Submitted by: Maria Pisnetz 313087744

Homework in Java: Banking Exercise

**IBANK**

This is a general Interface to serve the behaviors of all the bank, I built this as instructed in the exercise and the library import was suggested by intellij, so I accepted.

**IAccount**

This is a general Interface to serve the behaviors of all the accounts, I built this as instructed in the exercise.

**BasicAccount:**

The constructor was built as ordered in the exercise, in the beginning I declared the variables in the constructor as well as credit Limit and Balance, which are required. (It is a shame that you cannot built a modified constructor of other class, otherwise I just could inherit from the Standard account, and add the Withdrawal limitation)

In order to apply the Withdrawl test, I applied if conditions to test if the amount asked is more then the withdrawal limit, and also additional tests such as if this is in tact with the account balance and if the account that was entered makes sense (e.g. not negative))

Deposit, Currentbalance and account number was straight forward and I copy pasted them later on to the other bank account types.

**StandardAccount**

Mostly copy paste from the standard account, except that in this case the constructor has credit balance and doesn't have a withdrawal balance. As the credit balance changes and it is not zero, I had to change the if tests , that it will not be limited to withdrawl amount but just to the overall available amount that can be withdrawn which is Balance minus credit limit.

**PremiumAccount**

The most simple account, no limitations.

**Bank**

After inheriting the IBANK, I had to specify the methods. First, as requested I created an empty array that contains bank accounts.

Then OpenAccount function adds an account with it's type, number, and relevant details based on the type of account

CloseAccount is parsing over the list of accounts and looks for their bank account number (integer), if this is matches it goes through a list of ifs. If positive ,it will close, if negative it won't. There was not mentioned the case of balance 0 – so I added this as a case where it closes and says that it closed at 0.

Getallaccounts – Straightforward

GetAllaccounts in Debt – I parsed over every account and used the GetCurrentBalance to locate accounts under 0 balance.

GetAllaccountswithBalance -Same, just replaced the zero with the variable.

**Main**

I used the Main to test all and see that it is correct and works as expected