**Important Instructions:**

* Please read the document thoroughly before you code.
* Import the given skeleton code into your Eclipse.
* Do not change the Skeleton code or the package structure, method names, variable names, return types, exception clauses, access specifiers etc.
* You can create any number of private methods inside the given class.
* You can test your code from main() method of the program

**Time: 2 hours**

**Assessment Coverage:**

* **Date & Time API,**
* **JDBC API, NIO/File API**
* **Lambda Expression & Functional Interface**

Jerry is working a controller of examination in a small institution. As a COE, Jerry has to prepare examination schedule report for each assessment. Hence he created a csv file which has all assessment information in certain format (refer **data.txt file**). Now he requires an application which needs to do,

1. Read all lines from file (using NIO/File handling)
2. Convert each line into Assessment object and collect as List (using Java 8 Streams)
3. Upload assessments list into database table “assessment” (using JDBC API)
4. Search assessment details using assessment code and prints (using Date & Time API)

You are required to automate the above procedures using the given code skeleton

**Technical Requirements:**

You are required to develop an App which can provide the following service.

**Requirement 1: Implements and Use functional Interface “Function”**

You have been given a class called “**GenerateAssessmentFunction**” which implements “**java.util.function.Function**” interface. You are required to override the unimplemented function “apply” and implement the following logic

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class Name** | **Method Name** | **Input** | **Output** | **Logic** |
| GenerateAssessmentFunction | apply | String | Assessment | * Split the input string using comma (,). * Create Assessment object * Set assessment instance’s properties using the split strings and return the instance. |

**Design Constraints:**

* Method must be overridden from Function<String, Assessment>
* **Should NOT** use try…catch();

**Requirement 2: Load File Data**

You have been given a class called “**FileUtil**”. You are required to implement the following logic in the given skeleton code.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class Name** | **Method Name** | **Input** | **Output** | **Logic** |
| FileUtil | loadData | String | List<Assessment> | * Use NIO Paths to **get Path** using given input argument (or Create **BufferedReader** for the given file path) * Get File data as **Stream<String>** * Use **streams**, map each line with “**Requirement 1**” and **collect** them as List<Assessment> |

**Design Constraints:**

* Method must be **public static**
* You may use Java 8 & NIO features or normal IO API to handle file
* Must use streams, Collector features to convert streams into List
* **DO NOT** handle IOException (Do not use try…catch)

**Requirement 3: Store List<Assessment> in DB & Find Assessment using code:**

DatabaseUtil.getConnection() has been given as part of the skeleton file, which return database connection. You need to implement the following business logic in “**AssessmentDAO**” class as follows,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ClassName** | **Method Name** | **Input** | **Output** | **Logic** |
| AssessmentDAO | uploadAssessments | List<Assessment> | int | * Use prepared statement to insert each assessment * Iterate input parameter List, store in assessment table, and for each insertion increment row count * Return total number of rows affected |
| AssessmentDAO | findAssessment | String | Assessment | Implemented as part of Skeleton |

**Design Constraints:**

* Use DatabaseUtil to retrieve connection and create prepared statement
* For insertion, you may use batch update/normal execute update, but return total number of rows affected
* Do NOT handle exception

**Requirement 4: Display Assessment Details**

**Assessment** model class has been given with properties, constructors, getters & setters part of skeleton. You are required to write the display logic as described below,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ClassName** | **Method Name** | **Input** | **Output** | **Logic** |
| Assessment | printDetails | N/A | void | Display assessment details in following format,  Assessment Code: ASEJE025  Title: Technical-JSP/JSTL  Assessment Date: 23-May-2020  Start Time: 14:30  End Time: 18:00  Result Date: 01-Jun-2020 |

**Design Constraints:**

* Use **DateTimeFormatter** pattern to display the Date & Time fields as mentioned above.

**General Design Constraints:**

* Ensure that all the Java Coding Standards are followed.
* Assume that the method inputs are valid always, hence exceptional blocks are not needed to be included in the development.