Bài 15

|  |  |
| --- | --- |
| 1. Phân biệt Errors và Exceptions ? | * Errors:   + Lỗi nghiêm trọng liên quan đến hệ thống + Không thể hoặc không nên xử lý - VD: OutOfMemoryError, StackOverflowError   * Exception:   + Lỗi logic chương trình + Có thể bỏ qua hoặc xử lý bằng try- catch + NullPointerException, IOException |
| 1. Exception là gì? Các loại Exception ? | * Exception: Lỗi xảy ra khi chạy chương trình, do logic sai, dữ liệu không hợp lệ,… * Các loại Exception:   + Checked Exception (bắt buộc xử lý): Ví dụ IOException, SQLException  + Unchecked Exception (runtime, không bắt buộc xử lý): Ví dụ NullPointerException, ArithmeticException |
| 1. Các cách xử lý Exception? Khi nào dùng? | * Cách xử lý   + try-catch: Bắt lỗi và xử lý trực tiếp.  + throws: Đẩy lỗi ra cho nơi gọi xử lý.   * Khi nào dùng   + try-catch: Khi muốn xử lý lỗi ngay tại chỗ (ghi log, thông báo, retry…)  + throws: Khi không xử lý tại chỗ mà để phương thức gọi xử lý. |
| 1. Lưu ý khi dùng try-catch ? | + Bắt đúng loại Exception cần xử lý (đừng bắt chung chung như Exception)  + Không để khối catch rỗng.  + Không lạm dụng: chỉ bọc code dễ phát sinh lỗi.  + Có thể dùng nhiều catch cho các loại lỗi khác nhau. |
| 1. try-with-resources dùng để làm gì? Cách dung ? | * Dùng để: Tự động đóng tài nguyên (file, socket, stream) mà không cần gọi close(). * Cách dùng   try (BufferedReader br = new BufferedReader(new FileReader("file.txt"))) {  // đọc file  } catch (IOException e) {  e.printStackTrace();  }     * JVM tự đóng br sau khi try kết thúc. |
| 1. Khi nào sử dụng finally? | * Khi luôn cần thực hiện một hành động dù có lỗi hay không (ví dụ: đóng kết nối, file, giải phóng tài nguyên…). |