Task2:

**Q1**

1)public: This modifier allows a class, method, or variable to be accessible from anywhere, both within the same package and from other packages. For classes, it means the class can be instantiated and used from any part of the program. For methods and variables, it indicates that they can be accessed from any other class or package.

2)private: This modifier restricts the access to within the same class. Private members (methods or variables) are not visible or accessible outside the class they are declared in. This is often used to encapsulate internal details of a class and prevent direct manipulation from other classes.

3)protected: Members with the protected modifier can be accessed within the same package as well as by subclasses (even if the subclasses are in different packages). This allows for controlled inheritance and access to certain elements of a class.

4)Default : If no access modifier is specified, the default access level is applied. Members with default access are accessible only within the same package. They are not accessible to classes in other packages.

1)public: Accessible from anywhere.  
2)private: Accessible only within the same class.  
3)protected: Accessible within the same package and by subclasses.  
4)Default (no modifier): Accessible only within the same package.

Q2:

**Exception:**

1)An exception is an abnormal condition or situation that arises during the execution of a program.  
2)Exceptions are typically caused by factors that can be handled within the program, such as incorrect user input, file not found, or network issues.  
3)Exceptions are represented by classes derived from the Exception class or its subclasses. These are usually recoverable situations.  
4)Examples of exceptions include NullPointerException, ArrayIndexOutOfBoundsException, and FileNotFoundException.

**Error;**

1)An error, on the other hand, is a severe problem or issue that generally occurs due to factors outside the control of the program.  
2)Errors often indicate issues that cannot be easily recovered from, such as running out of memory (OutOfMemoryError), hardware or system failures, or issues with the Java Virtual Machine (JVM) itself.  
3)Errors are represented by classes derived from the Error class or its subclasses. These situations are typically unrecoverable and are not meant to be caught or handled by the application.  
4)Examples of errors include StackOverflowError, OutOfMemoryError, and InternalError.

Q3:

**Checked Exceptions:**

1)Checked exceptions are exceptions that are checked at compile-time by the Java compiler.  
2)These exceptions are subclasses of the Exception class (but not RuntimeException) and usually indicate conditions that a well-behaved program should anticipate and handle.  
3)When a method can potentially throw a checked exception, the compiler forces the programmer to either catch and handle the exception using a try-catch block or declare that the method may throw the exception using the throws keyword.  
4)Examples of checked exceptions include IOException, SQLException, and ClassNotFoundException.

**Unchecked Exceptions (Runtime Exceptions):**

1)Unchecked exceptions, also known as runtime exceptions, are exceptions that are not checked at compile-time. They are subclasses of the RuntimeException class.  
2)These exceptions usually indicate programming errors or unexpected conditions that the program might not be able to handle. They often result from improper use of methods or data structures.  
3)Unlike checked exceptions, the compiler does not enforce handling or declaring unchecked exceptions using try-catch or throws.  
4)Examples of unchecked exceptions include NullPointerException, ArrayIndexOutOfBoundsException, and IllegalArgumentException.