Exception Handling:

Exception handling in java is a mechanism that allows developers to deal with runtime errors or exceptional conditions that may occur during the execution of a program.

It provides a greatway to gracefully handle these errors and take appropriate action to prevent the program from crashing or behaving unpredictabily

In java exceptions are represented by objects that are derived from the 'Throwable' class.

There are two main types of exceptions:

- → Checked
- → Unchecked

Checked Exception: These are need to declared in the method signature or handled using a 'try-catch' block.

Un-checked: (known as run-time exceptions) do not require explicit handling.

Exception handling in Java follows a structured approach using three main keyword:

- Try
- catch
- finally.

→ The 'try' block contains the code that may throw exception.

→ The 'catch' block contains handling the exception by specifying the type of exception it can catch and the actions to be taken when the exception occurs.

→ The 'finally' block, which is optional, is used to specify the code that should be executed regardless of whether an exception occurred or not.

File I/O: when reading or writing file exceptions may occur if the file is not found, permissions are inadequate, or the file is corrupted. Exception handling allows developers to handle these cases gracefully and provide appropriate feedback to the user.

classes - Getters and Setters :

In java, Getter and setter methods are used to retrive and update the private variables. when you hide the implementation of the object of the class from the outer world, you declare them as private.

As private members of the class are not accessible from outside the class, so we can use the getter & setter method to retrieve & update the values of private members.

```
public class Car {
    private String doors;
    private String engine;
    private String drivers;
    private int speed;
    public void setSpeed (int speed){
        this.speed =speed;
    }
    public int getSpeed (){
        return speed;
    }
}
```

To access the properties (variables) and functionalities (methods) of car in another class, we have to create the object of car class.

```
public class Hello {
    public static void main(String[]args){
        Car car = new Car();
        car.setSpeed(100);
        S.O.P(car.getSpeed());
    }
}
```

This:

'This' is the keyword in java that can be used to invoke the constructor methods, static members etc.. or the current instance of a class.

Super:

super is the reserved keyword in java that is used to invoke constructors and methods of the parent class. This is possible only when one class inherits another class. (child class extends parent class).

Array List:

Array list in java supports dynamic arrays & it is based on array data structure. ArrayList are resizable which means they can grow or shrink dynamically as we add & remove elements.

The Array List class is a part of collection framework, extends ~~Abstract class~~ Abstract List class & implements the List interface. This class is declared in java.util. package. It is used for creating dynamic arrays that are resizable in nature.

Syntax:

How to create array list in JAVA?

example:

```
ArrayList <Type> arr = new ArrayList <Type> ();
```

```
import java.util.*;
class CreateArrayList {
    public static void main (String [] args){
        int n=5;
        ArrayList <Integer> arr = new ArrayList
                                  <Integer> (n);
        S.O.P("Array list is created");
    }
}
```

We can add elements in array list with the help of add() method.

Syntax:

add(object o) : this method is used to directly add at end.

23/06/23