Task 11

1.Access Modifiers available in Java and What is the significance in terms of class,method, and Variable accessibility ?

Public :

Classes: Public classes can be accessed from any other class in any package.

Methods: Public methods can be accessed from any class in any package.

Variables: Public variables can be accessed from any class in any package.

Protected :

Classes: While classes cannot be declared as protected, all classes in the same package can access protected members of other classes in the same package.

Methods: Protected methods can be accessed by classes in the same package and by subclasses (even if they are in different packages).

Variables: Protected variables have similar accessibility as protected methods.

Default (Package-Private or No Modifier):

Classes: If no access modifier is specified, the class is given "default" access, which means it can only be accessed by classes in the same package.

Methods: Methods without an access modifier (package-private) can be accessed by classes in the same package.

Variables: Variables without an access modifier (package-private) have similar accessibility as package-private methods.

Private :

Classes: Classes cannot be declared as private.

Methods: Private methods can only be accessed within the same class.

Variables: Private variables can only be accessed within the same class.

Access modifiers play a crucial role in encapsulation, which is a fundamental principle in object-oriented programming. Encapsulation helps control the visibility and access to internal details of a class, allowing developers to create well-defined and maintainable code by limiting the interaction between different parts of the program. By choosing appropriate access modifiers, you can ensure that your classes, methods, and variables are only accessible where they are intended to be used, reducing the risk of unintended side effects and enhancing code security.

2.Difference between Exception and Error?

|  |  |  |
| --- | --- | --- |
| Aspect | Checked Exception | Unchecked Exception |
| Handling Requirement | Must be either caught or declare with throws keyword . | No explicit handling or declaration required. |
| Inheritance | Subclass of Exception | Subclass of runtime exception |
| Compile time checked | Compilers enforces handling or declaration. | Compilers does not enforce handling. |
| Examples | IOException,SQL  Exception | NullPointerException,  ArrayIndexOutOf  BoundsException |
| Typical Use Case | Recoverable scenarios,like I/O erros. | Programming errors,like null references. |
| Try-Catch Block | Often used with try-Catch blocks. | Can be used with try-catch , but not required. |
| Code Clarity | Explicity indicates potential issues. | Might result in cleaner but riskier code. |
| Forced Handling | Enforced by compiler. | Not enforced by compiler. |
| RunTimeException | Does not include RuntimeException. | Includes RuntimeException and Subclasses. |

3.Difference between checked Exception and unchecked Exception?

|  |  |
| --- | --- |
| CHECKED EXCEPTION | UNCHECKED EXCEPTION |
| Checked Exceptions are the exceptions that are checked and handled at compile time. | Unchecked Exceptions are the exceptions that are not checked at compiled time. |
| The program gives a compilation error if a method throws a checked exception. | The program compiles fine because the compiler is not able to check the exception. |
| If some code within a method throws a checked exception, then the method must either handle the exception or it must specify the exception using throws keyword. | A method is not forced by compiler to declare the unchecked exceptions thrown by its implementation. Generally, such methods almost always do not declare them, as well. |
| A checked exceptions occur when the chances of failure are too high. | Unchecked exception occurs mostly due to programming mistakes. |
| They are direct subclass of Exception class but do not inherit from RuntimeException. | They are direct subclass of RuntimeException class. |
| For errors that the programmer cannot directly prevent from occurring. | For errors that the programmer can directly prevent from occurring. |
| These Exceptions are occurred at compiler time. | These exceptions are occurred at run time . |

4.Coding :

A screenshot of a computer

Description automatically generated

5.Coding:

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

6.Coding :

A screenshot of a computer program

Description automatically generated

7.Coding :

A screen shot of a computer

Description automatically generated

8.Coding:

A screenshot of a computer program

Description automatically generated