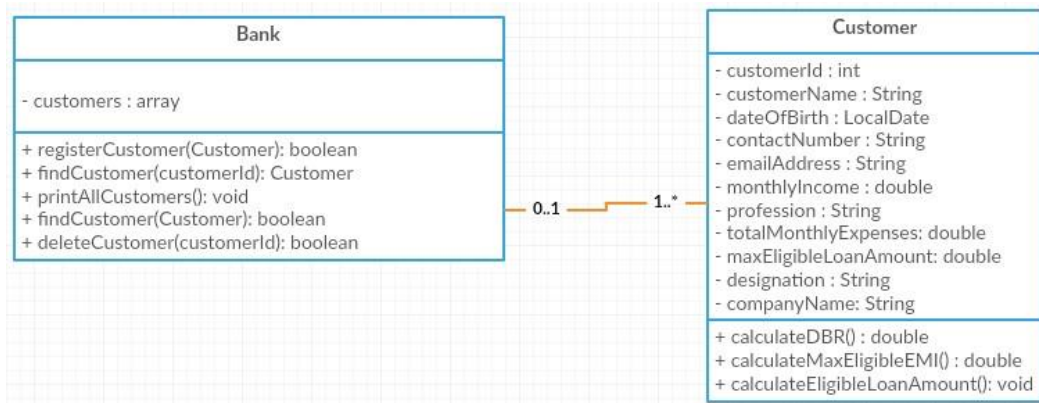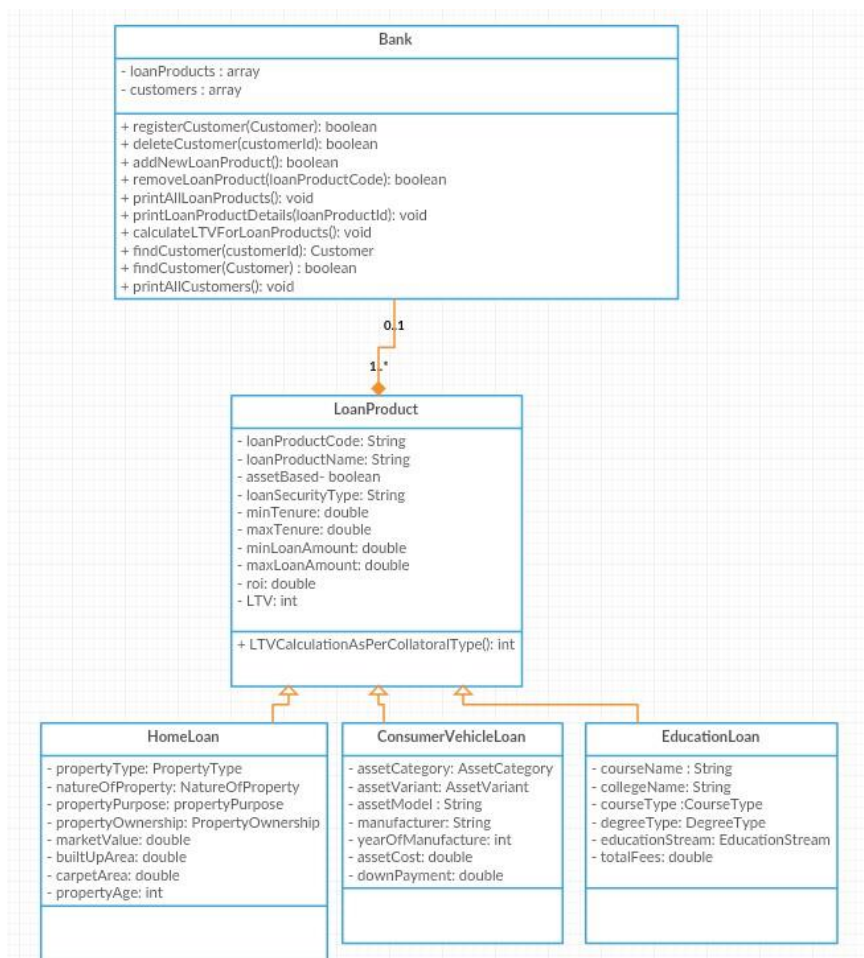1. Create a Bank class and create an array of Customer in the Bank class according to the given class diagram below:

Bank
- customers : array
+ registerCustomer(Customer): boolean + findCustomer(customerId): Customer + printAllCustomers(): void + findCustomer(Customer): boolean + deleteCustomer(customerId): boolean

0..1 ——— 1..*

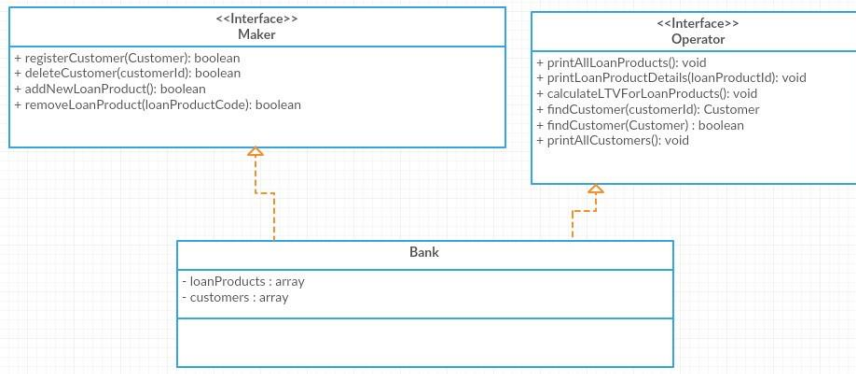| Customer |
| --- |
| - customerId : int
- customerName : String
- dateOfBirth : LocalDate
- contactNumber : String
- emailAddress : String
- monthlyIncome : double
- profession : String
- totalMonthlyExpenses: double
- maxEligibleLoanAmount: double
- designation : String
- companyName: String |
| + calculateDBR() : double
+ calculateMaxEligibleEMI() : double
+ calculateEligibleLoanAmount(): void |

The Customer Array should be able to hold object of Customers. Perform the functionalities given in the Bank class.

2. Extend the functionality of Bank and add an array of LoanProduct as per the diagram below:

Bank
- loanProducts : array - customers : array
+ registerCustomer(Customer): boolean + deleteCustomer(customerId): boolean + addNewLoanProduct(): boolean + removeLoanProduct(loanProductCode): boolean + printAllLoanProducts(): void + printLoanProductDetails(loanProductId): void + calculateLTVForLoanProducts(): void + findCustomer(customerId): Customer + findCustomer(Customer) : boolean + printAllCustomers(): void

0..1

1..*

LoanProduct
- loanProductCode: String - loanProductName: String - assetBased- boolean - loanSecurityType: String - minTenure: double - maxTenure: double - minLoanAmount: double - maxLoanAmount: double - roi: double - LTV: int
+ LTVCalculationAsPerCollatoralType(): int

HomeLoan
- propertyType: PropertyType - natureOfProperty: NatureOfProperty - propertyPurpose: propertyPurpose - propertyOwnership: PropertyOwnership - marketValue: double - builtUpArea: double - carpetArea: double - propertyAge: int

ConsumerVehicleLoan
- assetCategory: AssetCategory - assetVariant: AssetVariant - assetModel : String - manufacturer: String - yearOfManufacture: int - assetCost: double - downPayment: double

EducationLoan
- courseName : String - collegeName: String - courseType :CourseType - degreeType: DegreeType - educationStream: EducationStream - totalFees: double

The array must be able to hold any LoanProduct type objects. Provide the methods for the functionalities mentioned in the above diagram.

   a. addNewLoanProduct() method to be overloaded for all three types of LoanProduct.
   b. calculateLTVforLoanProducts() method iterates over the loanProducts array and call the abstract calcualateLTVforLoanProduct() method of the LoanProduct class.

3. Create two interfaces – Maker and Operator. The interfaces are implemented by the Bank class and the functionalities are as shown below:

The Bank class will implement these methods. We need to create a Bank object and return the same using the Interface reference so that only the provided methods in the corresponding interface are invoked. There must be a single Bank instance.