**DO NOT SHARE THIS EXAM WITH ANYONE ELSE**

**FAILING DOING THIS WILL RESULT IN A SERIOUS PENALTY**

**Submission: Students can submit up to 10 minutes late without penalty. After that, every 1 hour late will result in 30% deduction.**

**Final Test**

**Total hours:  3 hours**

Rules

1. This is an **open book and INDIVIDUAL.** Students can use the Internet and Canvas to support their answers.
2. Exchanging answers or ideas between students, online chatting or posting questions to get support from other people, family, friends, or verbal chat during the test time is strictly prohibited, and will be considered as cheating
3. Use your own words. Do not just copy and paste from the Internet. Failing doing this can be considered as plagiarism.

Notes

1. Read through the test carefully and plan your answers. Don’t jump to answer them right away. Students can ask for clarity via Teams or email (I prefer Teams).
2. There are 2 parts of the final test: Theory and Programming.
3. Students provide answers for theoretical questions in a doc file named StudentName\_StudentID\_Theory.doc
4. For the programming question, provide the answer in a folder with all source code and related files. Zip the folder and doc into a folder name StudentName\_StudentID.zip.
5. Submit the zip file to Canvas.

**Theory (60 marks)**

**(10 marks) Question 1:** What is Dependency Injection? And why do we call it Inversion of Control? Why do we need Dependency Injection? List a few frameworks that support Dependency Injection in Java.

**Answer:**

* Dependency injection is “design pattern that allows a program design to follow the dependency inversion principle”. The client will delegate the XOR code, the injector and the responsibility of providing its dependencies. In other words, dependency injection allows the construction of dependent objects outside of a class in different paths and then injection of the object to an external entity. We then change the construction and binding of the dependents object outside of class that is depended on them.

Inversion of Control is the design process of externalizing, the construction and management of objects. It basically says that your application's going to outsource the creation and management of the objects, and that outsourcing will be handled by a object factory.

* The reason why Dependency injection is one of two ways of how to remove dependencies in the code. It is helpful when we change the configuration after compile-time. Additionally, it supports unit testing process as it makes it easier to inject stubs or mocks.

Practically speaking, there are nothing the developer cannot do without a service locator. In case that we cannot specified the number of instances we need for a specific interface: A Dependency Injection framework always injects only one instance per parameter, hence most of DI framework provides a service locator to support this.

For example, of why we need to use dependency injection in real world. Let’s assume that we have a car class which includes various objects such as wheels, engine, etc. The car class is responsible for the creation for all the dependency objects. But what if in the future we will decide to change the brand or type of wheel using for the car? If we don’t use dependency injection, we need to recreate the car object with new wheel brand, for instance Toyota dependency. Here is where DI is taken into account as we can change the wheels at runtimes since it is a dependency that can be injected at runtime rather than compile time.

In conclusion, the DI is the middleman in the code who does all work of creating the preferred object and providing it to the specific class. It makes the class – for instance car class be independent from the creation of other object such as wheels, battery, engine, etc.

* List a few frameworks that support Dependency Injection in Java.
* CDI (Contexts and Dependency Injection for Java EE (CDI) 1.0
* Weld
* Spring
* Guice
* Play framework
* Salta.
* Glassfish HK2
* Dagger
* Managed Extensibility Framework.

**(10 marks) Question 2**: What is Object-relational Mapping (ORM)? List some popular ORM tools. Compare between ORM framework and traditional JDBC.

* Object Relational Mapping or ORM is a system that is an implementation of the responsibility of mapping the object to Relational model. Its main responsibility is to store Object Model data into Relational Model. Additionally, ORM will be used to further read the data from the Relational Model into Object Model. Furthermore, when working with object-oriented programming to persist data in RDBMS, there is mismatches between object model and relational model if dealing with traditional techniques like JDBC. Hence, ORM can fill the gap of mentioned mismatches between Object model and relational model.
* Some popular ORM tools online:

1. [**Hibernate**](https://docs.jboss.org/hibernate/orm/current/userguide/html_single/Preface.html) – Open Source

Hibernate ORM allows developers to more easily build application that data outlives the application process. As an ORM framework, Hibernate is involved in data persistence since it applies to relational database (via JDBC).

1. [**Top Link**](https://docs.oracle.com/middleware/1213/toplink/solutions/index.html) – By Oracle

TopLink is another ORM tools produced by Oracle and is a part of Oracle’s OracleAS, WebLogic, and OC4J servers. It is an object-persistence and object-transformation framework. It provides development tools and run-time functionalities which is an ease to development process and support the increasement functionality.

1. [**Eclipse Link**](https://www.eclipse.org/eclipselink/documentation/2.7/concepts/toc.htm) – Eclipse Persistence Platform

EsclipseLink helps any relational database that is compliant with SQL and has a compliant JDBC driver. It contains extension of support package for several database platforms, and mainly consists of providing native sequencing support, the creation of schema and c ertain database functions.

1. [**Open JPA**](https://openjpa.apache.org/builds/3.0.0/apache-openjpa/docs/main.html) – By Apache

Apache OpenJPA is a Java endurance project of The Apache Software Foundation that is a stand-alone POJO usage of persistence layer or integrated into any Java EE compliant container and other frameworks such as Tomcat and Spring.

1. [**MyBatis**](https://mybatis.org/mybatis-3/index.html) – Open Source – Formerly known as iBATIS

MyBatis is persistence framework with support to be used for custom SQL, stored procedures and advanced mapping. It can eradicates almost all of the JDBC code and manual setting of parameter and retrieval of results.

Compare between ORM framework and traditional JDBC.

|  |  |  |
| --- | --- | --- |
|  | ORM | JDBC |
| **Definition and implementation** | ORM has responsibility for the establishment of database connection, unlike JDBC. ORM enhances the use Query Language to communicate with the database an execute the queries. Later, ORM maps itself the results to corresponding Java objects. The mapping to the objects is done automatically and depends on the application properties from the ORM configuration XML file. | The JDBC includes many methods that establish the connection to database server, send queries to create a table, update, delete and insert records, date in a database, retrieve and process the results obtained from a database. Although, there are many framework that are constructed to work for easier work with database, they al contains the JDBC inside the hood. |
| **Workflow** | Initially, the database connection can be get from the application using the Session. The session also supports the add, update, delete and object persistent retrieval processes. The instance of a session is creating with Session factory interface. There is only one session factory for each database server or infrastructure. For the case, it is necessary to create one session factory for MySQL and another one for Oracle (it is redundant to have more than 1 session factory for one database alone). | First, we have to open a database connection with the support of JDBC driver. Then after the connection is established, we can send the query to database using a JDBC driver. Then we run the execution of queries and get the result. In this workflow the data are sent to the application via driver manager. It deals with data procession when the application gets the results. Manually we copy values from the set result to the corresponding application’s object. Lastly, the connection to database must be close after usage. |
| **Advantages and Disadvantages** | Advantages:   * Let business code access object rather than the database’ table. It supports to hide the details of SQL queries from the object-oriented logic. * ORMs are based on JDBC ‘under the hood. * There is no need to deal with database implementation. Entities transform the application based on the business concept not on the database structure. All of the transaction management is get with automatic key generation.   Disadvanatges:  ORM has slow performance in case of large batch updates. | Advantages:   * It is good as we usually have clean SQL processing, great performance even with large data. * It is good for small application. * It has a simple syntax that is not difficult to learn and use for the application to see immediate result.   Disadvantages:  JDBC things is not good for large project as it would make the project to be more complex to build. If there is a large programming overhead, no encapsulation, it will be hard to implements the MVC concepts.  JDBC has SQL queries that are DBMS specific |

<https://www.h2kinfosys.com/blog/jdbc-vs-orm-framework/>

**(25 marks) Question 3:** List at least 5 design patterns that we have learned in this course, and provide an example for each of them (For the snippets of codes, you can take screenshots, edit, and include them in your doc) .

* 1. **PROTOTYPE**

**Prototype** is a creational design pattern that lets you copy existing objects without making your code dependent on their classes.

Text

Description automatically generated

* 1. **DEPENDENCY INJECTION**

**Dependency injection** is a technique in which an [object](https://en.wikipedia.org/wiki/Object_(computer_science)) receives other objects that it depends on. These other objects are called dependencies.

The intent behind dependency injection is to achieve [separation of concerns](https://en.wikipedia.org/wiki/Separation_of_concerns) of construction and use of objects. This can increase readability and code reuse.

Text

Description automatically generated

* 1. **BUILDER**

**Builder** is a creational design pattern that lets you construct complex objects step by step. The pattern allows you to produce different types and representations of an object using the same construction code.

Text

Description automatically generated

* 1. **FACTORY METHOD**

**Factory Method** is a creational design pattern that provides an interface for creating objects in a superclass, but allows subclasses to alter the type of objects that will be created.

Text

Description automatically generated

* 1. **DECTORATOR**

**Decorator** is a structural design pattern that lets you attach new behaviors to objects by placing these objects inside special wrapper objects that contain the behaviors.

Text

Description automatically generated

**(10 marks)** Question 4. What are Java IO streams? List some stream implementations and discuss about their abilities.

http://tutorials.jenkov.com/java-io/streams.html

* Java IO streams is data flow, or extended collection of bytes that allows user to either write to or read from. Streams basically establish of connection to a data source or specific data destination such as file or network connection.
* Common usage of stream in Java is for input/output (I/O) based operations such as devices, files, socket (streams over a network), object serialization, pipes between thread etc.
* Additionally, by using streams the considerable amount of reuse is achieved when implement these different functionalities, which makes it easier for the programmer since common operations are used for the different tasks (polymorphically).
* The abstract class name “java.io.OutputStream” allows bytes and writes them to a data destination. For instance, when wiritng the component that is in need of writing output to a stream, the use of OutputStream is preferred to its subclasses.
* The abstract class “java.io.InputStream” read bytes from a data source. For instance, when writing a component that is in need of reading input from a stream, the dependency of components on an InputStrean is preferred to its subclasses(e.g FileInputStream).It enables code to work with all types of input stream.
* The abstract class “BufferedInputStream” is a combination effect when combining input and output streams into chains.
* **Stream implementations:** 
  1. PushbackInputStream: Is an implementation of InputStream that support user to parse data from an InputStream. As there is time that we need to read ahead a few bytes to see what is coming in the program flow, and before user can determine how to interpret the current byte. The PashBackInputStream supports us to do that, it basically helps user to push the rewad bytes back into the stream. These bytes will be reread the next time user call read().

PushbackInputStream input = new PushbackInputStream(

new FileInputStream("c:\\user\\student.txt"));

int data = input.read();

input.unread(data);

“How it works: the call to read() reads byte just like from an InputStream, the call to unread() pushes a byte back into the PushBackInputStream.” (Oracle documentation)

* 1. ObjectInputStream has the ability to read Java object from an InputStream instead of just raw bytes. This implementation can be used to read objects when user wraps an InputStream in a ObjectInputStream. The condition of this implementation to successfully execute is that the bytes read must represent a valid serialized Java object.

ObjectInputStream objectInputStream =

new ObjectInputStream(new FileInputStream("product.data"));

Product object = (Product) objectInputStream.readObject();

//etc.

objectInputStream.close();

The first and foremost condition to run this code is to make sure that the object read must be an instance of Product and must have been serialized into the file “ product.data” via an ObjectOutputStream.

* 1. DataInputStream has ability to read Java primitives via ‘the DataInputStream” – it read data(numbers) rather than the bytes itself. This class expects the multi byte primitives to be constructed in the network byte order such as data that includes int, long, float, double etc. When using this class there is one note that user must know for sure and ahead of time hat datat types to read and in specific sequence as there is no way to differentiate a valid int value of -1 and end-of-stream marker.

DataInputStream dataInputStream = new DataInputStream(

new FileInputStream("somethingbinary.data"));

int aByte = input.read();

int anInt = input.readInt();

float aFloat = input.readFloat();

double aDouble = input.readDouble();

//etc.

input.close();

**(5 marks) Question 5:** Tell me some topics in this course that you like, and the field of Computer Science you are into, and why.

My top favorite topics in this course list:

1. Abstract classes and Interface

I like this topic as before I have learnt in the Software Engineering Fundamental course about Interface and abstract class, even design a class diagram for specific software application. However, I still cannot understand how it actually works in practice. I also misunderstand and mismatch the implementation of abstract class and interface. After I learn this topic, I felt satisfied as my concern is now no longer exist. I can describe and demonstrate in code and even explain for my friend who begin to learn Java about these usages.

1. Software Quality and OOP.

This topic is the first week topic and the lecture time was so impressive that keeping me from remember the day and the knowledge I absorbed from that day. I can have idea how real-world software development really is with the practical sharing from Mr.Minh which makes all of us have more insight and prepare for necessary skills before we dive into the working environment. Additionally, I also learnt about purpose and practical case of encapsulation, information hiding and know how to measure the quality of specific software.

1. OOP for enterprise:

In this topic which is week 5, I have known the new concept which is tight-coupling and loose coupling with insights and live code about these concepts which was quite amazing. Furthermore, the term dependency injection is analyzed in detail with funny example which hooked me into the lecturer time. Before I cannot understand what Dependency Injection is although I have heard about it usually.

I am into Data Analytic, I have tried to build a project with my friend on cancer prediction machine learning project and I am quite into the beauty of data and process of cleaning the data before we can put into the machine learning model.

**Programming (40 marks)** 10 + 10 + 10 + 10

Library system is a very important tool to support researching. Use Spring framework to build a backend component for that system. Students can use Spring with or without SpringBoot, Hibernate or JPA, and SpringMVC RESTful API architecture.

The system allows users to manage sub-libraries. Each sub-library contains a list of authors. Each author is associated with 1 or more books. For the sake of simplicity, each book belongs to only 1 author.

Information about a sub-library is id (integer), subject (string)

Information of an author is id (integer), and name (string), and academic credentials (string)

Information of a book is id (integer), name (string), and date of creation (date)

* 1. Build entities class for sub-library, author, and book. Make sure the relationships between authors and books are **one to many and bi-directional.** The relationship between sub-library and author is **one to many and uni-directional.**
  2. Build a REST controller named **LibraryController**to perform **add and update** operations on sub-libraries. Also, a sub-library can be searched by subject and the search results will be ordered (descending or ascending)
  3. Build a REST controller named **AuthorController**to perform **add, update,** and **delete** operations on authors. Also, an author can be searched by name or academic credentials and the search results will be ordered (descending or ascending)
  4. Build a REST controller named **BookController**to perform **add, update,** and **delete** operations on books. Also, a book can be searched by name or the created date and the search results will be ordered (descending or ascending)

Students must define necessary request params or path variables in the controllers in order to provide above-mentioned features.

Students must build necessary repositories and service classes to be called by these controllers.

Test your system using Postman and show me the testing results in a doc.

**LibraryControler Endpoint :**

**For create or update**

\*/api/library/save

Postman example :

Call 1 create Fictional library

Graphical user interface, text, application, email

Description automatically generated

Call 2 Create Non fictional library

Graphical user interface, text, application, email

Description automatically generatedCall 3 : update fictional library

Graphical user interface, text, application, email

Description automatically generated

**For listing results :**

api/library/search

Call 1 : Search all

Graphical user interface, text, application, email

Description automatically generatedCall 2 : Search only library which start by MY

( adding a parameter with key subject and value my)

Graphical user interface, text, application, email, Teams

Description automatically generated

Call 3 : by default it is ordering by name Asc so we do the same call ( still only the library starting by my but by order desc)

( we add the parameter descOrdering to true)

Graphical user interface, text, application, email, Teams

Description automatically generated

**AuthorControler Endpoint :**

**Add and update :**

Same as before only one endpoint :

*api/*author/save ( if we provide ID is updating it, if we don t provide id or id doesn t exist it is creating it)

Call one create  with link to a library for example fictional

Graphical user interface, text, application, email

Description automatically generated

Let say I made a mistake so I will update it

Graphical user interface, text, application, email

Description automatically generated

**Listing result search**

*api/*author/search

*Graphical user interface, text, application, email, Teams

Description automatically generated*

Same as before we can add parameter to search by name and creditential and ordering by name and creditential desc

So one example by creditential with Name desc

List author and see all his book :

Graphical user interface, text, application, email, Teams

Description automatically generated

You can see that now Harry potter is linked with Jk rollings

*Graphical user interface, application, Teams

Description automatically generated*

**Delete :**

*api/*author/delete

***Graphical user interface, text, application, email

Description automatically generated*BookControler Endpoint :**

Update and save :

api/book/save :

Graphical user interface, text, application, email

Description automatically generatedTo update it works the same as before : ( put the id and it will update )

**Listing Results :**

api/book/search

I want to search on the book I just created  ( harry potter )

Graphical user interface, text, application, email, Teams

Description automatically generatedSame as before we can add ordering by desc and filtering by created after a certain date

Graphical user interface, text, email, website

Description automatically generated

api/book/delete

Graphical user interface, text, application, email, website

Description automatically generated

\*\*\*\*\*

**References:**

<https://www.h2kinfosys.com/blog/jdbc-vs-orm-framework/>

<https://rmit.instructure.com/courses/86345/pages/w5-oop-for-enterprise?module_item_id=2982034>

<https://rmit.instructure.com/courses/86345/pages/w8-lecture-slide?module_item_id=2982054>

<https://rmit.instructure.com/courses/86345/pages/w1-software-quality-and-oop-design?module_item_id=2981996>

<https://rmit.instructure.com/courses/86345/pages/w3-abstract-classes-and-interfaces?module_item_id=2982016>

Design Patterns aresimpler than youthought

<http://tutorials.jenkov.com/java-io/pushbackinputstream.html>

<http://tutorials.jenkov.com/java-io/arrays.html>

<http://tutorials.jenkov.com/java-io/streams.html>

<https://www.tutorialsteacher.com/ioc/dependency-injection>

openjpa.apache.org

<https://mybatis.org/mybatis-3/index.html>

<https://stackify.com/dependency-injection/>