Note		Comments	Variables	Test Name	Test Data	Test Setup	Length	Lack of Documentation
Its all quite straightforward to understand, the comments clearly explain the behavior to test, the variables and the assertions, the names are also quite descriptive	∫	+	+	+				
I think the entire test is very clear, but I find it very confusing that the values in the last comment and last assertion don't match. Funnily though, the comment still makes it clear what the 21666 value was meant to be when you realize the mistake.	≈	-	+		+ (crossed out)			
It was difficult to figure out what was exactly different between both values.	∫				-			
Just missing context – and the comments are not good at describing missing context – for example, I see that the budgetUpToNow is 21666 from the assertion, but it is not clear to me WHY that should be the case.	≈	-			- (boxed)			
comments and methods name are easier to understand	∫	+		+				
This test case is really easy to understand. The naming is proper, the intent is defined from the name. The assertions can be derived from the name and the comments guide you trough the test. Nothing is difficult to understand.	∫	+	+	+	+			
Nicely written test, with comments being mostly unnecessary.	≈	+/- (circled)	+	+	+			+
All parts are easy to understand	∫	+	+	+	+			
This one looks pretty good. The first comment is excessive since these two lines of code are pretty obvious. And the last data is a bit strange (2000 calories per hour seems way to high, unless you're not like a 300 kg bodybuilder doing heavy athletic several times a day).	≈	+/-			+/-			
I believe this test is not very useful, but at least it is well organized.	???							
The whole test was written to be as understandable as possible. I am slightly distracted by the "default" and "custom" prefixes in the weapon variable names.	≈		+/-					
Some comments are not explaining everything that is important (what is the 0.1 in the second last assert?)	∫	-						
(small text)	∫	+/-						
It is not clear from the test case why the budget up to now should be 21666. It feels like an arbitrary number. Everything else is very clear and understandable (test case name, variables, comments).	∫	+	+	+	-			
Easy to read Clean distinction between setting up the test case, acting on the system, and asserting There is a bug in the comments, so I would not repeat test data or method names in the comments, it's not necessary. (Given, When, Then is enough)	∫	+/-	+	+	+			
Comments: the constants do not match those of the comments (which also throws me off guard during Task 2). Test data: what the constants mean can be a bit difficult to understand at first. Test name: some domain knowledge needed, but other than that it is clear what is done. Variables: very clear what they mean.	∫	-	+	+	-			
(small text)	∫	+/-	+		+			
1. Comments seems to be redundant in this method as involved operations are simple and naming is ok 2. toString is tested along with equals which might be a bit confusing given the test named testQuadisDifferentAndSingleValueRecursIsFalse	≈	+/- (circled)		-				
Test data values and comments do not cover why one weapon is custom, but another is default. Test name is too long, test mixes several tests. Comments too general. Var is simple to read, but there is unnecessary information	∫	-	+	-				
It's not quite clear why the following constant was chosen: 21666. Also seems that it's not necessary assertion: assertEquals(20003, budget.getCaloriesPerHour(), 0.01); Test name seems to be too cumbersome. Then the calories per hour is 2003.3 and the budget up to now is 60760 - why?	∫				-			
(long text)	∫					-	-	
Test data is not meaningful without entering the actual method implementation to check what they mean. The equals() method, which overrides the superclass method, confuses me a bit because I didn't imagine the overriden method behind. The assertFalse is a bit weird for group, because it negates an equality... is a fine line. Assertions in general are easy to grasp.	∫				-			
The part difficult to understand is WHY something is the expected value. I have to figure that out by myself, but that is difficult when the numbers are so large.	∫				-			-
I think it's easy to understand even lines up until getCaloriesPerHour, after that it gets unclear what the value 21666 is supposed to present the comment also has the wrong value of 60760, as in what is up until now in the context of a test case? The same goes for this comment, what does up to now mean in the context of a test case? :) When we get the budget up to now	∫	-			-			-
The comments expect you to be embedded into the code base and understand what they are talking about. Also using numbers and values in your comments are bad because they might not align when the code changes the comments are not updated.	∫	+/-						
variable names: digi, default and custom; test data: especially the string is self descriptive, however in my case because the enum things were so abstract with no context (meaning the values of integers are repeated 17 and 10k), it confused me to catch the difference. Also value of "Ninja Sword" would be different in default and custom variables. But overall, it was understandable.	∫		+		+			
Budget default behavior Comments are almost useless	∫	-						
(long text)	∫	-	-					
Most of the comments are not really useful because I just replicated the text into the assertions, where in case of the comments, they are not something obvious. The quote provides the information... test data...	∫	+/-						