Primordial data types

1. [**boolean**: Kiểu dữ liệu boolean có hai giá trị có thể là true hoặc false1](https://codefly.vn/cac-kieu-du-lieu-java-nguyen-thuy/12895).
2. [**byte**: Kiểu dữ liệu byte có thể có giá trị từ -128 đến 127 (số nguyên có dấu 8 bit)1](https://codefly.vn/cac-kieu-du-lieu-java-nguyen-thuy/12895).
3. [**short**: Kiểu dữ liệu short có thể có giá trị từ -32768 đến 32767 (số nguyên có dấu 16bit)1](https://codefly.vn/cac-kieu-du-lieu-java-nguyen-thuy/12895).
4. [**int**: Kiểu dữ liệu int có thể có giá trị từ -2^31 đến 2^31-1 (số nguyên có dấu 32 bit)1](https://codefly.vn/cac-kieu-du-lieu-java-nguyen-thuy/12895).
5. [**long**: Kiểu dữ liệu long có thể có giá trị từ -2^63 đến 2^63-1 (số nguyên có dấu 64 bit)1](https://codefly.vn/cac-kieu-du-lieu-java-nguyen-thuy/12895).
6. [**float**: Kiểu dữ liệu float là số thực dấu chấm động đơn (32 bit)](https://codefly.vn/cac-kieu-du-lieu-java-nguyen-thuy/12895)[1](https://codefly.vn/cac-kieu-du-lieu-java-nguyen-thuy/12895).
7. [**double**: Kiểu dữ liệu double là số thực dấu chấm động kép (64 bit)](https://codefly.vn/cac-kieu-du-lieu-java-nguyen-thuy/12895)[1](https://codefly.vn/cac-kieu-du-lieu-java-nguyen-thuy/12895).
8. [**char**: Kiểu dữ liệu char được sử dụng để lưu trữ một ký tự (16 bit Unicode)](https://codefly.vn/cac-kieu-du-lieu-java-nguyen-thuy/12895)[1](https://codefly.vn/cac-kieu-du-lieu-java-nguyen-thuy/12895).[2](https://stackjava.com/java/cac-loai-du-lieu-nguyen-thuy-trong-java-primitives-variable.html).

Non-primordial data types

* [**Class**: Đây là một kiểu dữ liệu do người dùng định nghĩa, được sử dụng để tạo ra các đối tượng](https://www.youtube.com/watch?v=ZzKsLg1Uqng)[1](https://www.youtube.com/watch?v=ZzKsLg1Uqng).
* [**String**: Đây là một kiểu dữ liệu đặc biệt trong Java, được sử dụng để lưu trữ chuỗi ký tự1](https://www.youtube.com/watch?v=ZzKsLg1Uqng).
* [**Array**: Đây là một kiểu dữ liệu được sử dụng để lưu trữ một tập hợp các giá trị có cùng kiểu1](https://www.youtube.com/watch?v=ZzKsLg1Uqng).
* [**Interface**: Đây là một kiểu dữ liệu đặc biệt trong Java, được sử dụng để định nghĩa các phương thức mà một lớp phải triển khai1](https://www.youtube.com/watch?v=ZzKsLg1Uqng).
* [Các kiểu dữ liệu do chúng ta tự tạo ra1](https://www.youtube.com/watch?v=ZzKsLg1Uqng).

1. [**abstract**: Từ khóa abstract trong Java được sử dụng để khai báo lớp trừu tượng](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
2. [**boolean**: Từ khóa boolean trong Java được sử dụng để khai báo một biến là kiểu dữ liệu boolean](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
3. [**break**: Từ khóa break trong Java được sử dụng để ngắt vòng lặp hoặc chuyển đổi câu lệnh](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
4. [**byte**: Từ khóa byte trong Java được sử dụng để khai báo một biến có thể chứa các giá trị dữ liệu 8 bit](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
5. [**case**: Từ khóa case trong Java được sử dụng với các câu lệnh switch để kiểm tra một danh sách các trường hợp](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
6. [**catch**: Từ khóa catch trong Java được sử dụng để bắt các ngoại lệ được tạo bởi các câu lệnh try](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
7. [**char**: Từ khóa char trong Java được sử dụng để khai báo một biến có thể chứa các ký tự Unicode 16 bit không dấu](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
8. [**class**: Từ khóa class trong Java được sử dụng để khai báo một lớp](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
9. [**continue**: Từ khóa continue trong Java được sử dụng để tiếp tục vòng lặp](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
10. [**default**: Từ khóa default trong Java được sử dụng để chỉ định khối mã mặc định trong câu lệnh switch](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
11. [**do**: Từ khóa do trong Java được sử dụng trong câu lệnh điều khiển để khai báo một vòng lặp](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
12. [**double**: Từ khóa double trong Java được sử dụng để khai báo một biến có thể chứa các số dấu phẩy động 64 bit](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
13. [**else**: Từ khóa else trong Java được sử dụng để chỉ ra các nhánh thay thế trong câu lệnh if](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
14. [**enum**: Từ khóa enum trong Java được sử dụng để xác định một bộ hằng cố định](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
15. [**extends**: Từ khóa extends trong Java được sử dụng để chỉ ra rằng một lớp có nguồn gốc từ một lớp hoặc interface khác1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
16. [**final**: Từ khóa final trong Java được sử dụng để chỉ ra rằng một biến giữ giá trị không đổi](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).
17. [**finally**: Từ khóa finally trong Java chỉ ra một khối mã trong cấu trúc try…catch](https://laptrinhvienjava.com/tu-khoa-trong-java/)[1](https://laptrinhvienjava.com/tu-khoa-trong-java/).

[Trong lập trình hướng đối tượng, **lớp (Class)** và **đối tượng (Object)** là hai khái niệm cơ bản](https://viettuts.vn/java/lop-va-doi-tuong-trong-java)[1](https://viettuts.vn/java/lop-va-doi-tuong-trong-java)[2](https://funix.edu.vn/chia-se-kien-thuc/phan-bie%CC%A3t-lop-va-doi-tuong-class-va-object/)[3](https://www.guru99.com/vi/difference-between-object-and-class.html).

* [**Lớp (Class)**: Lớp là một kế hoạch chi tiết hoặc khuôn mẫu mà từ đó các đối tượng được tạo ra](https://viettuts.vn/java/lop-va-doi-tuong-trong-java)[4](https://bing.com/search?q=Ph%C3%A2n+bi%E1%BB%87t+l%E1%BB%9Bp+v%C3%A0+%C4%91%E1%BB%91i+t%C6%B0%E1%BB%A3ng+trong+Java)[5](https://askanydifference.com/vi/difference-between-class-and-object-in-java/). [Nói cách khác, một lớp chỉ là một bản thiết kế hoặc một tập hợp các hướng dẫn để xây dựng các đặc tính của một đối tượng cụ thể sau này](https://viettuts.vn/java/lop-va-doi-tuong-trong-java)[2](https://funix.edu.vn/chia-se-kien-thuc/phan-bie%CC%A3t-lop-va-doi-tuong-class-va-object/). [Một lớp trong Java có thể chứa: thành viên dữ liệu, constructor, phương thức, khối lệnh](https://viettuts.vn/java/lop-va-doi-tuong-trong-java)[1](https://viettuts.vn/java/lop-va-doi-tuong-trong-java).

Ví dụ về lớp trong Java:

public class Student {

int id; // thành viên dữ liệu

String name; // thành viên dữ liệu

}

* [**Đối tượng (Object)**: Đối tượng là một thể hiện của một lớp](https://viettuts.vn/java/lop-va-doi-tuong-trong-java)[1](https://viettuts.vn/java/lop-va-doi-tuong-trong-java)[2](https://funix.edu.vn/chia-se-kien-thuc/phan-bie%CC%A3t-lop-va-doi-tuong-class-va-object/)[3](https://www.guru99.com/vi/difference-between-object-and-class.html). [Mỗi đối tượng có ba đặc điểm: trạng thái (đại diện cho dữ liệu), hành vi (đại diện cho chức năng), và danh tính (được cài đặt thông qua một ID duy nhất)1](https://viettuts.vn/java/lop-va-doi-tuong-trong-java). [Đối tượng cấp phát bộ nhớ khi tạo](https://viettuts.vn/java/lop-va-doi-tuong-trong-java)[4](https://bing.com/search?q=Ph%C3%A2n+bi%E1%BB%87t+l%E1%BB%9Bp+v%C3%A0+%C4%91%E1%BB%91i+t%C6%B0%E1%BB%A3ng+trong+Java)[5](https://askanydifference.com/vi/difference-between-class-and-object-in-java/).

Ví dụ về đối tượng trong Java:

Student student1 = new Student(); // tạo một đối tượng student1

Vậy, sự khác biệt giữa lớp và đối tượng là gì? [Một lớp là một bản thiết kế hoặc nguyên mẫu xác định các biến và phương thức (hoặc hàm) chung cho tất cả các đối tượng cùng kiểu. Đối tượng là các thể hiện cụ thể của các lớp](https://askanydifference.com/vi/difference-between-class-and-object-in-java/)[4](https://bing.com/search?q=Ph%C3%A2n+bi%E1%BB%87t+l%E1%BB%9Bp+v%C3%A0+%C4%91%E1%BB%91i+t%C6%B0%E1%BB%A3ng+trong+Java)[5](https://askanydifference.com/vi/difference-between-class-and-object-in-java/)[3](https://www.guru99.com/vi/difference-between-object-and-class.html).

[Trong Java, **Abstract class** và **Interface** đều được sử dụng để có được sự trừu tượng, nhưng có một số sự khác biệt quan trọng giữa chúng](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[1](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[2](https://hocjava.com/abstract-class-va-interface-trong-java/)[3](https://viblo.asia/p/su-khac-nhau-giua-interface-voi-abstract-va-cach-su-dung-chung-trong-java-3Q75wxPDKWb):

1. **Phương thức**: Abstract class có thể có cả phương thức abstract (không có thân hàm) và phương thức non-abstract (có thân hàm). Trong khi đó, Interface chỉ có phương thức abstract. [Từ Java 8, Interface có thêm các phương thức default và static](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[1](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface).
2. **Kế thừa**: Abstract class không hỗ trợ đa kế thừa. [Trái lại, Interface hỗ trợ đa kế thừa](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[1](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface).
3. **Biến**: Abstract class có thể có các biến final, non-final, static và non-static. [Trong khi đó, Interface chỉ có các biến static và final](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[1](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface).
4. **Cài đặt phương thức**: Abstract class có thể cung cấp nội dung cài đặt cho phương thức của Interface. [Ngược lại, Interface không thể cung cấp nội dung cài đặt cho phương thức của Abstract class](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[1](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface).
5. **Từ khóa**: Từ khóa abstract được sử dụng để khai báo Abstract class. [Trong khi đó, từ khóa interface được sử dụng để khai báo Interface](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[1](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface).

[Về mặt thiết kế, Interface thường được dùng để chỉ rằng 2 hay nhiều class cùng làm việc gì đó (ví dụ cùng in - Printable), trong khi Abstract class thường hướng đến quan hệ cha con trong lập trình hướng đối tượng](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[4](https://bing.com/search?q=s%E1%BB%B1+kh%C3%A1c+nhau+gi%E1%BB%AFa+abstract+v%C3%A0+interface+trong+Java). [Đơn giản để hiểu, đó là Abstract class có được trừu tượng 1 phần (0 tới 100%), còn Interface có được trừu tượng toàn phần (100%)](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[1](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface).

abstract class Person {  
 abstract void show();  
}  
class SinhVien extends Person {  
 void show() {  
 System.*out*.println("Sinh Vien");  
 }  
}  
Person a = new Person() {  
 void show() {}  
};  
a.show();

Để sử dụng interface thay cho abstract class trong ví dụ của bạn, chúng ta có thể thực hiện như sau:

interface Person {

void show();

}

class SinhVien implements Person {

public void show() {

System.out.println("Sinh Vien");

}

}

Person a = new SinhVien();

a.show();

Trong đoạn mã trên, Person được khai báo như một interface thay vì một abstract class. Lớp SinhVien thực hiện (implements) interface Person và cung cấp cài đặt cho phương thức show(). [Cuối cùng, chúng ta tạo một đối tượng a của kiểu Person và gán nó là một đối tượng mới của lớp SinhVien, sau đó gọi phương thức show()1](https://hocjava.com/abstract-class-va-interface-trong-java/)[2](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[3](https://viblo.asia/p/abstract-class-va-interface-ban-co-thuc-su-hieu-chung-bJzKmwNrl9N)[4](https://hocjava.com/abstract-class-trong-java/)[5](https://viblo.asia/p/su-khac-nhau-giua-interface-voi-abstract-va-cach-su-dung-chung-trong-java-3Q75wxPDKWb).

[Lưu ý rằng, khi sử dụng interface, tất cả các phương thức đều mặc định là public và abstract](https://hocjava.com/abstract-class-va-interface-trong-java/)[1](https://hocjava.com/abstract-class-va-interface-trong-java/)[2](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[3](https://viblo.asia/p/abstract-class-va-interface-ban-co-thuc-su-hieu-chung-bJzKmwNrl9N)[4](https://hocjava.com/abstract-class-trong-java/)[5](https://viblo.asia/p/su-khac-nhau-giua-interface-voi-abstract-va-cach-su-dung-chung-trong-java-3Q75wxPDKWb). [Do đó, khi một lớp thực hiện một interface, nó phải cung cấp cài đặt cho tất cả các phương thức của interface đó1](https://hocjava.com/abstract-class-va-interface-trong-java/)[2](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[3](https://viblo.asia/p/abstract-class-va-interface-ban-co-thuc-su-hieu-chung-bJzKmwNrl9N)[4](https://hocjava.com/abstract-class-trong-java/)[5](https://viblo.asia/p/su-khac-nhau-giua-interface-voi-abstract-va-cach-su-dung-chung-trong-java-3Q75wxPDKWb). [Trong trường hợp này, lớp SinhVien phải cung cấp cài đặt cho phương thức show()](https://hocjava.com/abstract-class-va-interface-trong-java/)[1](https://hocjava.com/abstract-class-va-interface-trong-java/)[2](https://viettuts.vn/java/su-khac-nhau-giua-abstract-class-va-interface)[3](https://viblo.asia/p/abstract-class-va-interface-ban-co-thuc-su-hieu-chung-bJzKmwNrl9N)[4](https://hocjava.com/abstract-class-trong-java/)[5](https://viblo.asia/p/su-khac-nhau-giua-interface-voi-abstract-va-cach-su-dung-chung-trong-java-3Q75wxPDKWb).