**Core Java**

**Topic: String**

Problems:

1) Given string “Algorithms”, return “go” and “Algo” using substring

2) Given two strings, compare if these two two strings are equal by comparing each character

3) Given string “<https://www.amazon.com/demo?test=abc>”, return [“https”,”www”,”amazon”,”com”,”demo”,”test”,”abc”]

4) Given a list of string array, combine them into one string sentence, return the string sentence

**Topic: final**

Problems: define a final class and final method and final variable, modify the value of the variable in final method

**Topic: Static**

Problems:

1. Given a database table “Book” with columns (id, isbn, name, author, publish date), define a java class that matches this table and then use a static block to initialize this table with some records
2. Define a Java class “BookSeller” and then define a static inner class “Book”, and use a static method “sellBooks” to initialize several books, and in the main method display all the books by calling the “sellBooks” method

Topic: **OOP**

Problems:

1. Define an interface “DatabaseConnection” and then define class “OracleConnection”, “MySqlConnection”, “SqlServerConnection”. They should all implements the “getConnection” method from the interface. The method should initialize data source and return a database connection.
2. Define an abstract class “CreditCard” which contains fields (holderName, cardNumber, accountBalance, cardType), and not-implemented method “isCardAcceptable” with argument cardType, and implemented method “payBill(double bill)”. Define two classes “VisaCard” and “MasterCard” both should inherit this “CreditCard” class and you should define constructor for both classes and implement the “isCardAcceptable” method.
3. Implement static and dynamic polymorphism.

**Topic: Serializable**

Problems: Define a “Employee” POJO class and make it serializable

**Topic: Design Pattern**

Problems

1. Create a singleton class called “AppleDesignerFactory”
2. Create a factory pattern called “CurrencyExchange” which takes in the country name and return the currency object for that country.
3. Implement the (in-class) PARKING LOT OOP design system -> your implementation should include main method and is runnable.

**Topic: Collection**

Problems:

1. (Set)Find true friends: Given two arraylists containing friend names, find the true friends that appear in both list.
2. (Map)Given a string, output duplicate characters and their counts
3. Use a map to simulate database table, key should be the primary key (assume only one column), value is the record, your simulation should include CRUD operation methods

**Topic: Exception Handling**

Problems:

1. Define two exceptions “CardTypeException” and “AddressException”. Create a separate class “ExceptionHandler” which contains one method “handleException”. The method takes input of cardType and address. If cardType is “AMEX”, throw CardTypeException. If address is outside US, return AddressException, for other errors, just return generic exception. Your exception should be caught and you should display message to tell which exception is returned.

**Topic: ExecutorService**

Problems:

1. Define a inner class A has method “getMethod()” which return string “A.getMethod” and another inner class B has method “getMethod()” which return string “B.getMethod”. write another method “runSameTime()” which should let A and B getMethod to run at the same time, but the “runSameTime()” method should return a string “B.getMethod A.getMethod”

**Topic: Java 8**

Problems:

1. Define an interface “CreditCard” which contains fields (holderName, cardNumber, accountBalance, cardType), and not-implemented method “isCardAcceptable” with argument cardType. Define two classes “VisaCard” and “MasterCard” both should inherit this “CreditCard” interface and you should define constructor for both classes and implement the “isCardAcceptable” method. Now Add a default method “payBill(double bill)” and static method “refund(double amount)” to the interface. Verify that VisaCard and MasterCard class can read these two methods (use Main method to verify).
2. Use functional interface to implement Java 8 stream.map() function. Define a “MyStream” class containing a functional interface and a static method “MyMap”.

In main method, test your code by calling: MyStream.MyMap(list, o -> o \*3) to multiple each element in the “list” by 3.

1. "walabcwalexywalxzsfwalmx” -- replace "wal" with "sams"
2. "Eclipse eclipse Eclipse eclipse amc clip ECLIPSE" – count the occurrence of each unique word (ignore case), return result as a map. For example (eclipse->5, amc->1, clip->1)