Java Fundamental

Bài 4

MẢNG VÀ CHUỖI

Company: DEVPRO VIỆT NAM

Website: devpro.edu.vn

Design by Minh An

Nội dung

- **Mảng**
  - Định nghĩa mảng
  - Truy xuất các phần tử trong mảng
  - Mảng một chiều
  - Mảng đa chiều
- **Chuỗi**
  - String
  - StringBuilder
  - StringBuffer

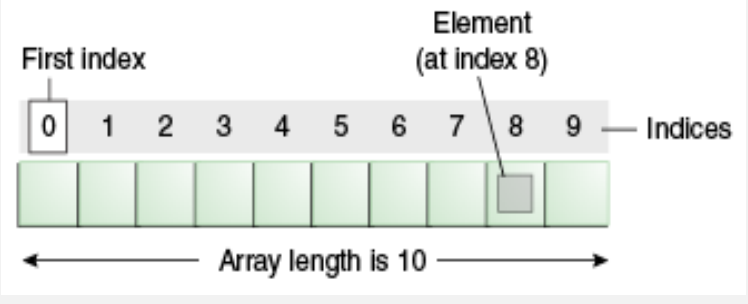
Design by Minh An

Giới thiệu về mảng

- Chỉ một giá trị được lưu trữ trong một biến tại một thời điểm.
- Một vòng lặp đơn không thể truy cập và thao tác trên các giá trị rời rạc và không liên kết với nhau.
- Mảng là một cách lưu trữ dữ liệu đặc biệt, nó có thể lưu trữ nhiều phần tử có cùng một kiểu dữ liệu trong các ô nhớ có vị trí liên tiếp nhau.
- Mảng là cách thức tốt nhất cho việc thao tác trên nhiều phần tử dữ liệu có cùng kiểu tại cùng một thời điểm.

Cấu trúc của mảng

- Mảng có một tên riêng, và mỗi phần tử trong mảng sẽ được truy cập thông qua một số nguyên gọi là chỉ số của mảng, và chỉ số này bắt đầu từ 0.
- Kích cỡ của mảng là số lượng lớn nhất các phần tử mà mảng có thể lưu trữ.



Mảng một chiều

- Mảng một chiều là một dãy liên tiếp các biến có cùng kiểu dữ liệu.
- Khai báo mảng 1 chiều:

```
datatype[] arrayName = new datatype[size];
```

- Ví dụ:

```
int[] arri = new int[10];
float[] arrf = new float[15];
```

- Hoặc khai báo và khởi tạo mảng:

```
double arrd[] = {10.1, 11.3, 12.5,13.7,
                 14.9, 3.21, -5.63};
```

Design by Minh An

Sử dụng Mảng một chiều

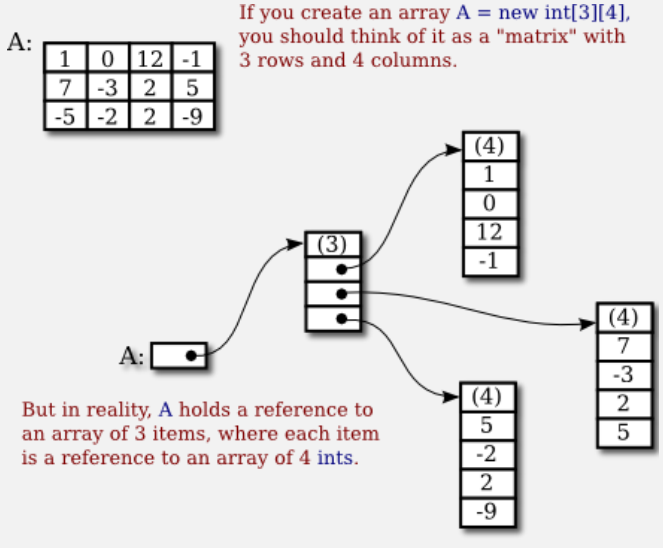
- Mảng một chiều được sử dụng để lưu trữ danh sách.
- Ví dụ:
  1. Tạo mảng a chứa n số, cho biết số lần xuất hiện số x trong mảng a và các vị trí của nó.
  2. Cho số nguyên dương n, tìm mã nhị phân của n, hiển thị kết quả.

Design by Minh An

Mảng hai chiều

- Mảng hai chiều (2D arrays) là mảng một chiều với các phần tử là một mảng một chiều khác.
- Có thể tạo ra mảng ba chiều hay nhiều hơn, nhưng mảng hai chiều được sử dụng nhiều nhất.
- Trong mảng hai chiều, phần tử đầu tiên là: arrayName[0][0].

Mảng hai chiều



Mảng hai chiều

```
// Khai báo một mảng có 5 dòng, 10 cột
DataType[][] myArray1 = new DataType[5][10];
// Khai báo một mảng 2 chiều có 5 dòng.
DataType[][] myArray2 = new DataType[5][];
// Khai báo một mảng 2 chiều, chỉ định giá trị các phần tử.
DataType[][] myArray3 = new DataType[][] {
    { value00, value01, value02 , value03 },
    { value10, value11, value12 }
};
```


Mảng hai chiều

- Ví dụ:
  - Tạo ma trận a với m dòng, n cột chứa các số
  - Hiển thị ra màn hình:
    - Ma trận a sau khi tạo
    - Dòng có tổng lớn nhất và giá trị của tổng đó
    - Cột có tổng nhỏ nhất và giá trị của tổng đó
    - Ma trận b là ma trận chuyển vị của ma trận a
    - Tích của ma trận a và ma trận b

Lớp String

- Dữ liệu văn bản (text data) là một tập hợp các ký tự rời rạc mà chúng kết hợp với nhau để tạo thành các từ có nghĩa.
- Java cung cấp lớp String để biểu diễn dữ liệu văn bản.
- Chuỗi (String) là bất biến, nghĩa là giá trị của nó không thể thay đổi sau khi chúng được tạo ra.
- Một đối tượng chuỗi có thể được tạo ra như sau:

```
String str = new String();
```

- Để gán dữ liệu cho một biến chuỗi:

```
str = "Hello World";
```


Lớp String

- Một số phương thức quan trọng trong lớp **String**:
  - **length()**: viết là str.length() - trả về độ dài của chuỗi str
  - **charAt(i)**: str.charAt(i) - trả về ký tự str[i]
  - **indexOf()**: str.indexOf(c | s) trả về vị trí (chỉ số) xuất hiện đầu tiên của ký tự c hay chuỗi s nếu c hay s có trong chuỗi str, ngược lại trả về -1.
  - **concat(s)**: str.concat(s) – trả về chuỗi mới bằng cách cộng chuỗi s vào cuối str.

Lớp String

- Một số phương thức quan trọng trong lớp **String**:
  - **compareTo(s)**: str.compareTo(s) - so sánh hai chuỗi str và chuỗi s (theo thứ tự bảng chữ cái – thứ tự từ điển – phân biệt chữ hoa và chữ thường) và trả về một số nguyên.
    - Số nguyên âm nếu chuỗi str đứng trước (nhỏ hơn) chuỗi s.
    - Số nguyên dương nếu chuỗi str đứng sau (lớn hơn) chuỗi s.
    - Số không nếu chuỗi str giống hệt chuỗi s.
  - **compareToInogeCase()**: so sánh 2 chuỗi không phân biệt chữ hoa, chữ thường.

Lớp String

- Một số phương thức quan trọng trong lớp **String**:
  - **str.lastIndexOf(c|s)**: trả về vị trí (chỉ số) xuất hiện cuối cùng của ký tự c hay chuỗi s trong chuỗi str.
  - **str.replace(c|s, cn|sn)**: trả về chuỗi mới sau khi thay thế tất cả các ký tự c hay chuỗi s bằng ký tự cn hay chuỗi cn trong chuỗi str.
  - **str.substring(m[, n])**: trả về một chuỗi con lấy từ vị trí m [đến vị trí n] trong chuỗi str (không có n thì lấy đến cuối str).
  - **str.toString()**: trả về đối tượng chuỗi.
  - **str.trim()**: cắt các khoảng trắng ở đầu và cuối chuỗi str.
  - **str.split()**: cắt xâu ra thành một mảng các phần tử dựa vào xâu đầu vào
  - .

Array of String

- Là một mảng mà các phần tử mảng là các chuỗi.
- Khai báo và khởi tạo một mảng các đối tượng chuỗi:

  String[] **numbers** = new String[10];
- Cần phải cấp phát bộ nhớ cho từng đối tượng của mảng:

  numbers[0]= new String("twenty");

Hoặc: **str = {"one two three four five six seven eight nine ten"};**

numbers = str.split(" ");

Hoặc: **String[] numbers = {"one", "two", "three", "four", "five",**

"six", "seven", "eight", "nine", "ten"};

Ta có:

numbers[0] = "one", numbers[1] = "two", …, numbers[9] = "ten"

Lớp StringBuilder

- Các đối tượng của lớp **StringBuilder** có thể mở rộng và rất linh hoạt, nghĩa là đối tượng của lớp **StringBuilder** có thể thay đổi được giá trị.
- Đối tượng **StringBuilder** có thể được tạo ra bằng những hàm tạo sau:
  - `StringBuilder()`
  - `StringBuilder(int capacity)`
  - `StringBuilder(String str)`

Lớp StringBuilder

- **append(str):** nối giá trị chuỗi str vào cuối đối tượng StringBuilder.
- **insert(index, str):** chèn giá trị chuỗi str vào trong đối tượng StringBuilder tại vị trí index.
- **delete(startIndex, endIndex):** xóa các ký tự từ vị trí startIndex đến vị trí endIndex trong đối tượng StringBuilder.
- **reverse():** đảo ngược các ký tự bên trong đối tượng StringBuilder.

Lớp StringBuffer

- Giống như StringBuider, StringBuffer cũng phục vụ cho các trường hợp chúng ta cần làm việc với các chuỗi String mà cần mở rộng và linh hoạt.
- Điểm khác biệt cơ bản giữa StringBuffer và StringBuilder là StringBuffer là an toàn luồng (thread-safe): nhiều thread không thể truy nhập nó đồng thời.

Lớp StringBuffer - Constructor

- StringBuffer(): Khởi tạo một StringBuffer rỗng với dung tích mặc định là 16 ký tự.
- StringBuffer(CharSequence seq): Khởi tạo một StringBuffer với một chuỗi ký tự cho sẵn chứa trong nó.
- StringBuffer(int capacity): Khởi tạo một StringBuffer rỗng với dung tích xác định.
- StringBuffer(String str): Khởi tạo một StringBuffer với nội dung lấy từ một chuỗi string

Lớp StringBuffer - Methods

- public StringBuffer append(String s)
- public StringBuffer reverse()
- public delete(int start, int end)
- public insert(int offset, int i)
- replace(int start, int end, String str)
- int capacity() - Returns the current capacity of the StringBuffer.
- char charAt(int index)
- void ensureCapacity(int minimumCapacity)
- void getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin)
- int indexOf(String str)

Lớp StringBuffer - Methods

- int indexOf(String str, int fromIndex)
- int lastIndexOf(String str)
- int lastIndexOf(String str, int fromIndex)
- int length()
- void setCharAt(int index, char ch)
- void setLength(int newLength) - Sets the length of this String buffer.
- CharSequence subSequence(int start, int end)
- String substring(int start)
- String substring(int start, int end)
- String toString()

Bài tập – Sử dụng mảng & mảng String

- Tạo một danh sách ghi tên các chủ hộ của các gia đình trong một tổ dân phố (danh sách có từ 10 hộ gia đình trở lên).
- Tạo một danh sách ghi số kwh điện tiêu thụ của các hộ gia đình tương ứng với danh sách chủ hộ.
- Tính số tiền phải trả tương ứng của các hộ gia đình.
- Hiển thị danh sách các hộ gia đình cùng với số kwh điện tiêu thụ và số tiền phải trả, mỗi hộ gia đình trên 1 dòng.
- Hiển thị 3 hộ gia đình có số kwh điện tiêu thụ nhiều nhất.
- Hiển thị các hộ gia đình có chủ hộ tên "Truong".
- Sắp danh sách các hộ gia đình theo tên với thứ tự bảng chữ cái.
- Sắp xếp danh sách các hộ gia đình theo số kwh điện tăng dần.
- Chương trình có menu chọn các chức năng.

Bài tập về nhà – Sử dụng mảng & mảng String

- Một vòng thi đấu cử tạ gồm 10 vận động viên, mỗi vận động viên cử tạ 3 lần.
- Viết chương trình (có menu chọn chức năng) quản lý vòng đấu gồm các chức năng sau:
- Tạo danh sách (ghi tên) các vận động viên và danh sách ghi thành tích của các vận động viên trong 3 lần cử tạ.
- Hiển thị danh sách vận động viên cùng với thành tích trong 3 lần cử tạ và tổng thành tích.
- Hiển thị thông tin về vận động viên có tổng thành tích (tổng 3 lần cử tạ) cao nhất.
- Hiển thị 3 vận động viên có tổng thành tích cao nhất.
- Sắp sếp danh sách vận động viên theo tên (lưu ý kèm thành tích).

Bài tập về nhà – Sử dụng mảng & mảng String

- Hiển thị danh sách vận động viên có tên bắt đầu bằng chữ 'T'.
- Hiển thị danh sách vận động viên mà tên có vấn "an".
- Sắp xếp danh sách vận động viên theo chiều dài của tên tăng dần.
- Hiển thị danh sách các vận động viên có cùng tổng thành tích.
- Hiển thị danh sách các vận động viên có lần cử tạ thứ 3 thấp nhất.
- Hiển thị danh sách các vận động viên có lần cử tạ thứ nhất cao nhất.

Design by Minh An