**Task 5: Xử lý ngoại lệ và Luồng vào/ra**

1. **Xử lý ngoại lệ**

- Ngoại lệ: sự kiện xảy ra trong quá trình thực thi chương trình, và có thể làm gián đoạn luồng xử lý thông thường của chương tình

=> Cần có cơ chế xử lý các ngoại lệ.

- Lớp **Throwable** là lớp cơ sở cho các đối tượng có thể được ném (throw) và bắt (catch) bằng câu lệnh try-catch. Hai lớp con trực tiếp là **Exception** và **Error**

- Lớp Exception: đại diện cho các ngoại lệ. Exception là những sự cố có thể dự đoán và xử lý.

- Lớp Error: đại diện cho lỗi nghiêm trọng mà chương trình không nên bắt. Error là những vấn đề mà chương trình không thể giải quyết, thường liên quan đến môi trường của JVM

1. **try-catch-finally**

- try: Khối chứa đoạn mã có thể gặp lỗi hoặc ném ra ngoại lệ khi thực thi. Nếu có ngoại lệ xảy ra, chương trình sẽ chuyển quyền kiểm soát đến khối catch

- catch: Khối chứa đoạn mã được thực thi nếu có ngoại lệ xảy ra trong khối try tương ứng

- finally: Khối này chứa đoạn mã sẽ được thực thi bất kể có ngoại lệ xảy ra hay không. Khối này có thể có hoặc không trong quá trình xử lý ngoại lệ

1. **Các loại Exception**
2. **Exception**

- RuntimeException (unchecked exception): được phát hiện trong runtime, là ngoại lệ có thể tránh được

* ArithmeticException
* NullPointerException
* NumberFormatException
* ClassCastException

- Ngoại lệ khác (checked exception): được phát hiện trong quá trình biên dịch

* IOException
* ParseException
* ClassNotFoundException
* SQLException

1. **Error**

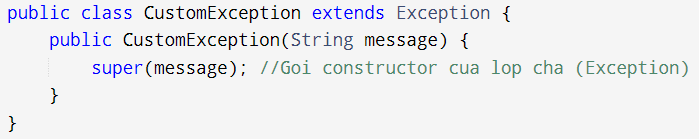
- Không nên cố gắng để bắt nó và xử lý. Error là tình trạng bất thường mà ứng dụng không nên gặp phải

- Một số loại:

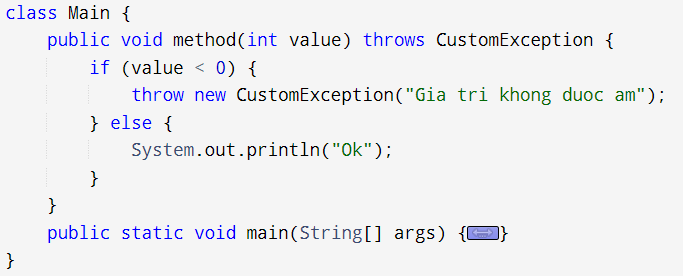
* OutOfMemoryError
* StackOverflowError
* UnknownError

1. **Ngoại lệ tùy chỉnh**

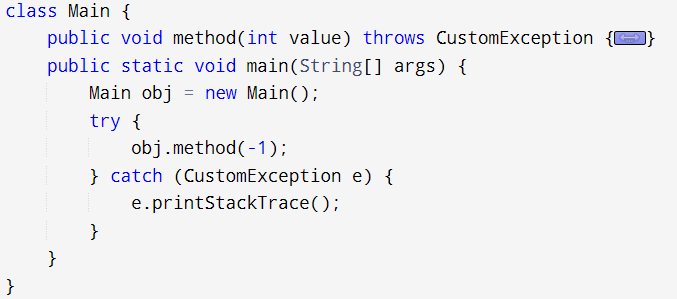
- Kế thừa từ lớp Exception đối với **checked exception** hoặc lớp RuntimeException với **unchecked exception**.



- Khai báo method với throws và ném ra ngoại lệ với từ khóa throw



- Sử dụng:



**\* throws và throw**

- throws: Được dùng để khai báo rằng phương thức có thể ném ra ngoại lệ tương ứng. Trách nhiệm xử lý ngoại lệ thuộc phương thức gọi

- throw: Được dùng để tạo ra một ngoại lệ cụ thể và chuyển quyền kiểm soát từ phương thức hiện tại cho phương thức gọi

1. **Luồng vào/ra (I/O)**
2. **Làm việc với tập tin**
3. **FileReader**

- Lớp FileReader thuộc gói java.io và được kế thừa từ lớp InputStreamReader.

- Được dùng để đọc dữ liệu là các ký tự từ tập tin. Thường là tệp tin văn bản

- Khởi tạo một đối tượng của lớp FileReader:

*FileReader reader = new FileReader(String filePath);*

hoặc:

*FileReader reader = new FileReader(File fileObj);*

- Các phương thức khác:

|  |  |
| --- | --- |
| **Phương thức** | **Sử dụng** |
| read() | Đọc 1 ký tự từ reader |
| read(char[] arr) | Đọc các ký tự từ reader và lưu trữ trong arr |
| read(char[] arr, int start, int n) | Đọc **n** ký tự từ reader và lưu trong arr bắt đầu tại vị trí start của mảng arr |
| ready() | Trả về true nếu FileReader sẵn sàng để đọc |
| skip(long n) | Bỏ qua n ký tự, trả về số ký tự đã bỏ qua thực tế |
| close() | Đóng FileReader, giải phóng tài nguyên |

- Sử dụng **try-with-resources** đảm bảo tài nguyên được đóng tự động, cú pháp: *try ( fileReaderObj ) { //code } catch ...*

1. **FileWriter**

- Lớp FileWriter thuộc gói java.io và được kế thừa từ lớp OutputStreamWriter.

- Được dùng để ghi dữ liệu là các ký tự vào tập tin. Thường là tệp tin văn bản

- Khởi tạo một đối tượng của lớp FileWriter:

*FileWriter writer = new FileWriter(String filePath, boolean append);*

hoặc:

*FileWriter writer = new FileWriter(File fileObj, boolean append);*

- Lưu ý:

* File được tạo mới nếu chưa có
* Nếu **append = false** hoặc **không có tham số append** thì dữ liệu hiện có sẽ bị ghi đè. Ngược lại, dữ liệu được ghi vào cuối file

- Các phương thức khác:

|  |  |
| --- | --- |
| **Phương thức** | **Sử dụng** |
| write(int c) | Ghi 1 ký tự (có mã Unicode là c) vào tệp |
| write(char[] arr) | Ghi mảng ký tự vào tệp |
| write(char[] arr, int start, int n) | Ghi **n** phần tử trong mảng **arr** vào tệp bắt đầu từ chỉ mục **start** của mảng arr |
| write(String str) | Ghi một chuỗi ký tự vào tệp |
| write(String str, int start, int n) | Ghi một phần gồm **n** ký tự của **str** vào tệp bắt đầu từ chỉ mục **start** của str |
| close() | Đóng FileReader, giải phóng tài nguyên |
| flush() | Ghi mọi dữ liệu còn lại trong bộ đệm ra tệp |

- Sử dụng **try-with-resources** đảm bảo tài nguyên được đóng tự động ngay cả khi có ngoại lệ xảy ra

1. **Đọc và ghi tệp văn bản**
2. **Đọc tệp văn bản**

- Sử dụng các lớp như **FileReader**, **BufferedReader** để đọc tệp văn bản

- Sử dụng cú pháp **try-with-resources** đảm bảo đóng tệp tự động

**\*FileReader**: đọc tệp dưới dạng ký tự

**\*BufferedReader**:

- Khởi tạo: *BufferedReader br = new BufferedReader(fileReaderObj);*

- Sử dụng bộ đệm để tăng hiệu suất khi đọc tệp

- Cung cấp phương thức readLine() để đọc từng dòng

1. **Ghi tệp văn bản**

- Sử dụng các lớp **FileWriter**, **BufferedWriter** để ghi tệp văn bản

\***FileWriter**: ghi ký tự vào tệp

\***BufferedWriter**:

- Khởi tạo: *BufferedWriter bw = new BufferedWriter(fileWriterObj);*

- Sử dụng bộ đệm để tối ưu việc ghi tệp

- Cung cấp phương thức newLine() để ghi ký tự xuống dòng