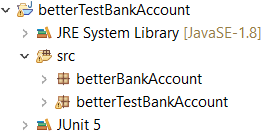
System Development 2

**Exercise: Unit and integration test**

**Exercise 1**

Create a Java project with this structure using the file BankAccountTesting.zip:



You have to **unit test** the class Account

1. First specify test cases for the class Account (read the comments for the specification of the methods).
2. Define Junit tests and run them (there is already an implementation of one test case in AccountTest.java).
3. Ensure all tests are successful

Test cases for the withdraw method

|  |  |  |  |
| --- | --- | --- | --- |
| Nr | balance | amount (hæv) | Resultat True/False - Balance |
| WM1 | 3000 | 55 | T - 2945 |
| WM2 | 15001 | 15001 | F - 15001 |

|  |  |  |  |
| --- | --- | --- | --- |
| Nr | Deposit amount | balance | Resultat True/False - Balance |
| DM1 | 3000 | 3000 | T - 6000 |
| DM2 | -1000 | 3000 | F - 3000 |

**Exercise 2**

Consider a system which is partially defined by the interaction diagrams below:

|  |
| --- |
|  |
|  |

The method ***public boolean addProduct(id, count)*** is defined in the class SaleCtr and returns a boolean indicating whether you get the items you would like to order.  
  
The rules for ordering products are the following:

* As a minimum order for 1 item and may not exceed ordered 20 of a product
* If the quantity ordered is greater than the number of stock, the order cannot be executed

You have to test the method addProduct(id,count).

**Define test cases**

1. Do you need stubs? Do you use unit test or integration test?
2. Find the equivalence classes!
   1. Hint: define equivalence classes for valid and invalid data for count and quantity (stock)
3. Define test cases covering all the equivalence classes (by filling in the table below).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test ID** | **Test Data**  id, count | **Execution condition**  Quantity in stock for the product **id**, … | **Expected output** | **Actual Result** |
| 1 | 3, 5 | 30 | Return: True  Quantity = 25 |  |
| 2 | 3, 0 | 30 | Return: False  Quantity = 30 |  |
| 3 | 3, -5 | 30 | Return: False  Quantity = 30 |  |
| 4 | 3, 21 | 30 | Return: False  Quantity = 30 |  |
| 5 | 3, -21 | 30 | Return: False  Quantity = 30 |  |

**Junit test**

1. Make a SalesTesting Java project based on SalesTesting.zip
2. Implement the previously defined test cases
3. Correct the errors in the code.   
   (Assume that the method findById of ProductCtr.java has already been tested.)

**Exercise 3**

Consider again the partial system based on SalesTesting.zip

Test the exceptions in the SaleCtr.java class.