1 and there is a language 2, it is not actually relevant what precise languages they are"					-																		
Not readable, the test seems simple but very mechanic and not natural at all. The test data is not descriptive at all, a test intending to evaluate and "equals" method should use actual values. The styling is also not something I would like to see in my codebase.					-								-								-		
Same, the domain is difficult to understand, making it difficult to have an idea of what is being tested.																							