# Assignment 03 Application Design: Patterns and Frameworks 44642

Answer **all** the questions below. In your answer for each question explain with sample code or image whichever is preferable.

1. What are generics?

Generics are a programming concept that lets developers create code that can operate with different data types. It offers advantages such as flexibility, code reuse, and compile-time safety by checking types. Using generic methods in Java, for instance, enables handling any list type, minimizing repetitive code. In summary, generics are a beneficial programming tool that enhances efficiency and reduces errors.

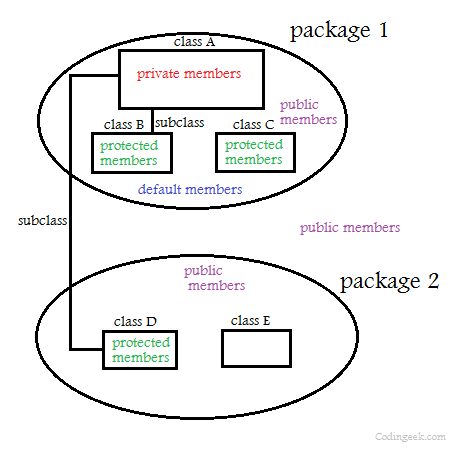
Graphical user interface, text, application, email

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?

When a method is overridden in a subclass, the visibility or access modifier of the overridden method in the superclass cannot be reduced in the subclass.

* If the superclass method is private, it cannot be overridden in the subclass.
* If the superclass method is declared as public, the subclass method can be declared as public or protected but not private or default.
* If the superclass method is declared as protected, the subclass method can be declared as public or protected but not private or default.
* If the superclass method is declared as default, the subclass method can be declared as public, protected or default but not private.



1. What is the covariant return type?

Covariant return type is a Java feature that enables an overridden method in a subclass to have a return type that is a subclass of the return type of the superclass method. This feature was introduced in Java 5, and it helps improve code readability by avoiding the need to cast return values. It allows for greater flexibility in coding and simplifies code design by permitting more specific types to be returned without casting. covariant return type is a useful feature that promotes efficient code design and enhances the functionality of Java programs.

Graphical user interface, text, application

Description automatically generated

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

In Java, we cannot override static and private methods because they belong to the class rather than a specific object. Because they are not inherited by subclasses, they cannot be overridden. Private methods are not visible to subclasses, while static methods are associated with the class and not the object of the class. As a result, it is impossible to override these methods in Java.

Graphical user interface, text, application

Description automatically generated

1. Difference between String Buffer and StringBuilder?

StringBuffer and StringBuilder are two classes in Java that are used to manipulate strings. The primary difference between these two classes is that StringBuffer is thread-safe, while StringBuilder is not. This means that StringBuffer can be used by multiple threads simultaneously, while StringBuilder can only be used by a single thread.

Additionally, StringBuffer is slower than StringBuilder due to its thread-safety features. StringBuilder is faster than StringBuffer because it lacks thread-safety features.

If thread-safety is a concern, StringBuffer is the better choice, while StringBuilder is the better choice for performance-critical applications.

Graphical user interface, text, application

Description automatically generated

1. Difference between String class and String Buffer?

Strings are immutable, meaning their values cannot be modified, while String Buffers are mutable, meaning their values can be changed. StringBuffer is more efficient when working with large or frequently changing strings, as it can change its value in the same object without creating a new one. In contrast, changing a string value creates a new string object in memory. This makes StringBuffer a better choice for applications that require frequent string manipulation.

Graphical user interface, text, application

Description automatically generated

1. Can we declare constructor as final?

In Java, constructors cannot be overridden, so there is no need to declare them as final. Declaring a constructor as final is not allowed, and it will result in a compilation error. The final keyword is used to prevent a class or method from being overridden or modified, but since constructors cannot be overridden, there is no need to use the final keyword with them. In summary, it is not possible to declare a constructor as final in Java, and it serves no purpose to do so.

Graphical user interface, text, application

Description automatically generated

1. Can we have try without catch block in java?

In Java, it is possible to have a try block without a catch block, but a finally block must be included. The finally block will execute regardless of whether an exception is thrown in the try block or not. Here is an example of a try block without a catch block:

Graphical user interface, text, application

Description automatically generated

1. What is try with the resource?

Try-with-resources is a feature in Java that allows automatic closing of resources such as files, network sockets, and database connections. The resources are automatically closed at the end of the block, eliminating the need for a finally block. The try-with-resources block requires that the resource being used implements the AutoCloseable interface. An example of try-with-resources could be opening and reading a file, where the file is automatically closed at the end of the block.

~~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?

In Java, the throws clause of a superclass method cannot be modified when we override it in the subclass. The overriding method must declare the same exception or a subclass of that exception as the superclass method. It is not allowed to declare fewer or different exceptions in the overriding method.

Graphical user interface, text, application, email

Description automatically generated

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

In UML, an association refers to a connection between two or more classes, depicting how objects of one class are related to those of another. Aggregation is a form of association in which a class acts as a container for other classes or objects, indicating a "has-a" relationship. Composition is also an association where the contained object is a fundamental part of the container object, signifying a "part-of" relationship.

Diagram

Description automatically generated

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

In Java, "final" is a keyword used to create a constant value that cannot be modified. This keyword is used primarily to create variables, methods, and classes that cannot be changed during runtime.

The "finally" block is used in exception handling to execute a block of code irrespective of whether an exception is thrown or not. This block is typically used to release resources or perform cleanup operations.

The "finalize()" method is a special method in Java that is called by the garbage collector before an object is destroyed. This method allows an object to perform final cleanup tasks before it is removed from memory. It is important to note that the "finalize()" method is not guaranteed to be called, and its execution is dependent on the garbage collector. It is also important to note that the use of the "finalize()" method is discouraged in modern Java development, as it can cause performance issues.

Diagram

Description automatically generated

1. Difference between Vector and ArrayList?

Both Vector and ArrayList are classes in Java that implement the List interface and provide dynamic arrays to store elements.

However, there are some differences between the two. While Vector is synchronized, ArrayList is not, which means Vector is thread-safe and can be used in multi-threaded applications. Vector increases its size by doubling the array size when it runs out of space, while ArrayList increases its size by 50% of the current size. This means Vector is slower when it comes to increasing its size than ArrayList.

Another difference is that Vector can be used as a stack or a queue, while ArrayList is only used as a list. This is because Vector has additional methods like "addElement()", "removeElementAt()", "elementAt()" that ArrayList does not have.

Diagram

Description automatically generated

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

There are few ways to make ArrayList methods synchronized in java.

* 1. Using Collections synchronizedList()method.
  2. Using copyOnWriteArrayList.

Graphical user interface, text, application, email

Description automatically generated

1. Difference between Hash table and Hash Map?

In Java, both HashTable and HashMap are used to store and retrieve key-value pairs. However, HashTable is a legacy class that has been around since the early days of Java, while HashMap is a newer class that was introduced in Java 1.2.

One of the main differences between the two is that HashTable is synchronized, which means it is thread-safe and can be used in multi-threaded applications without any external synchronization. On the other hand, HashMap is not synchronized by default, but it can be made thread-safe by synchronizing on a particular object.

Another difference is that HashTable does not allow null keys or values, while HashMap allows one null key and any number of null values.

Graphical user interface, text, application

Description automatically generated

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

In Java 8, HashMap internally uses an array to store key-value pairs, where each array element is a linked list of nodes. The hash code of the key determines the index of the array where the value is stored. If two keys have the same hash code, they are stored in the same linked list at that index. HashMap uses a hashing algorithm to distribute the keys across the array, making it more efficient to search for a specific key. The load factor of the HashMap determines how full the array can become before it needs to be resized, which affects the performance of the HashMap. The implementation of HashMap in Java 8 uses an optimization called tree-based buckets, which improves performance when a large number of elements are stored in the same linked list.

Text, letter

Description automatically generated

1. Difference between fail fast and fail-safe iterator?

Fail-fast and fail-safe are two types of iterators in Java that are used to iterate over collections. The main difference between them is that fail-fast iterators throw a ConcurrentModificationException if the collection is modified while iterating, while fail-safe iterators do not throw an exception and make a copy of the collection before iterating.

Fail-fast iterators are designed to quickly detect if a collection is modified while being iterated, and to prevent any further operations that might corrupt the iteration. Fail-safe iterators, on the other hand, are designed to create a copy of the collection at the beginning of the iteration and iterate over the copy, so that any changes to the original collection do not affect the iteration.

Ex:

ArrayList<Integer> numbers = // ...

Iterator<Integer> iterator = numbers.iterator();

while (iterator.hasNext()) {

Integer number = iterator.next();

numbers.add(50);

}

Iterators on Collections from java.util.concurrent package such as ConcurrentHashMap, CopyOnWriteArrayList, etc. are Fail-Safe in nature.

Ex:

ConcurrentHashMap<String, Integer> map = new ConcurrentHashMap<>(); map.put("First", 10);

map.put("Second", 20);

map.put("Third", 30);

map.put("Fourth", 40);

Iterator<String> iterator = map.keySet().iterator();

while (iterator.hasNext()) {

String key = iterator.next();

map.put("Fifth", 50);

}

1. Can we start the thread twice?

No, it is not possible to start a thread twice in Java. If you attempt to start a thread that has already been started, it will throw an "IllegalThreadStateException". Once a thread has completed its execution, you can create a new thread with the same name and start it, but you cannot start the same thread again.

Graphical user interface, text, application

Description automatically generated

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

When creating a thread in Java, there are two ways to do it: by implementing the Runnable interface or by extending the Thread class.

The Runnable interface is preferred because it provides better code reusability and separation of concerns. If a class extends the Thread class, it cannot extend any other class because Java does not support multiple inheritance.

On the other hand, if a class implements the Runnable interface, it can still extend another class and implement the Runnable interface to create a thread. Using the Runnable interface also makes it easier to use thread pools and execute tasks concurrently.

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?

In Java, a thread can undergo various states during its lifecycle, which include:

New: This is the state of a thread when it is first created.

Runnable: This is the state when a thread is ready to run but is waiting for a processor to execute it.

Running: This is the state of a thread when it is being executed.

Blocked: This state indicates that a thread is waiting for a monitor lock to be released. Waiting: In this state, a thread is waiting for another thread to perform a specific action. Timed Waiting: This state indicates that a thread is waiting for a specified period of time. Terminated: This is the state of a thread when it has finished its execution or has been terminated.

Diagram

Description automatically generated

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

In Java, serialization is the process of converting an object into a sequence of bytes that can be stored or transmitted over a network. The opposite process of turning a sequence of bytes back into an object is called deserialization. To achieve serialization, the class of the object being serialized must implement the Serializable interface. This interface does not have any methods, but serves as a marker to the Java Virtual Machine (JVM) that the class can be serialized. During the serialization process, the object's instance variables are written to an output stream in a specific format. This format includes information about the object's class, its superclass, and the values of its instance variables.

Graphical user interface, text, application

Description automatically generated

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

In Java, an immutable class is one that cannot be modified after creation and has final instance variables and no mutator methods. The String class is immutable in Java, meaning its value cannot be changed once created. Modifying a String creates a new String object. This makes String objects cacheable and thread-safe.

Graphical user interface, text, application

Description automatically generated

1. Do immutable classes thread safe? If yes then how?

Immutable classes are considered thread-safe because they cannot be modified after their creation. Because the state of immutable objects cannot change, multiple threads can read the same object without causing issues with synchronization. Therefore, immutable objects are safe to use in concurrent environments.

Text

Description automatically generated

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

We can call the garbage collector to run explicitly in Java using the System.gc() method. But there is no assurance that the garbage collector will execute right away when this method is called. The JVM decides when to run the garbage collector, and it may choose to postpone or disregard the request.

A screenshot of a computer

Description automatically generated with medium confidence

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

Java 8 provides following features for Java Programming:

* + Lambda expressions,
  + Method references,
  + Functional interfaces,
  + Stream API,
  + Optional class,
  + Collectors class,
  + ForEach() method,
  + Nashorn JavaScript Engine,
  + Parallel Array Sorting,
  + Default methods,
  + Base64 Encode Decode,
  + Static methods in interface,
  + Type and Repating Annotations,
  + IO Enhancements,
  + Concurrency Enhancements,
  + JDBC Enhancements etc.

1. How to make a pure singleton?

The process of creating a pure singleton involves ensuring that only one instance of a class is created throughout the lifetime of an application. This can be achieved by making the class constructor private to prevent external instantiation and providing a public static method to access the single instance of the class. This method creates the instance only if it does not already exist, otherwise it returns the existing instance. In addition, the class should be made final to prevent any subclass from creating another instance. This approach ensures that the singleton object is thread-safe and can be accessed from multiple parts of the code without creating multiple instances.

Text

Description automatically generated

1. How to make a singleton synchronized?

To make a singleton synchronized, the getInstance() method should be synchronized. This ensures that only one thread at a time can create an instance of the singleton class, preventing multiple instances from being created. However, this approach may impact performance since threads have to wait for each other to access the method. An alternative approach is to use double-checked locking, where synchronization is done only if the instance is null. This improves performance by avoiding unnecessary synchronization.

Graphical user interface, text, application, email

Description automatically generated

gitHub Link :

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