Experiment No. 7

Aim: Implementation of Exception Handling in Java.

Problem Statement:

1. Write a Program to demonstrate Exception Handling. Handle minimum two exceptions.
2. Write a Program to create and handle own user-level exception subclass.
3. Develop application which can handle any combination of predefined compile time and runtime exceptions using multiple catch blocks.

Theory:

An exception (or exceptional event) is a problem that arises during the execution of a program. When an Exception occur the normal flow of the program is disrupted and the program/Application terminates abnormally, which is not recommended, therefore, these exceptions are to be handled.
An exception can occur for many different reasons. Following are some scenarios where an exception occurs.

- A user has entered an invalid data.
- A file that needs to be opened cannot be found.
- A network connection has been lost in the middle of communications or the JVM has run out of memory.

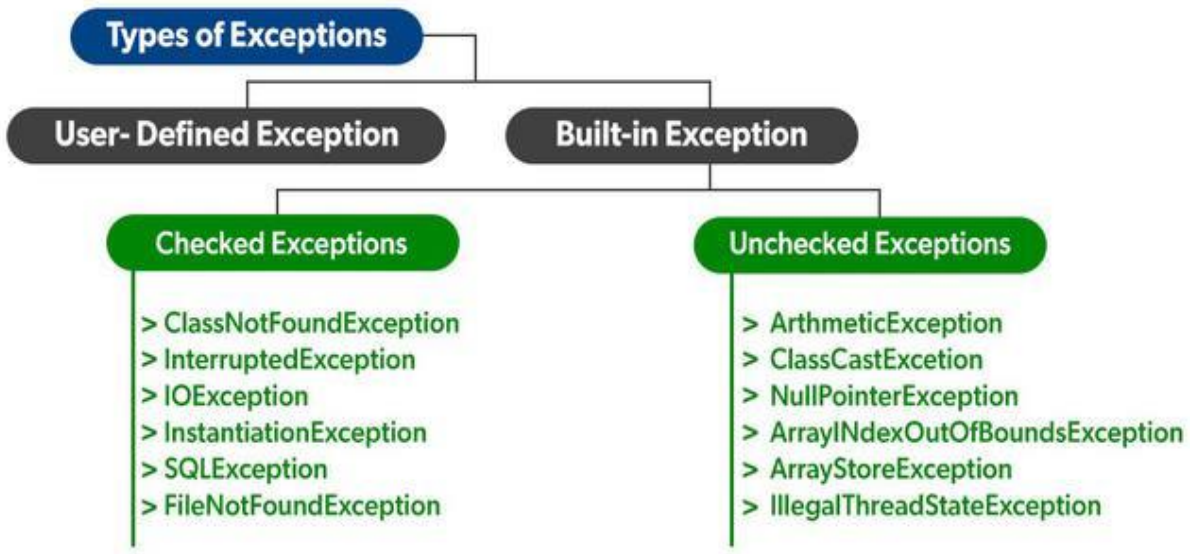
Exception Handler

- The code that is responsible for doing something about the exception is called an exception handler.
- It is a mechanism to handle the runtime errors so that the normal flow of the application can be maintained.
- Exception handler is block of code that is executed automatically when an error occurs.

Exception Class Hierarchy:

Types of Exception:

Some of exceptions are caused by user error, some by programmer error, and some by physical resources that have failed in some manner. Based on these, we have following categories of Exceptions.



1. **Checked Exceptions:**
   - Exceptions checked at compile-time by the compiler are called as Checked exceptions.
   - Checked exceptions are compile-time exceptions.

2. **Unchecked Exceptions:**
   - Exceptions not checked at compile-time, but they are checked at runtime are called as Unchecked exceptions.
   - Unchecked exceptions are run-time exceptions.
   - If a program throws an unchecked exception, and even if we didn't handle it, the program would not give a compilation error.

Exception Handling Mechanism

In java, exception handling is done using five keywords-

1. try
2. catch
3. throw
4. throws
5. finally

try keyword:

- The "try" keyword is used to specify a block where we should place an exception code.
- The try block must be followed by either catch or finally.

catch keyword:

- If an exception occurs in guarded code, the catch block that follows the try is checked.
- The "catch" block is used to handle the exception.

Syntax of try-catch block

```
try{
        //code that may throw an exception
}catch(Exception_class_Name ref)
{
        //exception description
}
```

throw keyword:

- The "throw" keyword is used to throw an exception.
- It is used to manually throw an exception.
- The throw keyword is used to throw exceptions to the runtime to handle it.

throws keyword:

- The "throws" keyword is used to declare exceptions.
- If a method generates an exception that it does not handle, it must declare in a throws clause.

finally keyword:

- The "finally" block is used to execute the necessary code of the program.
- Finally block is always executed whether an exception is handled/thrown or not.

Creating Own Exception Subclasses

- Exceptions created by user is called as user-defined exception or costumexception.
- Create Exception types
    - -To handle situations specific to your application.
    - -To manage errors that relate specifically to your application.
- Define a subclass of Exception (subclass of Throwable)
- Exception defines public constructors.
    1. Exception( )
    2. Exception(String *msg)*
- toString( ) defined by Throwable (inherited by Exception)

Conclusion:

Student should able to handle and create the exceptions.