**1. Phân biệt throw và throws**

**throw:**

Dùng để ném ra một đối tượng exception cụ thể trong mã.

Thường nằm bên trong phương thức hoặc khối lệnh.

Ví dụ:

public void validateAge(int age) {

if (age < 18) {

throw new IllegalArgumentException("Tuổi phải lớn hơn 18");

}

}

**throws:**

Khai báo những exception mà phương thức có thể ném ra.

Được dùng trong chữ ký của phương thức.

Ví dụ:

public void readFile() throws IOException {

// Mã đọc file có thể ném IOException

}

**2. Checked và Unchecked Exception**

**Checked Exception:**

Được kiểm tra trong quá trình biên dịch (compile-time).

Lập trình viên bắt buộc phải xử lý chúng bằng try-catch hoặc khai báo với throws.

Ví dụ: IOException, SQLException.

public void readFile() throws IOException {

// Checked exception cần được khai báo hoặc xử lý

FileReader file = new FileReader("file.txt");

}

**Unchecked Exception:**

Không bị kiểm tra trong quá trình biên dịch (runtime).

Thường là lỗi logic mà lập trình viên có thể tránh được nếu cẩn thận.

Kế thừa từ lớp RuntimeException.

Ví dụ: NullPointerException, ArrayIndexOutOfBoundsException.

public void divide(int a, int b) {

// Unchecked exception xảy ra nếu b = 0

int result = a / b;

}

**3. Phân biệt try-catch và try-with-resources**

**try-catch:**

Dùng để bắt và xử lý ngoại lệ trong mã.

Không tự động đóng tài nguyên (nếu sử dụng file, kết nối mạng…).

Ví dụ:

try {

FileReader file = new FileReader("file.txt");

// Đọc file

} catch (IOException e) {

System.out.println("Lỗi: " + e.getMessage());

}

**try-with-resources:**

Tự động đóng tài nguyên (nếu tài nguyên triển khai interface AutoCloseable).

Giảm thiểu việc phải gọi close() thủ công.

Ví dụ:

java

Sao chép mã

try (FileReader file = new FileReader("file.txt")) {

// Đọc file

} catch (IOException e) {

System.out.println("Lỗi: " + e.getMessage());

}

**4. Tạo Custom Exception**

Custom Exception: Là một loại ngoại lệ do người dùng tự định nghĩa bằng cách kế thừa từ Exception hoặc RuntimeException.

Bước 1: Tạo một class Custom Exception.

// Kế thừa từ Exception (checked exception)

public class InvalidAgeException extends Exception {

public InvalidAgeException(String message) {

super(message);

}

}

Bước 2: Sử dụng Custom Exception.

public void validateAge(int age) throws InvalidAgeException {

if (age < 18) {

throw new InvalidAgeException("Tuổi phải lớn hơn 18");

} else {

System.out.println("Tuổi hợp lệ");

}

}

public static void main(String[] args) {

try {

new Demo().validateAge(15);

} catch (InvalidAgeException e) {

System.out.println("Lỗi: " + e.getMessage());

}

}