Bank Account – 100 min

1. Create a Python class called BankAccount which represents a bank account, having these attributes: ID, account_number, owner_name and balance. **– 10 min**
2. Create a constructor with needed parameters and data types. **– 20 min**
3. Create a deposit () method which manages the deposit actions. **– 20 min**
4. Create a Withdraw () method which manages withdrawals actions. **– 20 min**
5. Create a bankFees() method to apply the bank fees with a percentage of 1% of every withdraw operation. **– 20 min**
6. Create a display () method to display account details. **– 10 min**

Credit Bank Account – 120 min

7. Let's program CreditBankAccount class which will have same attributes of BankAccount and add credit_balance credit_limit and interest_percent. **– 20 min**
8. Create a constructor with parameters for CreditBankAccount. **– 20 min**
9. Rebuild deposit () and withdrawal() so we can do these operation to credit account as well! (Do we need to rename these functions? Why?) **– 40 min**
10. Create an interest_cost() method that calculates the interest on the credit being allowed! **– 30 min**

11. Upgrade display to display all info om Credit and debit accounts. **– 10 min**

Bankcards: - 160 min

This bank would like to issue bankcards, so every client has possibility to order one card and connect it to his both credit and debit accounts simultaneously.

12. Let's program Bankcard class which will have attributes: card_number(16 digits) , random_pin_code_4digits , card_holder: name and last name , random_security_code_3digits , valid_month, valid_year , connected_bank_account , connected_credit_bank_account , fixed withdraw_fees as 1% of the amount when withdraw from debit account and 1.5% from credit accounts ,no fees for setting funds into accounts. **– 20 min**
13. Program with python the __init__ method. **– 20 min**
14. Program methode print_card() that print the card info **ps. none of private info should be printed. – 20 min**

Now let's program the next cases :

15. **Case 1**: At ATM machine a client likes to withdraw an amount of 10_000sek, and client has in his debit bank account 14_000sek and in credit one 19_000sek . Write python method withdraw_from_debit_or_credit () into Bankcard class to manage this case that we can ask for **both options**, deduct the right fees, and update

the right amount into the right bank account. **– 30 min**

16. **Case 2**: At ATM machine the client likes to withdraw an amount of 10_000sek, and client has in his debit bank account 4_000sek and in credit one 19_000sek. Write python method withdraw_from_credit() into Bankcard class to manage this case, describe the situation to the user with a message as : "*you don't have sufficient funds in your debit account so we withdraw from credit*" , deduct the right fees, and update the right amount into the right bank account. **– 30 min**

17. **Case 3**: At ATM machine the client likes to withdraw an amount of 10_000sek, and client has in his debit bank account 4_000sek and in credit one 9_000sek. Write python method withdraw_from_both () into Bankcard class to manage this case, deduct the right fees, and update the right amount into the right bank account. **– 40 min**