**Java Assessment - II**

**Provide solutions for the below problem statements.**

1. **Stream API and collection:**

Using below dataset, implement streaming intermediate and terminal operators for the below requirements

**Sample dataset.**

("Messi", 32, Gender. MALE, ("CF","CAM", "RF")),

("Griezmann", 28, Gender. MALE, ("CF", "CAM", "LF")),

("Arthur", 23, Gender. MALE, ("CM", "CAM")),

("Ter Stegen", 27, Gender. MALE, ("GK")),

("Puig", 20, Gender. MALE, ("CM", "CDM")),

("Jennifer", 29, Gender. FEMALE, ("CF", "CAM")),

("Jana", 17, Gender. FEMALE, ("CB")),

("Alexia", 25, Gender. FEMALE, ("CAM", "RF", "LF"))

1. Fetch the data gender value is matching with female and age more than 23.
2. Find the count of footballers who are females and age more than 23.
3. Get the position of male footballers who are the males and their age greater than 30 years.
4. Display the first there footballers who are male and age greater than 25.
5. Find the aggregate values of age of all male and female footballers.
6. Using list of elements (4, 1, 3, 7, 5, 6, 2, 28, 15, and 29) find first element which is greater than 5.
7. **JDBC**

In Mysql db create the table with below structure

Tbl Name: Orders

Field names: OrderId,Name,size,price,quantity,date

Using JDBC implement the following

1. Feed the data at least 5 records .

2. Fetch the total amount of orders delivered and minimum quantity for each name.

Once the above 2 completed

1.Feed the additional data with 3 records and having 1 duplicate.

In case of possible exceptions in both scenarios implement exception handling.

III) **JUNIT5**

1. create class String merge and method to implement combining two strings and return value.

b. create class for employees with method to search for name of employee in the given list of employees if exists or contains in the array list created.

If name found then display as FOUND else NOT FOUND and return the result.

Create JUNIT Test cases for above both scenarios, using Junit annoatations @display,@disabled,@Test,@Beforeall,@afterall and assertion assertequals().

**IV)** **USING Collection framework** , derive solution for the below problem statement:

create Seat and Theatre classes with below fields information.

Seat --> SeatNo, Price and Reserved

Theatre--> name and ArrayList of seats

--> constructor (name,noOfRows,SeatsperRow)

Business requirement to implement:

Build solution to fill the Theatre with seats of 10 rows and 15 seats per row

The seats name must be A01, A02, A03, ..., H15 as 'A' is the row and '01' as the seat number

The front seats as A01.....A05 are more expensive than the others seats.

**1. Create method that makes you reserve a seat**

**2. Create a method that makes you cancel seat reservation**

V. **Multithreading:**

Banking system shared resources like performing different operations like withdraw(),deposit(),Enquire() when ever we want to perform these operations at same time on same account causes data inconsistency and deadlock situations.

Provide solution with multithreaded program which will avoid deadlock situations.