**[JAVA COLLECTION](https://classroom.google.com/w/Njc0NDQwMTUzOTQ4/tc/Njc0NDQ3MDQ0NzIz)**

**1. Array**

# **1. Mảng là gì trong Java?**

* Mảng là một cấu trúc dữ liệu cho phép lưu trữ nhiều giá trị của cùng một kiểu dữ liệu dưới một tên biến. Mảng có kích thước cố định, nghĩa là bạn không thể thay đổi kích thước của nó sau khi đã được khởi tạo.

# **2. Cách truy cập phần tử trong mảng?**

* Bạn có thể truy cập phần tử của mảng bằng cách sử dụng chỉ số (index)



# **3. Mảng đa chiều là gì?**

* Mảng đa chiều là mảng có nhiều hơn một chỉ số, thường được sử dụng để lưu trữ dữ liệu trong dạng bảng



# **4. Cách sao chép mảng trong Java?**



# **5. Mảng có thể chứa kiểu dữ liệu nào?**

* Mảng trong Java có thể chứa kiểu dữ liệu nguyên thủy (primitive types) như int, char, boolean, hoặc các kiểu đối tượng (object types) như String, Object, v.v.

# **6. Sự khác biệt giữa mảng và ArrayList trong Java?**

* Mảng có kích thước cố định và không thể thay đổi sau khi đã được khởi tạo, trong khi ArrayList là một lớp trong Java Collections Framework cho phép thêm, xóa, và điều chỉnh kích thước một cách linh hoạt.

# **7. Dùng kiểu mảng mang lại ưu , nhược điểm gì?**

### **Ưu điểm của mảng trong Java**

1. **Hiệu suất cao**:
   * Mảng có hiệu suất tốt vì nó lưu trữ dữ liệu một cách liên tục trong bộ nhớ, giúp truy cập nhanh chóng thông qua chỉ số (index). Truy cập phần tử bất kỳ trong mảng có độ phức tạp thời gian là O(1).
2. **Dễ dàng quản lý dữ liệu cố định**:
   * Mảng là lựa chọn tốt khi bạn biết trước số lượng phần tử cần lưu trữ và không cần thay đổi kích thước của mảng trong suốt vòng đời chương trình.
3. **Đơn giản trong khái niệm và sử dụng**:
   * Mảng là một trong những cấu trúc dữ liệu đơn giản nhất để hiểu và sử dụng. Việc khai báo, khởi tạo và truy cập mảng khá dễ dàng.
4. **Sử dụng ít bộ nhớ hơn so với các cấu trúc dữ liệu khác**:
   * Mảng có xu hướng sử dụng ít bộ nhớ hơn so với các cấu trúc dữ liệu linh hoạt như ArrayList, vì không cần thêm các yếu tố quản lý dữ liệu như kích thước hiện tại, dung lượng, v.v.
5. **Hỗ trợ kiểu dữ liệu nguyên thủy**:
   * Mảng có thể lưu trữ các kiểu dữ liệu nguyên thủy (primitive types) như int, char, double, v.v. mà không cần phải bọc trong các lớp đối tượng như Integer hay Double, giúp tối ưu hóa bộ nhớ.
6. **Dễ dàng lặp qua và thao tác**:
   * Việc sử dụng các vòng lặp để lặp qua tất cả các phần tử của mảng là rất dễ dàng và hiệu quả.

### **Nhược điểm của mảng trong Java**

1. **Kích thước cố định**:
   * Một khi mảng đã được khởi tạo, bạn không thể thay đổi kích thước của nó. Điều này khiến mảng trở nên kém linh hoạt khi làm việc với dữ liệu có số lượng không xác định trước. Nếu bạn cần thêm hoặc bớt phần tử, bạn phải tạo một mảng mới có kích thước lớn hơn và sao chép dữ liệu.
2. **Thiếu tính năng linh hoạt**:
   * So với các cấu trúc dữ liệu như ArrayList, mảng thiếu các phương thức hữu ích như thêm, xóa, tìm kiếm phần tử mà không cần tự triển khai thủ công.
3. **Không kiểm soát tốt việc xử lý ngoại lệ**:
   * Mảng dễ gặp phải các lỗi phổ biến như ArrayIndexOutOfBoundsException nếu bạn cố truy cập phần tử nằm ngoài giới hạn của mảng. Điều này làm tăng khả năng xảy ra lỗi nếu không cẩn thận trong việc quản lý chỉ số.
4. **Không hỗ trợ tự động mở rộng**:
   * Với các cấu trúc như ArrayList, mảng tự động mở rộng khi cần thêm phần tử. Tuy nhiên, mảng cố định trong Java không có tính năng này, điều này khiến việc sử dụng mảng không phù hợp với các tình huống yêu cầu kích thước thay đổi liên tục.
5. **Thiếu tính năng xử lý đối tượng**:
   * Khi sử dụng mảng với các đối tượng, bạn cần quản lý từng đối tượng riêng lẻ. Không có các phương thức hỗ trợ việc thao tác với các đối tượng dễ dàng như trong các lớp thuộc Java Collections Framework (ArrayList, Set, v.v.).
6. **Không dễ dàng xóa hoặc chèn phần tử**:
   * Nếu bạn muốn xóa một phần tử trong mảng, bạn phải dịch chuyển tất cả các phần tử còn lại để lấp vào vị trí đã xóa, gây ra độ phức tạp là O(n). Điều này khiến mảng không phù hợp cho các thao tác thêm/xóa thường xuyên.