Mảng là một tập hợp các phần tử có cùng kiểu dữ liệu được lưu trữ liên tiếp trong bộ nhớ. Trong Java, mảng được coi là một đối tượng, vì vậy khi tạo mảng, ta phải khai báo kích thước và kiểu dữ liệu của nó trước.

Cú pháp:

C1: int[] arr = {1, 2, 3, 4, 5}; // Mảng có 5 phần tử

C2: int[] arr; // Khai báo mảng số nguyên

arr = new int[5]; // Cấp phát 5 ô nhớ cho mảng

Các tính chất chính của mảng:

* Kích thước cố định: Số phần tử của mảng phải được xác định ngay từ khi tạo và không thể thay đổi.
* Chỉ số (index): Phần tử đầu tiên có chỉ số là 0 và phần tử cuối cùng có chỉ số là n - 1 (với n là kích thước mảng).
* Các phần tử liên tiếp: Bộ nhớ cấp phát liên tục cho các phần tử.
* Mảng đa chiều: Java hỗ trợ mảng nhiều chiều, chẳng hạn như mảng hai chiều dùng để lưu bảng hoặc ma trận.

**Ưu điểm:**

* Truy cập nhanh: Mảng cho phép truy cập ngẫu nhiên (random access) thông qua chỉ số, tức là có thể lấy phần tử bất kỳ trong O(1) thời gian.
* Tối ưu về bộ nhớ: Mảng lưu các phần tử liên tiếp trong bộ nhớ nên có hiệu suất truy xuất tốt và ít gây phân mảnh bộ nhớ.
* Đơn giản: Việc khai báo và sử dụng mảng tương đối dễ hiểu, thuận tiện khi cần xử lý một tập hợp dữ liệu đồng nhất.

**Nhược điểm:**

* Kích thước cố định: Kích thước của mảng phải được xác định khi tạo và không thể thay đổi trong quá trình chạy. Điều này gây lãng phí nếu khai báo dư thừa, hoặc thiếu chỗ khi cần mở rộng.
* Không hỗ trợ chèn/xóa linh hoạt: Khi cần thêm/xóa phần tử giữa mảng, bạn phải dịch chuyển các phần tử còn lại, dẫn đến hiệu suất kém.
* Không có phương thức tích hợp: So với các cấu trúc dữ liệu như ArrayList, mảng không có các phương thức tích hợp như thêm, xóa hay tìm kiếm. Người lập trình phải tự viết mã để thực hiện những thao tác này.

Nên sử dụng mảng khi:

* Khi biết trước kích thước cố định của dữ liệu.
* Khi cần truy cập nhanh đến các phần tử theo chỉ số.
* Khi hiệu suất là ưu tiên hàng đầu và không cần nhiều thao tác thêm/xóa.