SNS COLLEGE OF TECHNOLOGY

Kurumbapalayam (Po), Coimbatore – 641 107

AN AUTONOMOUS INSTITUTION

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY

23CSB201-OBJECT ORIENTED PROGRAMMING

UNIT IV - EXCEPTION AND MULTITHREADING

TOPIC 3 - try catch and finally block, throws

1. try block

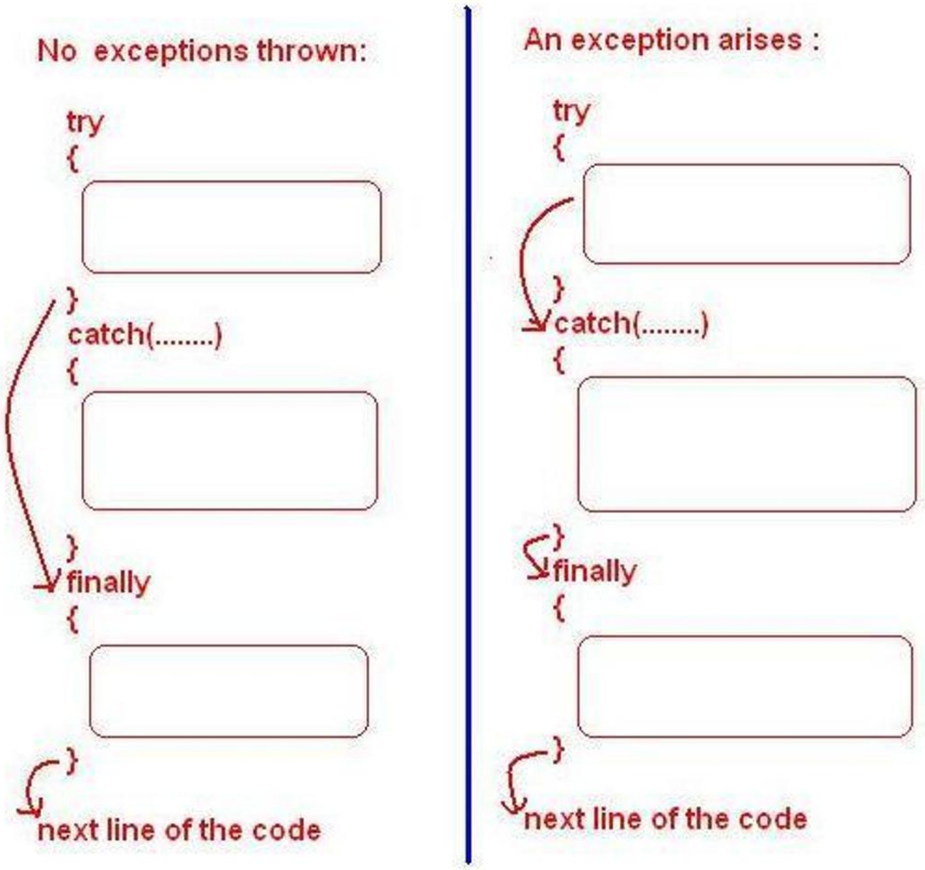
- The try **block** is used to write code that **may cause an exception**.
- It must be followed by at least one catch or a finally block.

Syntax:

```
try {
    // code that may cause exception
}
```

Example:

```
try {
    int a = 10 / 0; // ArithmeticException
}
```



2. catch block

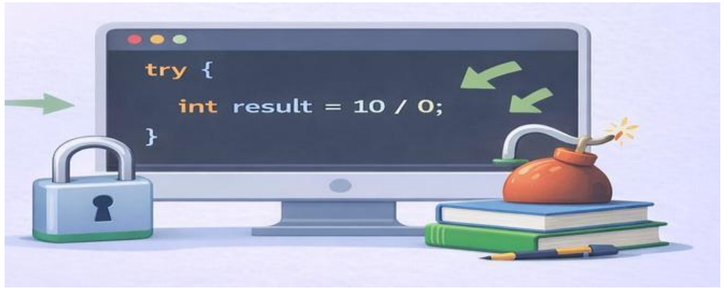
- The catch **block** handles the exception thrown in the try block.
- You can have **multiple catch blocks** for different exceptions.

Syntax:

```
try {
    // risky code
} catch (ExceptionType e) {
    // handling code
}
```

Example:

```
try {
    int a = 10 / 0;
} catch (ArithmeticException e) {
    System.out.println("Cannot divide by zero");
}
```

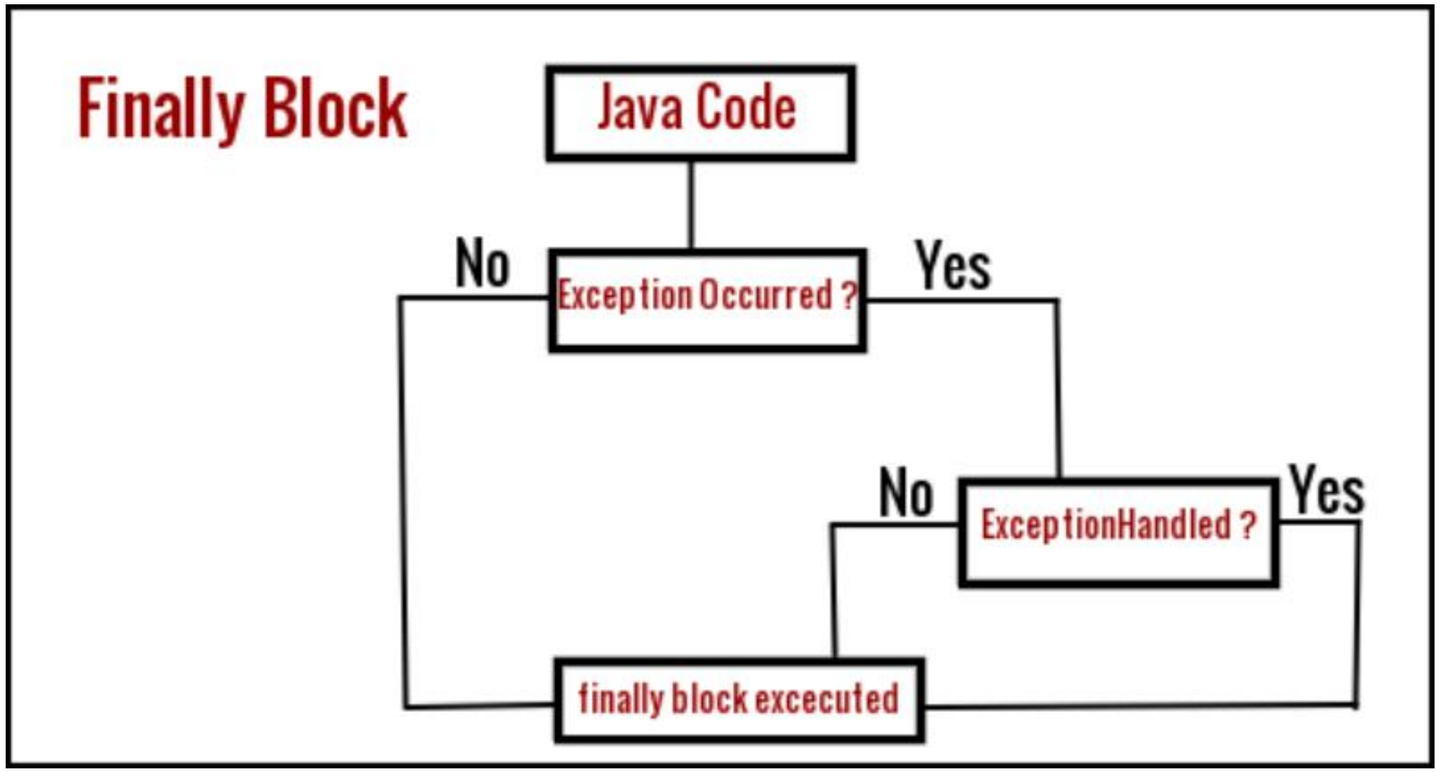


3. finally block

- The finally **block** is **always executed**, whether an exception occurs or not.

Used for:

- Closing files
- Releasing resources
- Cleanup operations



Syntax:

```
try {
    // risky code
} catch (ExceptionType e) {
    // handle exception
} finally {
    // always executes
}
```

Example:

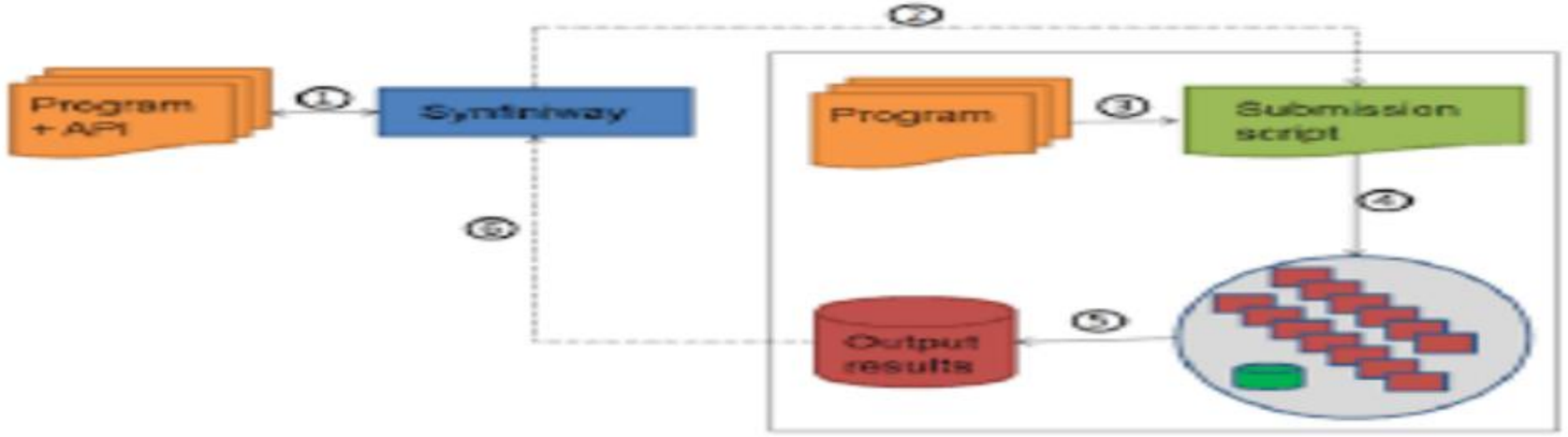
```
try {
    int a = 10 / 2;
} catch (ArithmeticException e) {
    System.out.println("Error");
} finally {
    System.out.println("This will always execute");
}
```

Output:

This will always execute

Flow of Execution

Situation	What Happens
No exception	try → finally
Exception handled	try → catch → finally
Exception not handled	try → finally → program stops

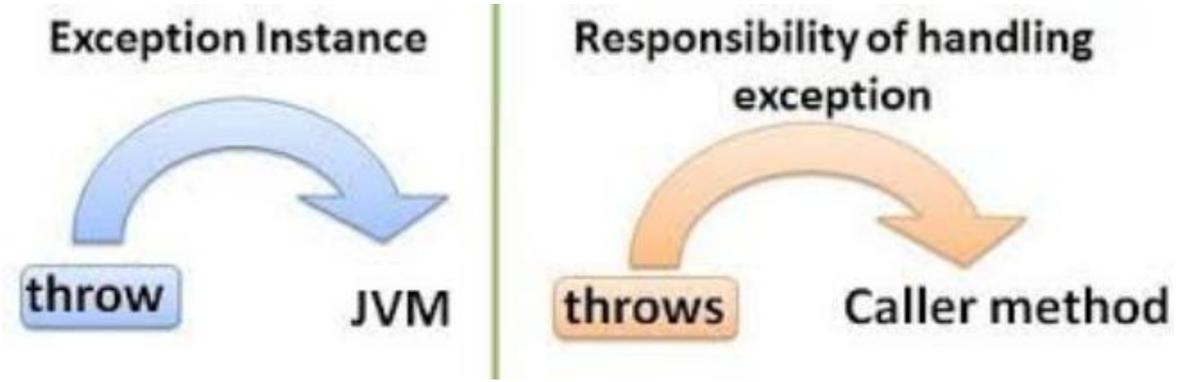


4. throws keyword

- The throws **keyword** is used to **declare exceptions** that a method may throw.
- It does NOT handle the exception
- It passes the exception to the caller

Syntax:

```
returnType methodName() throws ExceptionType {
    // code
}
```



Example:

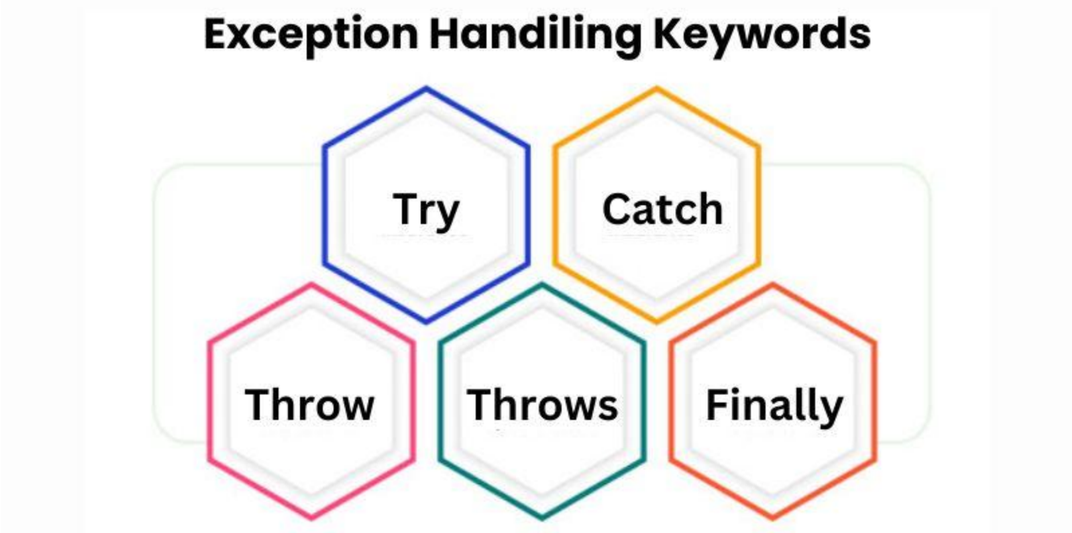
```
class Test {
    static void check() throws ArithmeticException {
        int a = 10 / 0;
    }
    public static void main(String[] args) {
        check(); // exception will occur here
    }
}
```

Here:

- check() declares throws ArithmeticException
- It does NOT handle it
- Exception is passed to main()

Handling throws with try-catch

```
class Test {
    static void check() throws ArithmeticException {
        int a = 10 / 0;
    }
    public static void main(String[] args) {
        try {
            check();
        } catch (ArithmeticException e) {
            System.out.println("Handled in main");
        }
    }
}
```



Difference: throw vs throws

Feature	throw	throws
Purpose	Used to explicitly throw exception	Declares exception
Used in	Method body	Method declaration
Keyword type	Statement	Keyword
Example	throw new Exception()	throws Exception

Example using throw

```
class Test {
    static void validate(int age) {
        if (age < 18) {
            throw new ArithmeticException("Not eligible");
        } else {
            System.out.println("Eligible");
        }
    }
    public static void main(String[] args) {
        validate(15);
    }
}
```

Important Points

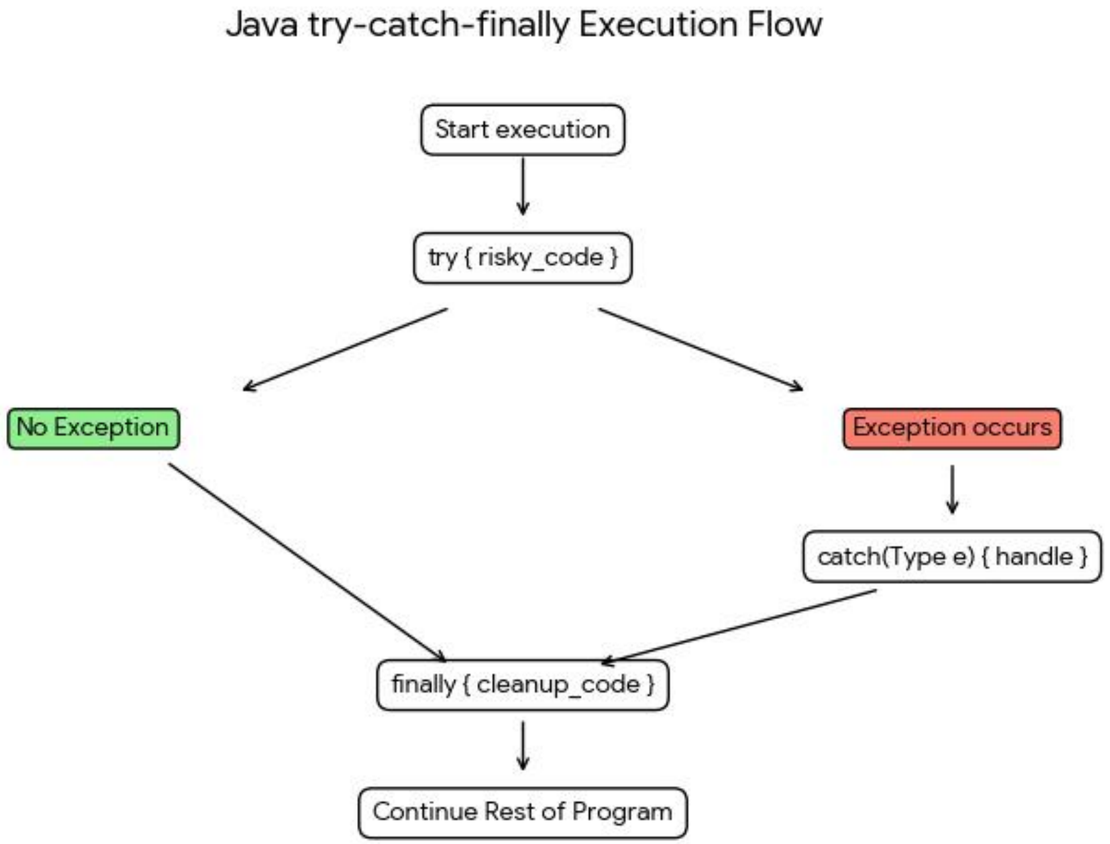
- ✓ try must be followed by catch or finally
- ✓ finally always executes (except system crash)
- ✓ Multiple catch blocks are allowed
- ✓ throws is used to delegate exception handling
- ✓ throw is used to create custom exceptions

Real-Time Example

```
import java.io.*;
class FileExample {
    public static void main(String[]
args) {
        FileReader file = null;
        try {
            file = new FileReader("tes
t.txt");
            int data = file.read();
            System.out.println(data);
        } catch (IOException e) {
            System.out.println("File not found");
        } finally {
            try {
                if (file != null) {
                    file.close();
                }
            } catch (IOException e) {
                System.out.println("Error closing file");
            }
        }
    }
}
```

Summary

- **try** → contains risky code
- **catch** → handles exception
- **finally** → always executes
- **throws** → declares exception



Assessment

1. Which keyword is used to explicitly throw a single exception from within a method?

a) try

b) throws

c) throw

d) catch

2. Which of the following statements about the finally block is true?

a) It only executes if an exception is caught.

b) It executes regardless of whether an exception occurs, unless System.exit(0) is called.

c) It is mandatory to include it after every try-catch block.

d) It is used to declare exceptions that a method might throw.

Assessment

3. What is the purpose of the throws keyword in a method signature?

a) To handle a specific exception immediately.

b) To catch any runtime exceptions.

c) To declare that a method may throw one or more specific exceptions to its caller.

d) To define a new custom exception class.

4. When using multiple catch blocks, what is the required order for the exception types?

a) General exceptions (like Exception) first, then specific exceptions.

b) Specific exceptions first, then general exceptions.

c) Any order is acceptable.

d) Checked exceptions first, then unchecked exceptions.

THANK YOU