EXERCISE 2

ATM2

For this programming exercise, you will write a new program based on the program from exercise 1.

In exercise 1, the ATM managed a single account. In this exercise you will use the tables to manage the accounts of several customers and use methods.

For this purpose :

- Replace the account variable with an *accounts* array of type Double. For the exercise, all accounts will be initialized to *1200.0* CHF.
- Replace the pin code variable with a *codespin* array of type String. For the purpose of this exercise, all pin code values will be initialized to : *INTRO1234*.

Each customer will be represented by an identifier whose value is of type Int. The customer with identifier 0 will have the value of *codespin(0)* as a pin code and the value of *accounts(0)* as the amount available on his/her account. And in general, the customer whose identifier is *id*, will have the value of *codespint(id)* as pin code and the value of *accounts(id) as the* amount available on his account.

The two tables, *accounts* and *codespin,* have the same size and this size will be defined for this exercise from a variable *nbclients* which will be initialized to *100* in the program.

At the start of the programme, the user will have to identify himself with his login and PIN code.

Enter your login code >

If the user provides an ID number that is strictly greater than nbclients, the program must stop after indicating the message :

This identifier is not valid.

The user then has 3 attempts to enter the correct pin code:

Enter your pin code >

The pin code entry is repeated as long as the customer makes a mistake. Each time a mistake is made, a message is displayed indicating that there is an error and that the code must be re-entered:

Wrong pin code, you still have ... attempts >

After 3 unsuccessful attempts, the program displays the information message:

Too many errors, abandonment of identification

And asks again for a login and then a pin code.

If the login and pin code are entered correctly, he/she will then have access to the following menu:

Choose your operation:
1) *Deposit*
2) *Withdrawal*
3) *Account consultation*
4) *Changing the pin code*
5) *Finish*

Your choice :

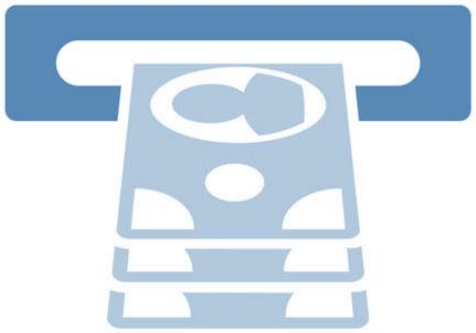
- Options 1, 2 and 3 work in the same way as in exercise 1. As long as the client has not chosen option 5) End, he/she can carry out transactions on his/her account.
- For option 5, the program does not stop. It displays the following message to the customer:

  End of operations, don't forget to collect your card.

  Then it asks again for an ID and pin code for another customer to enable him/her to perform banking operations.
- For option 4, the user can change his/her pin code:
  Enter your new pin code (it must contain at least 8 characters) > Enter your new pin code (it must contain at least 8 characters)

  The pin code must contain at least 8 characters. The entry must be repeated until you receive an information message:

  Your pin code does not contain at least 8 characters



The operations of deposit, withdrawal and modification of the pin code must each be carried out by a dedicated method:

1) Deposit operation

Method header: ***def deposit(id: Int, accounts: Array[Double]) : Unit***
With **id**, customer ID and **accounts**, the customer account table

The method should read the currency (CHF or EUR), the amount to be deposited - checking that it is divisible by 10 - from the keyboard, and add the amount to be deposited, if necessary converted from EUR to CHF, to the amount in the customer's account in the Accounts Receivable table from the customer's ID.

2) Withdrawal operation

Method header: ***def withdraw(id : Int, accounts : Array[Double]) : Unit***
With **id**, customer ID and **accounts**, the customer account table

The method should read the currency (CHF or EUR), the amount to be withdrawn - checking that it is divisible by 10 and below the limit - from the keyboard and subtract the withdrawal amount, if necessary converted from EUR to CHF, from the customer's account amount in the Accounts Receivable table using the customer's ID. It must also allocate and distribute the denominations for withdrawal.

3) Pin code change operation

Method header: ***def changepin(id : Int, codespin : Array[String]) : Unit***
With **id**, customer ID and **pin code**, the table of customer pin codes

The method should read the new pin code from the keyboard - checking that it contains at least 8 characters - and then update the customer's pin code in the pin code table with their ID.