**Task 3 - Mảng và chuỗi**

1. **Mảng**

- Mảng là kiểu cấu trúc dữ liệu lưu trữ các phần tử cùng kiểu, có kích thước cố định.

- Mảng là kiểu dữ liệu tham chiếu

- Mảng quản lý các phần tử theo chỉ số (index).

- Ưu điểm:

* Viết mã gọn hơn: gom các phần tử liên quan lại với nhau.
* Truy cập ngẫu nhiên theo chỉ mục.
* Dễ dàng thao tác, quản lý.

- Nhược điểm:

* Giới hạn kích thước
* Yêu cầu các ô nhớ liên tiếp trong không gian bộ nhớ

1. **Mảng một chiều**

- Khai báo:

* *dataType[] arrayName; //Phong cách của Java*
* *dataType arrayName[]; //Kế thừa từ C/C++*

- Khởi tạo mảng:

* *dataType[] arrayName = new dataType[size];*
* *dataType[] arrayName = { value1, value2, ... }; //value phải có cùng kiểu dữ liệu là dataType*

- Truy cập phần tử mảng thông qua chỉ mục: *arrayName[index]*

1. **Mảng nhiều chiều**

- Mảng nhiều chiều là một mảng gồm các mảng

- Ví dụ về mảng 2 chiều:

* Khai báo: *int[][] array2D = new int[2][3]; // 2 hàng, 3 cột*
* Khởi tạo: *int[][] array2D = {{1,2,3}, {4,5,6}};*
* Truy cập phần tử mảng theo chỉ mục hàng và chỉ mục cột: *array2D[0][1]; // 2*

1. **Lớp Arrays**

- Lớp Arrays thuộc gói java.util cung cấp các phương thức tĩnh để làm việc với mảng.

- Một số phương thức phổ biến: *sort(), binarySearch(), equals(), fill(), toString(), asList(), copyOf()...*

1. **Chuỗi**
2. **Lớp String**

- Được định nghĩa trong gói java.lang, là đối tượng bất biến.

- Khai báo: *String var;* hoặc *String var = new String();*

- Khởi tạo:

* Toán tử new: *String var = new String(“Chuỗi”);*
* Toán tử gán: *String var = “Chuỗi”;*
* Khai báo trong dấu nháy kép

- Sử dụng:

* Với toán tử new, các đối tượng riêng biệt sẽ được tạo ra và lưu trữ ở những nơi khác nhau trong bộ nhớ. Nếu so sánh hai đối tượng sử dụng toán tử quan hệ (“==”), kết quả luôn là **false**.
* Với toán tử gán (“=”) và sử dụng dấu nháy kép, các biến được lưu ở một vị trí cụ thể gọi là String pool. Nếu so sánh hai biến giống nhau bằng toán tử quan hệ thì kết quả là **true**.

1. **Các phương thức chuỗi**

- *length()*: Trả về độ dài chuỗi.

- *charAt(index)*: Trả về ký tự ở vị trí index trong chuỗi (0 <= index < length()).

- *substring(beginIdx, n)*: Trả về chuỗi con có độ dài bằng **n** bắt đầu từ chỉ số beginIdx. Nếu n không được chỉ định, trả về chuỗi con từ beginIdx đến cuối chuỗi.

- *indexOf(inp, fromIndex)*: Trả về vị trí xuất hiện đầu tiên của **inp** trong chuỗi (inp có thể là ký tự hoặc chuỗi). Nếu **fromIndex** không được chỉ định, mặc định tìm từ đầu chuỗi.

- Phương thức khác: *toUpperCase(), toLowerCase(), concat(), equals(),...*

1. **Lớp StringBuilder và StringBuffer**

- Được định nghĩa trong gói java.lang.

- Cả StringBuilder và StringBuffer đều là các lớp được dùng để thao tác với chuỗi ký tự có thể thay đổi.

- Khởi tạo:

* *StringBuilder objName = new StringBuilder(content);*
* *StringBuffer objName = new StringBuffer(content);*

- Các phương thức: *append(), insert(), delete(), reverse(),...*

- Hiệu quả hơn String khi cần thực hiện các thao tác chỉnh sửa chuỗi

- Sự khác nhau:

|  |  |  |
| --- | --- | --- |
| **Đặc điểm** | **StringBuilder** | **StringBuffer** |
| Đồng bộ hóa | Không đồng bộ | Đồng bộ |
| Hiệu suất | Nhanh hơn | Chậm hơn |
| An toàn | Không an toàn trong đa luồng | An toàn trong đa luồng |
| Nên dùng khi | Không có nhiều luồng truy cập cùng lúc | Có nhiều luồng truy cập cùng lúc |