**Assignment Questions 5**

💡 **Q1.What is Exception in Java?**

In Java, an exception is an event that disrupts the normal flow of a program during its execution. It represents an unexpected condition or error and allows the program to handle and recover from such situations gracefully.

💡 **Q2.What is Exception Handling?**

Exception handling is a mechanism in Java to manage runtime errors and abnormal conditions that may occur during program execution. It involves the use of try-catch blocks to catch and handle exceptions, preventing the program from terminating abruptly. Developers can gracefully handle exceptions, provide error messages, log information, or take corrective actions to maintain program stability and reliability.

💡 **Q3.What is the difference between Checked and Unchecked Exceptions and Error?**

Checked Exceptions:

1. Checked exceptions are checked at compile-time.

2. Methods that may throw checked exceptions must declare them in their method signature using "throws" keyword.

3. Forces developers to handle or propagate exceptions, ensuring proper error handling.

4. Examples: IOException, SQLException.

Unchecked Exceptions:

1. Unchecked exceptions are not checked at compile-time.

2. Subclasses of RuntimeException, they don't require explicit handling or declaration.

3. Typically caused by logical errors in code.

4. Examples: NullPointerException, ArrayIndexOutOfBoundsException.

Error:

1. Errors are unrecoverable conditions like OutOfMemoryError.

2. Not meant to be caught or handled by applications.

3. Usually caused by JVM or environment issues.

4. Examples: StackOverflowError, OutOfMemoryError.

💡 **Q4.What are the difference between throw and throws in Java?**

In Java, `throw` is used inside a method to explicitly throw an exception, while `throws` is used in a method's signature to declare checked exceptions that the method may throw. `throw` is used for actual exception throwing, whereas `throws` is used for exception declaration.

💡 **Q5.What is multithreading in Java? mention its advantages**

Multithreading in Java refers to the concurrent execution of multiple threads within a single Java program. Each thread represents a separate flow of control, allowing tasks to be executed simultaneously.

Advantages of multithreading in Java:

1. Improved performance by utilizing multiple CPU cores efficiently.

2. Enhanced responsiveness and better user experience in GUI applications.

3. Efficient handling of concurrent tasks and resource sharing.

4. Better utilization of system resources.

5. Simplified code organization for parallel tasks.

💡 **Q6.Write a program to create and call a custom exception**

class CustomException extends Exception {

    public CustomException(String message) {

        super(message);

    }

}

public class CustomExceptionExample {

    public static void main(String[] args) {

        try {

            throw new CustomException("Custom Exception this one");

        } catch (CustomException e) {

            System.out.println("Caught Exception as" + e.getMessage());

        }

    }

}

💡 **Q7.How can you handle exceptions in Java?**

1. try-catch block: Surrounds code that may throw an exception and provides a catch block to handle the exception.

2. Multiple catch blocks: Allows handling different types of exceptions separately.

3. finally block: Executes regardless of whether an exception occurs or not.

4. throws keyword: Used in method signature to propagate checked exceptions to the caller.

💡 **Q8.What is Thread in Java?**

In Java, a thread is the smallest unit of execution within a program. It represents an independent flow of control that allows concurrent execution of tasks, enabling a program to perform multiple operations simultaneously and efficiently utilize system resources.

💡 **Q9. What are the two ways of implementing thread in Java?**

1. Extending the `Thread` class and overriding `run()` method.

2. Implementing the `Runnable` interface and providing the `run()` method implementation.

💡 **Q10.What do you mean by garbage collection?**

Garbage collection in Java is an automatic memory management process. It identifies and reclaims unused objects (garbage) in the memory to free up space. The JVM periodically runs the garbage collector, which traces and deletes unreferenced objects, ensuring efficient memory utilization and preventing memory leaks.