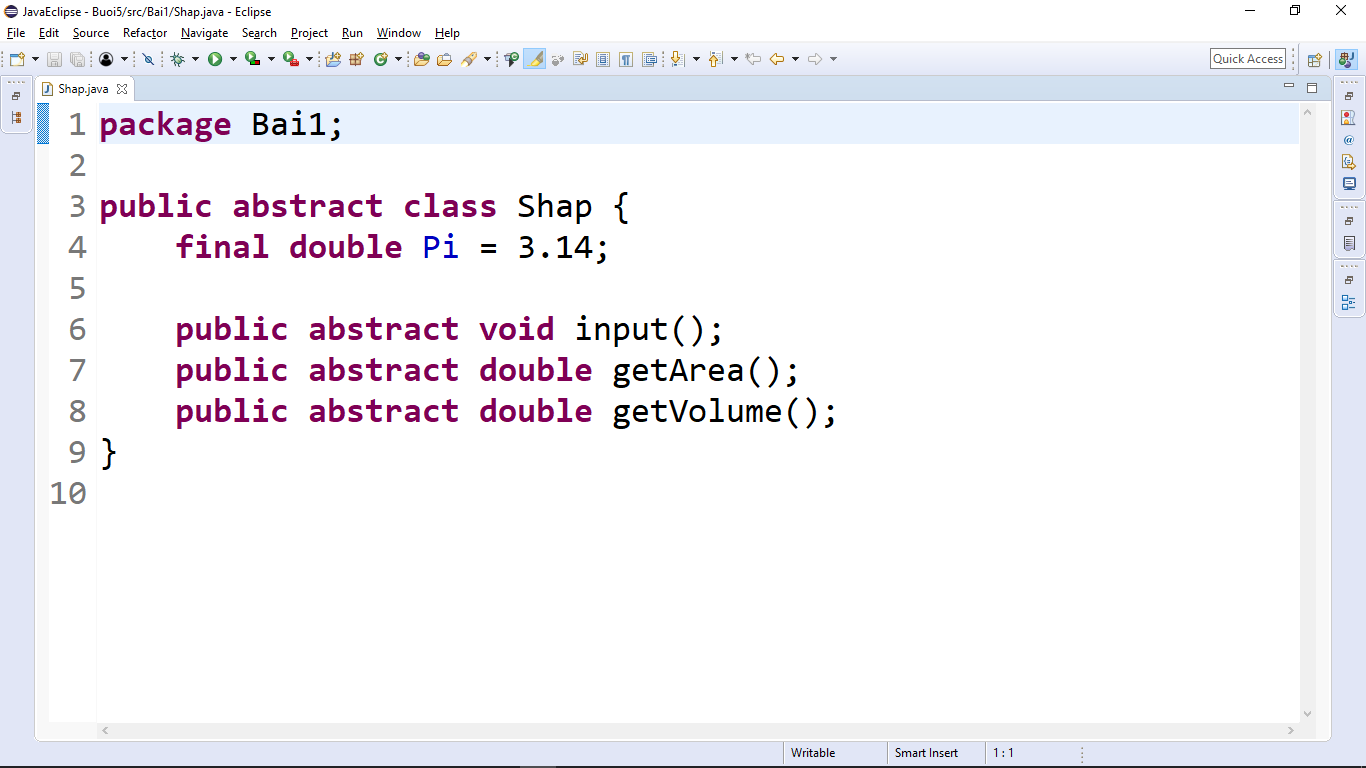
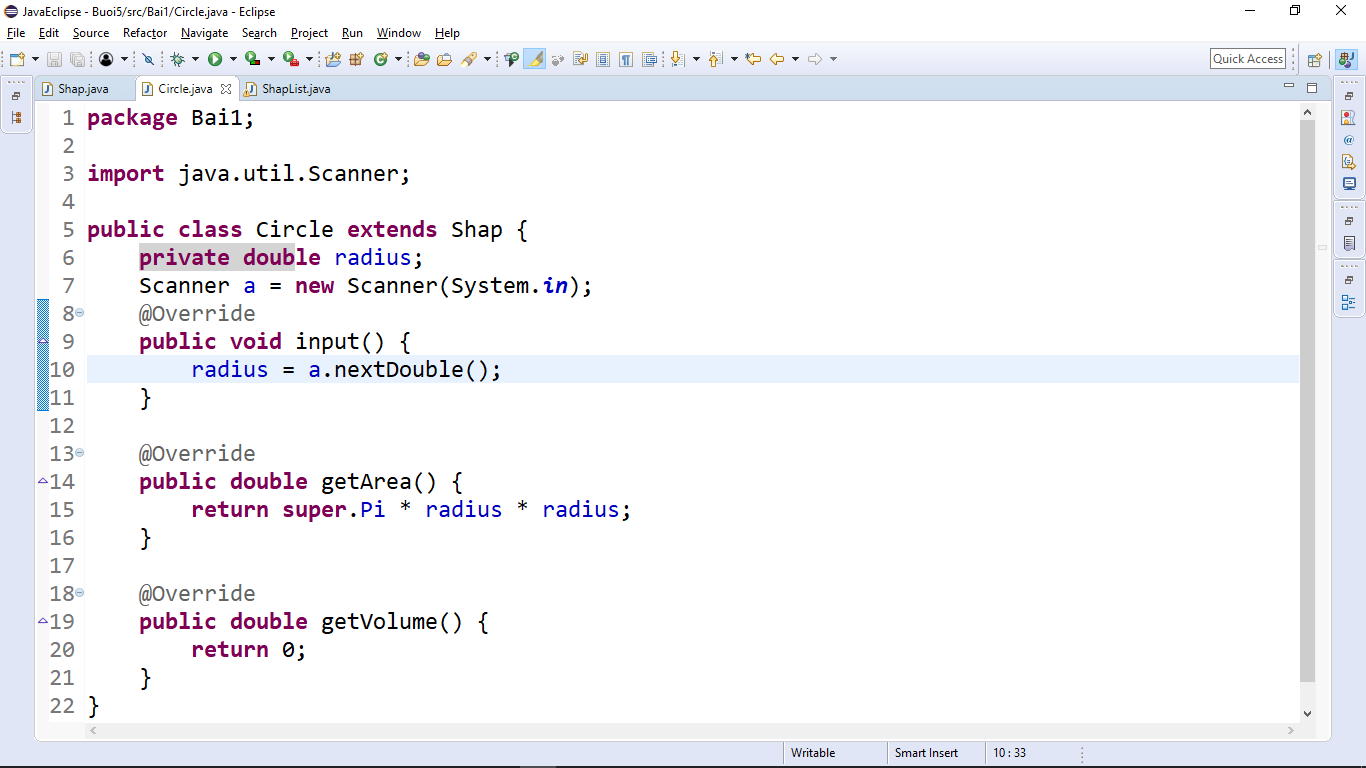
**LỘ TRÌNH ĐÀO TẠO PHẦN 8, 9**

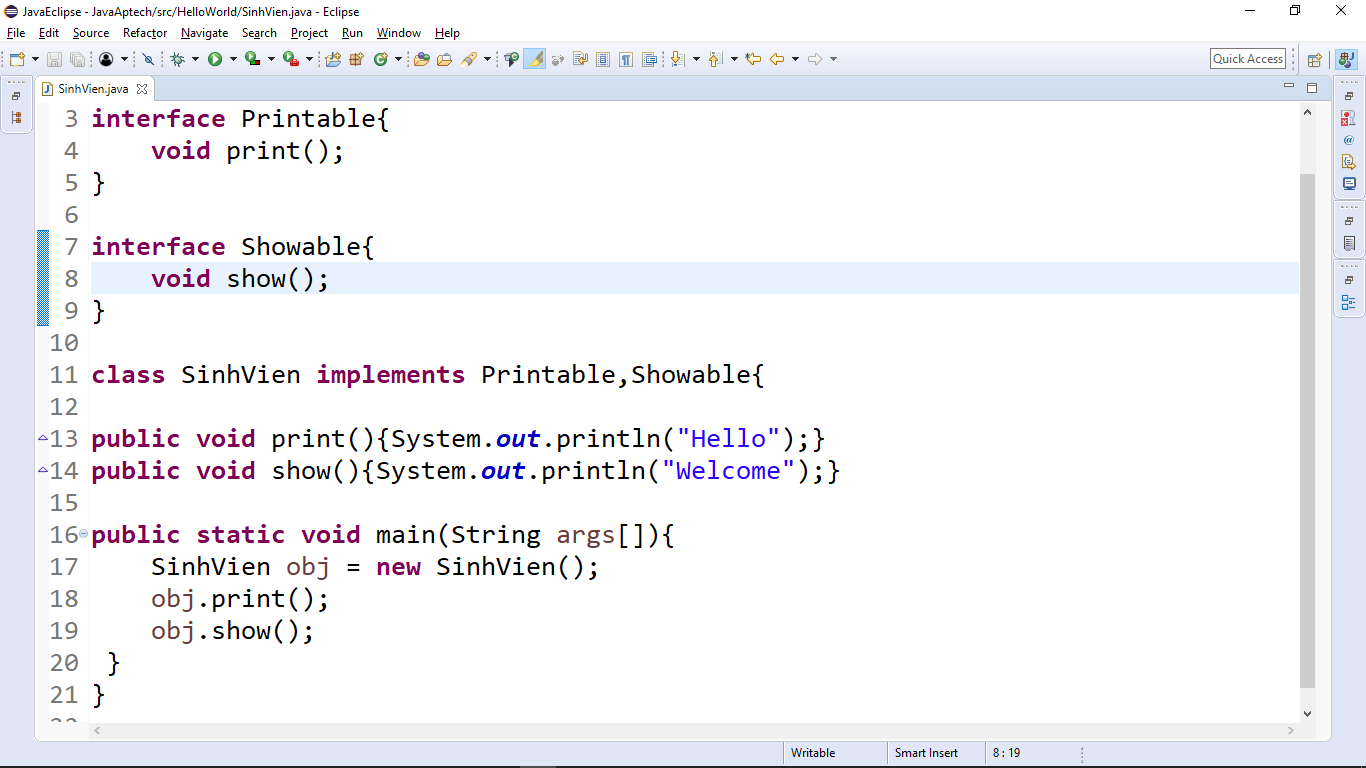
1. **Kế thừa trong java**
   1. **Giới thiệu về tính kế thừa**
      1. **Khái niệm về tính kế thừa**
         * Tính kế thừa trong Java là môt kỹ thuật mà trong đó một đối tượng thu được tất cả thuộc tính và hành vi của đối tượng cha. Ý tưởng đằng sau tính kế thừa trong Java là bạn có thể tạo các lớp mới mà được xây dựng dựa trên các lớp đang tồn tại. Khi bạn kế thừa từ một lớp đang tồn tại, bạn có thể tái sử dụng các phương thức và các trường của lớp cha, và bạn cũng có thể bổ sung thêm các phương thức và các trường khác.
      2. **Tại sao sử dụng tính kế thừa trong java**
         * Dùng để ghi đề các phương thức (Method overriding), nhằm thu được tính đa hình.
         * Để làm tăng tính tái sử dụng code.
   2. **Các kỹ năng cơ bản với tính kế thừa**
      1. **Định nghĩa lớp cha**
         * Lớp chứa các đặt tính tổng quát được kế thừa lại nhằm sử dụng các đặc tính đã có sẵn được gọi là lớp cha.
         * Từ khóa abstract: Một lớp được khai báo là abstract thì đó là lớp trừu tượng. Nó cần được kế thừa và phương thức của nó được triển khai. Nó không thể được khởi tạo.
         * Các phương thức trong lớp trừu tượng là các phương thức trừu tượng (Phương thức không có phần thân).



* + 1. **Định nghĩa lớp con**
       - Lớp mới kế thừa lại các lớp sẵn có trước đó nhằm tái sử dụng code đã có (tái sử dụng tài nguyên) được gọi là lớp con.
       - Từ khóa extends: Dùng để lớp con kế thừa từ lớp cha. Khi kế thừa phải ghi đè (override) các phương thức có ở lớp cha dù có sử dụng hay không sử dụng phương thức đó.



* + 1. **Định nghĩa về tính đa kế thừa**
       - Khi đã hiểu tính kế thừa và từ khóa extends thì nó chỉ được dùng để kế thừa từ 1 lớp cha duy nhất, vậy để kế thừa và tận dụng tính đa kế thừa trong java thì phỉa làm thế nào ?
* Ta có interface và từ khóa implements.
  + - * Khi triển khai interface, có và quy tắc sau:
* Một lớp có thể triển khai một hoặc nhiều interface tại một thời điểm.
* Một lớp chỉ có thể kế thừa một lớp khác, nhưng được triển khai nhiều interface.
* Một interface có thể kế thừa từ một interface khác, tương tự cách một lớp có thể kế thừa lớp khác.



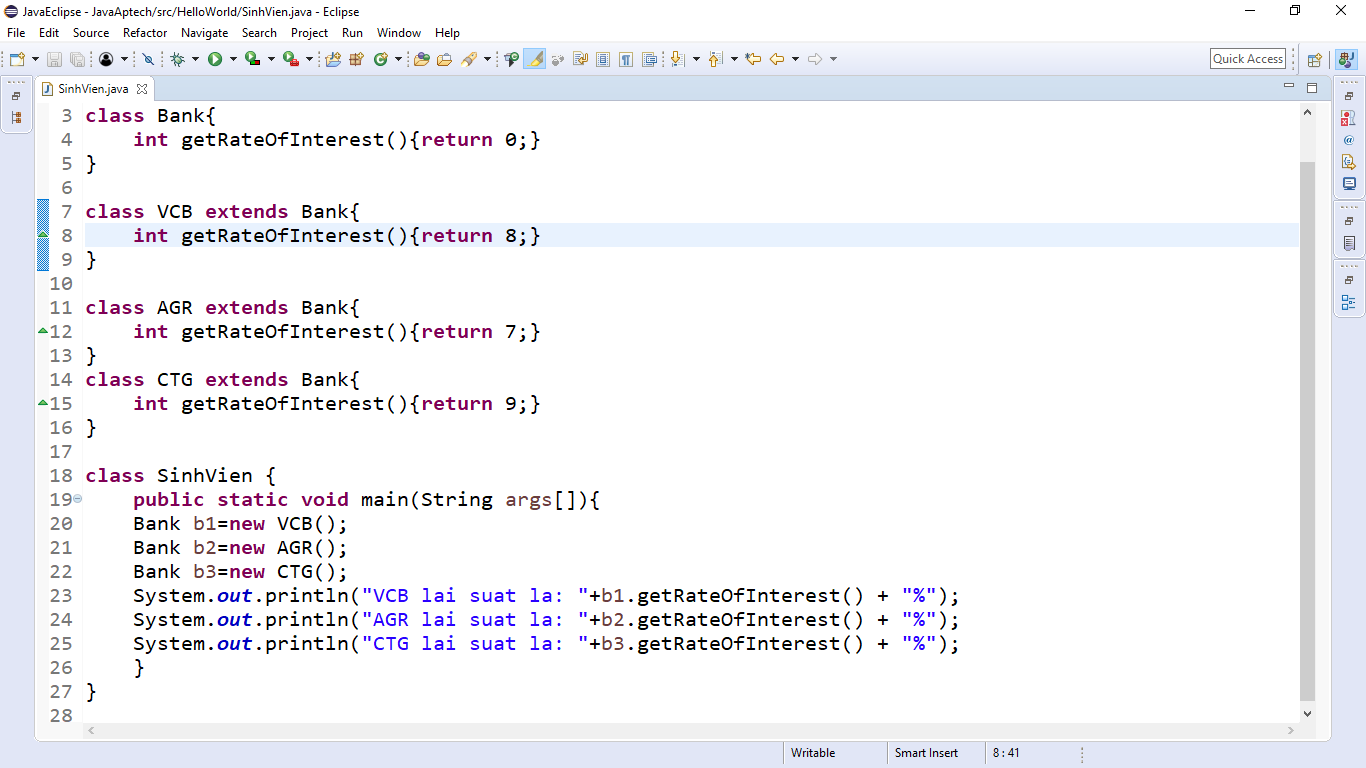
* + 1. Phân biệt abstract và interface trong java

|  |  |  |
| --- | --- | --- |
| STT | **Lớp trừu tượng** | **Interface** |
| 1 | Lớp trừu tượng có thể có các phương thức abstract và non-abstract | Interface chỉ có thể có phương thức abstract |
| 2 | Lớp trừu tượng không hỗ trợ đa kế thừa | Interface hỗ trợ **đa kế thừa** |
| 3 | Lớp trừu tượng có thể có các biến **final, non-final, static và non-static** | Interface chỉ có các biến **static và final** |
| 4 | Lớp trừu tượng có thể có phương thức static, phương thức main và constructor | Interface không thể có phương thức static, main hoặc constructor. |
| 5 | Từ khóa abstract được sử dụng để khai báo lớp trừu tượng | Từ khóa interface được sử dụng để khai báo Interface |
| 6 | Lớp trừu tượng có thể cung cấp trình triển khai của Interface | Interface không cung cấp trình triển khai cụ thể của lớp abstract |
| 7 | Ví dụ: public abstract class Shape{ public abstract void draw(); } | Ví dụ: public interface Drawable{ void draw(); } |

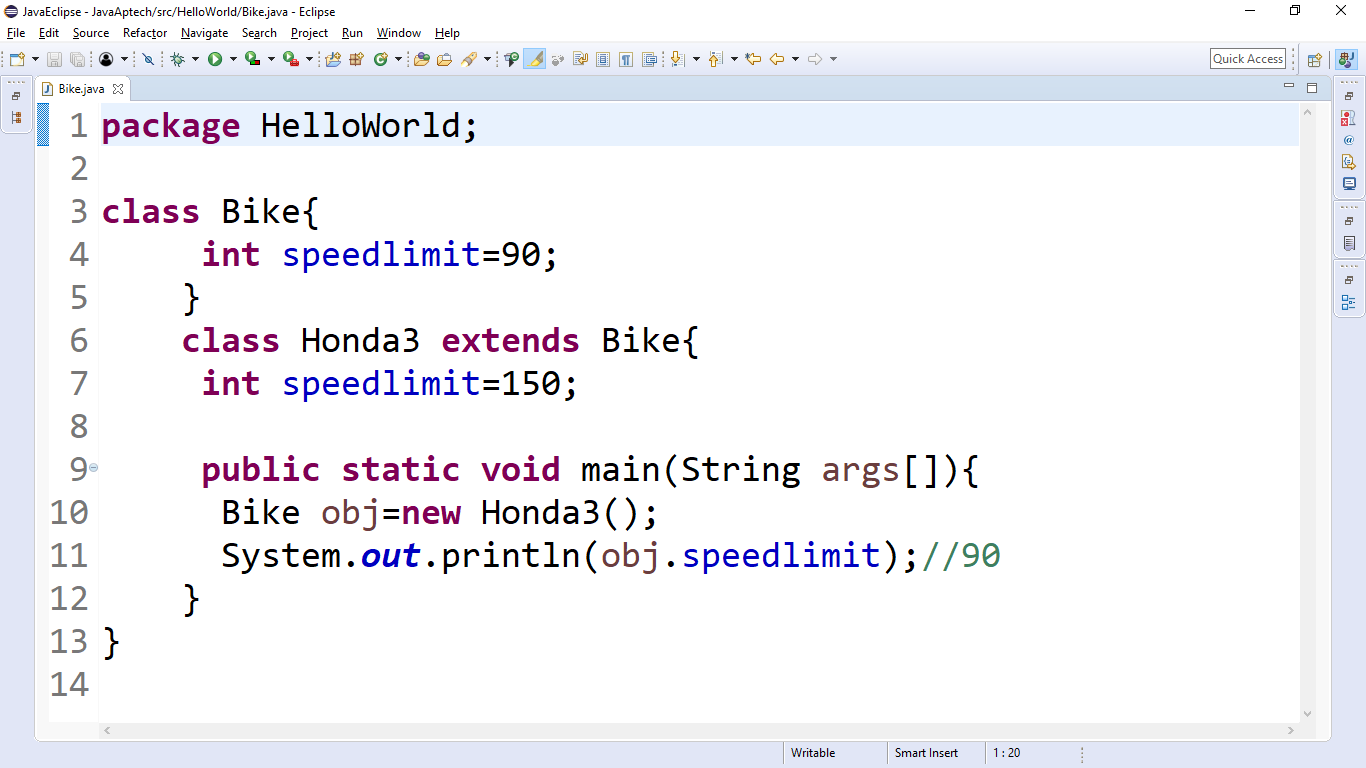
1. **Tính đa hình**
   1. **Định nghĩa**
      * + Tính đa hình trong Java là một khái niệm mà từ đó chúng ta có thể thực hiện một hành động đơn theo nhiều cách khác nhau.

**VD:** Trong trận chiến chỉ huy phải nói: Quân thủy đánh, quân bộ đánh,.. để toàn quân tham gia trận chiến. Thay vì vậy chỉ huy chỉ cần nói “Toàn quân đánh” thì ngay lập tức tất cả các binh sĩ đều hiểu và cùng thức hiện 1 hành động là đánh..

* 1. **Các thao tác cơ bản**
     + - Giả sử Bank là một lớp cung cấp phương thức để lấy lãi suất. Nhưng lãi suất lại khác nhau giữa từng ngân hàng. Ví dụ, các ngân hàng VCB, AGR và CTG có thể cung cấp các lãi suất lần lượt là 8%, 7% và 9%.



* + - * Không thể ghi đề lên các thành viên dữ liệu.



1. **Các trường hợp ngoại lệ trong java**
   1. **Exception ?**
      * + Là một tình trạng bất thường.

**VD:**

* Người dùng nhập dữ liệu không hợp lệ.
* Một file cần được mở nhưng không thể tìm thấy.
* Kết nối mạng bị ngắt trong quá trình thực hiện giao tiếp hoặc JVM hết bộ nhớ.
  1. **Phân biệt Exception với Error**
* **Error:** Nó không giống các exception, nhưng vấn đề xảy ra vượt quá tầm kiểm soát của lập trình viên hay người dùng. Error được bỏ qua trong code của bạn vì bạn hiếm khi có thể làm gì đó khi chương trình bị error. Ví dụ như OutOfMemoryError, VirtualMachineError, AssertionError, … Nó được bỏ qua trong quá trình Java biên dịch.
* **Checked Exception:** Là ngoại lệ thường xảy ra do người dùng mà không thể lường trước được bởi lập trình viên.
* **Unchecked Exception:** Một ngoại lệ xảy ra ở runtime là ngoại lệ có thể tránh được bởi lập trình viên.