# Assignment 03 Application Design: Patterns and Frameworks 44642

Answer **all** the questions below. In your answer to each question, explain which sample code or image is preferable.

1. **What are generics?**

Java **Generic** methods and generic classes enable programmers to specify, with a single method declaration, a set of related methods, or with a single class declaration, a set of related types, respectively.

Generics also provide compile-time type safety that allows programmers to catch invalid types at compile time.

Text

Description automatically generated

1. **Can we change the scope of the overridden method in the subclass for private, public, default and protected? Explain how can it be changed for each scope.**

Yes, an overridden method can have a different access modifier but it cannot lower the access scope.

The following rules for inherited methods are enforced -

* Methods declared public in a superclass also must be public in all subclasses.
* Methods declared protected in a superclass must either be protected or the public in subclasses; they cannot be private.
* Methods declared private are not inherited at all, so there is no rule for them.

1. **What is the covariant return type?**

Java allows for Covariant Return Types, which means **you can vary your return type as long you are returning a subclass of your specified return type**. Method Overriding allows a subclass to override the behaviour of an existing superclass method and specify a return type that is some subclass of the original return type.

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

1. **Can we override the static and private methods? Why?**

**You cannot override a private or static method in Java**. If you create a similar method with the same return type and the same method arguments in the child class, it will hide the super class method; this is known as method hiding. Similarly, you cannot override a private method in sub class because it's not accessible there.

1. **Difference between String Buffer and StringBuilder?**

**StringBuffer:**

StringBuffer is synchronized i.e. thread safe. It means two threads can't call the methods of StringBuffer simultaneously. StringBuffer is less efficient than StringBuilder. StringBuffer was introduced in Java 1.0.

**StringBuilder:**

StringBuilder is non-synchronized i.e. not thread safe. It means two threads can call the methods of StringBuilder simultaneously. StringBuilder is more efficient than StringBuffer. StringBuilder was introduced in Java 1.5.

Graphical user interface, text, application

Description automatically generated

1. **Difference between a String class and String Buffer?**

Strings, which are widely used in Java programming, are a sequence of characters. In Java programming language, strings are treated as objects. The Java platform provides the String class to create and manipulate strings.

Whereas, StringBuffer class is a thread-safe, mutable sequence of characters.

* A string buffer is like a String, but can be modified.
* It contains some particular sequence of characters, but the length and content of the sequence can be changed through certain method calls.
* They are safe for use by multiple threads.
* Every string buffer has a capacity.

Graphical user interface, text, application

Description automatically generated

1. **Can we declare the constructor as final?**

No, a constructor can't be made final. A final method cannot be overridden by any subclasses.

Graphical user interface, text, application, email

Description automatically generated

1. **Can we try without a catch block in java?**

Yes, It is possible to have a try block without a catch block by using a final block. As we know, a final block will always execute even there is an exception occurred in a try block, except System.

Graphical user interface, text, application

Description automatically generated

1. **What is try with the resource?**

The try -with-resources statement is **a try statement that declares one or more resources**. A resource is an object that must be closed after the program is finished with it. The try -with-resources statement ensures that each resource is closed at the end of the statement.

Graphical user interface, text, application

Description automatically generated

1. **Can we modify the throws clause of the superclass method while overriding it in the subclass?**

Yes, it is possible to modify the superclass method's throws clause when overriding it in the subclass. If a super class method is throwing unchecked exception, then it can be overrided in the sub class with same exception or any other unchecked exceptions but can not be overrided with checked exceptions.

1. **What is an association, aggregation, and composition in UML?**

A relationship between two objects is referred to as an association, and an association is known as composition when one object owns another while an association is known as aggregation when one object uses another object.

1. **Difference between final, finally and finalize()?**

Final keyword is used with the classes, methods and variables. Finally block is always related to the try and catch block in exception handling. finalize() method is used with the objects.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. **Difference between Vector and ArrayList?**

**ArrayList:**

ArrayList is not synchronized. ArrayList increments 50% of the current array size if the number of elements exceeds ts capacity. ArrayList is not a legacy class. It is introduced in JDK 1.2. ArrayList is fast because it is non-synchronized. ArrayList uses the Iterator interface to traverse the elements. ArrayList perfomance is high . Multiple threads is allowed.

**Vector:**

Vector is a legacy class. Vector is slow because it is synchronized, i.e., in a multithreading environment, it holds the other threads in a runnable or non-runnable state until the current thread releases the lock of the object. A Vector can use the Iterator interface or Enumeration interface to traverse the elements. Vector performance is low . only one threads are allowed.

Graphical user interface, text, application

Description automatically generated

1. **What are the different ways to make ArrayList methods synchronized?**

Implementation of ArrayList is not synchronized by default. It means if a thread modifies it structurally and multiple threads access it concurrently, it must be synchronized externally. Structural modification implies the addition or deletion of element(s) from the list or explicitly resizes the backing array. Changing the value of an existing element is not a structural modification.

**There are two ways to create a Synchronized ArrayList.**

1. Collections.synchronizedList() method.   
2. Using CopyOnWriteArrayList.

1. **Difference between Hash table ana d Hash Map?**

HashMap allows one null key and multiple null values whereas Hashtable doesn't allow any null key or value. HashMap is generally preferred over HashTable if thread synchronization is not needed.

Graphical user interface, text, application

Description automatically generated

1. **In Java 8, explain how Hasp Map internally works?**

Internally HashMap uses a hashCode of the key Object and this hashCode is further used by the hash function to find the index of the bucket where the new entry can be added. HashMap uses multiple buckets and each bucket points to a Singly Linked List where the entries (nodes) are stored.

Graphical user interface, text, application

Description automatically generated

1. **Difference between fail-fast and fail-safiteratorsor?**

Fail-Fast systems abort operation as-fast-as-possible exposing failures immediately and stopping the whole operation. Whereas, Fail-Safe systems don't abort an operation in the case of a failure. Such systems try to avoid raising failures as much as possible.

1. **Can we start the thread twice?**

No. After starting a thread, it can never be started again. If you does so, an IllegalThreadStateException is thrown. In such case, thread will run once but for second time, it will throw exception.

Graphical user interface, text, application

Description automatically generated

1. **What are the different ways to create a thread in java? Which one is preferred?**

Extends Thread class. Create a thread by a new class that extends Thread class and create an instance of that class. ...

Implementing the Runnable Interface. The easiest way to create a thread is to create a class that implements the runnable interface.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. **What are the different states a thread will go through?**

A thread state. A thread can be in one of the following states:

* [NEW](https://docs.oracle.com/javase/7/docs/api/java/lang/Thread.State.html#NEW)  
  A thread that has not yet started is in this state.
* [RUNNABLE](https://docs.oracle.com/javase/7/docs/api/java/lang/Thread.State.html#RUNNABLE)  
  A thread executing in the Java virtual machine is in this state.
* [BLOCKED](https://docs.oracle.com/javase/7/docs/api/java/lang/Thread.State.html#BLOCKED)  
  A thread that is blocked waiting for a monitor lock is in this state.
* [WAITING](https://docs.oracle.com/javase/7/docs/api/java/lang/Thread.State.html#WAITING)  
  A thread that is waiting indefinitely for another thread to perform a particular action is in this state.
* [TIMED\_WAITING](https://docs.oracle.com/javase/7/docs/api/java/lang/Thread.State.html#TIMED_WAITING)  
  A thread that is waiting for another thread to perform an action for up to a specified waiting time is in this state.
* [TERMINATED](https://docs.oracle.com/javase/7/docs/api/java/lang/Thread.State.html#TERMINATED)  
  A thread that has exited is in this state.

1. **What is Serialization? How do we achieve it?**

Serialization is the process of converting an object into a stream of bytes to store the object or transmit it to memory, a database, or a file. Its main purpose is to save the state of an object in order to be able to recreate it when needed. The reverse process is called deserialization.

1. **What is the immutable class? Is the String class immutable?**

In Java, **String objects are immutable**. Immutable simply means unmodifiable or unchangeable. Once String object is created its data or state can't be changed but a new String object is created.

Graphical user interface, application, Word

Description automatically generated

1. **Do immutable class thread-safe? If yes then how?**

An immutable object is one whose state can't be changed once the object is created. Immutable objects are, by their very nature, thread-safe simply because threads have to be able to write to an object's instance variables to experience a read/write or write/write conflict.

Declare the class as final so it can't be extended. Make all fields private so that direct access is not allowed. Don't provide setter methods for variables.

1. **Can we call the garbage collector explicitly? Will it trigger the garbage collector?**

The answer is Yes, the garbage collector can be called explicitly. But the point here to note is that it does not run every time instantly, i.e, it's unpredictable whether the GC (garbage collector) will run immediately or not.

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

1. **What are Java 8 features? Explain all of them with examples?**

Below are the some of the features of java8

**forEach():**

The “forEach” method takes the Functional Interface as a single parameter i.e. you can pass Lambda Expression as an argument.

importjava.util.ArrayList;

importjava.util.List;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

        List<String> subList = **new** ArrayList<String>();

        subList.add("Maths");

        subList.add("English");

        subList.add("French");

        subList.add("Sanskrit");

        subList.add("Abacus");

        System.out.println("------------Subject List--------------");

        subList.forEach(sub -> System.out.println(sub));

  }

}

**Optional Class:**

Java 8 introduced an optional class in the “java.util” package. “Optional” is a public final class and is used to deal with NullPointerException in the Java application. Using Optional, you can specify alternate code or values to run. By using Optional you don’t have to use too many null checks to avoid nullPointerException.

**import** java.util.Optional;

**public** **class** Main{

**public** **static** **void** main(String[] args) {

        String[] str = **new** String[10];

        Optional<String>checkNull =

                       Optional.ofNullable(str[5]);

**if** (checkNull.isPresent()) {

            String word = str[5].toLowerCase();

            System.out.print(str);

         } **else**

           System.out.println("string is null");

    }

}

**Default and Static Methods in Interface:**

In Java 8, you can add methods in the interface that are not abstract i.e. you can have interfaces with method implementation. You can use the Default and Static keyword to create interfaces with method implementation. Default methods mainly enable Lambda Expression functionality.

**import** java.util.Optional;

**interface** interface\_default {

**default** **void** default\_method(){

         System.out.println("I am default method of interface");

    }

}

**class** derived\_class **implements** interface\_default{

}

**class** Main{

**public** **static** **void** main(String[] args){

        derived\_class obj1 = **new** derived\_class();

        obj1.default\_method();

    }

}

**Method References:**

The Method reference feature introduced in Java 8 is a shorthand notation for Lambda Expressions to call a method of Functional Interface. So each time you use a Lambda Expression to refer a method, you can replace your Lambda Expression with method reference.

**import** java.util.Optional;

**interface** interface\_default {

**void** display();

}

**class** derived\_class{

**public** **void** classMethod(){

            System.out.println("Derived class Method");

    }

}

**class** Main{

**public** **static** **void** main(String[] args){

        derived\_class obj1 = **new** derived\_class();

        interface\_default  ref = obj1::classMethod;

        ref.display();

    }

}

**Java Stream API For Bulk Data Operations On Collections**

The Stream API is yet another major change introduced in Java 8. Stream API is used for processing the collection of objects and it supports a different type of iteration.

### Java Date Time API

Java 8 introduces a new date-time API under the package java.time.

**The most important classes among them are:**

* **Local:**Simplified date-time API with no complexity of timezone handling.
* **Zoned:**Specialized date-time API to deal with various timezones.

1. **How to make a pure singleton?**

The purpose of the Singleton class is to control object creation, limiting the number of objects to only one. The singleton allows only one entry point to create the new instance of the class.

The memory space wastage does not occur with the use of the singleton class because it restricts the instance creation. As the object creation will take place only once instead of creating it each time a new request is made.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. **How to make a singleton synchronized?**

To improve our Singleton pattern I have just added synchronized keyword in method declaration. In the following example only one thread can enter the getInstance() method and execute code at the time.

More specifically, the first thread is going to obtain a lock from Singleton class, execute the method, create an instance of Singleton class and return the monitor. Once the key monitor for Singleton class is free to use, the second thread is going to take lock from Singleton class, execute the method, obtain already created instance and return the key monitor.

**Submission:**  Change the document name to ***Lastname*Assignment03** where *Lastname* is your Last name and submit.