BKACAD®

HỌC VIỆN CÔNG NGHỆ BKACAD

Lập trình Java - Mảng (Arrays)

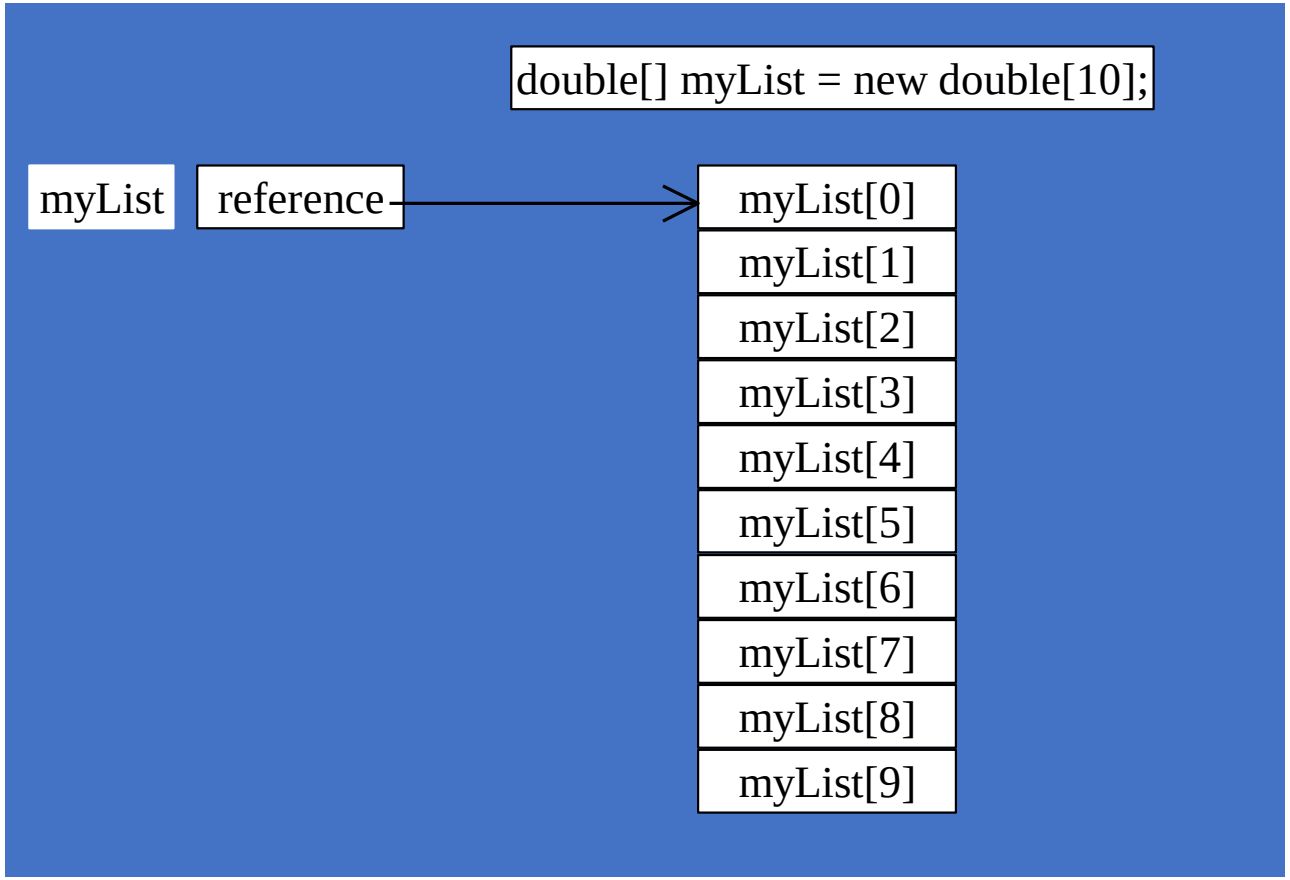
Ths. Vũ Duy Khương

Nội dung

1. Giới thiệu mảng
2. Khai báo, khởi tạo mảng
3. Truyền mảng cho phương thức
4. Mảng nhiều chiều
5. Các phương thức tìm kiếm và sắp xếp

Giới thiệu mảng

Mảng là một cấu trúc dữ liệu biểu diễn một tập các dữ liệu cùng kiểu.



Một mảng 10 phần tử kiểu `double`

Khai báo biến mảng

- `datatype[] arrayname;`

  Ví dụ:

  ```
  double[] myList;
  ```

- `datatype arrayname[];`

  Ví dụ:

  ```
  double myList[];
  ```

Tạo mảng

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

Ví dụ:

```
myList = new double[10];
```

`myList[0]` tham chiếu phần tử đầu tiên của mảng.

`myList[9]` tham chiếu phần tử cuối cùng của mảng.

Khai báo và tạo mảng trong một bước

- `datatype[] arrayname = new datatype[arraySize];`

```
double[] myList = new double[10];
```

- `datatype arrayname[] = new datatype[arraySize];`

```
double myList[] = new double[10];
```

Độ dài mảng

- Mỗi khi mảng được tạo, kích thước của nó được ấn định, không thể thay đổi. Bạn có thể lấy kích thước mảng bằng cách gọi:

arrayVariable.length

Ví dụ:

myList.length trả về giá trị 10

Khởi tạo mảng

- Sử dụng vòng lặp:

```
for (int i = 0; i < myList.length; i++)
  myList[i] = i;
```

- Khai báo, tạo, khởi tạo trong một lệnh:

```
double[] myList = {1.9, 2.9, 3.4, 3.5};
```

- Lưu ý: chỉ trong 1 lệnh, nhiều hơn 1 lệnh là sai:

```
double[] myList;
myList = {1.9, 2.9, 3.4, 3.5};
```

Khai báo, tạo, khởi tạo

```
double[] myList = {1.9, 2.9, 3.4, 3.5};
```

Câu lệnh trên tương đương với các câu lệnh sau:

```
double[] myList = new double[4];
myList[0] = 1.9;
myList[1] = 2.9;
myList[2] = 3.4;
myList[3] = 3.5;
```

Ví dụ 1

- Mục đính: Chương trình nhận 6 số từ bàn phím, tìm số lớn nhất và đếm số lần xuất hiện của giá trị đó.

  Giả sử bạn nhập vào 3, 5, 2, 5, 5, và 5, số lớn nhất là 5 và số lần xuất hiện là 4.

Ví dụ 2: Xếp loại

- Mục tiêu: đọc vào điểm của SV (int) từ bàn phím, lấy điểm cao nhất (best), rồi xếp loại theo quy tắc sau:
  - Loại A: điểm >= best–10;
  - Loại B: điểm >= best–20;
  - Loại C: điểm >= best–30;
  - Loại D: điểm >= best–40;
  - Trái lại là loại F.

Truyền mảng cho phương thức

- Java sử dụng *truyền tham trị* để truyền các tham số cho phương thức. Có nhiều sự khác nhau quan trọng khi truyền tham trị của biến có kiểu dữ liệu cơ sở và biến mảng.
- Với tham số kiểu dữ liệu cơ sở, giá trị thực được truyền. Thay đổi giá trị của tham số cục bộ trong phương thức không làm thay đổi giá trị của biến bên ngoài phương thức.
- Với tham số kiểu mảng, giá trị của tham số chứa một tham chiếu tới mảng; tham chiếu này được truyền cho phương thức. Bất kỳ sự thay đổi nào xuất hiện trong thân phương thức sẽ làm thay đổi mảng gốc được truyền.

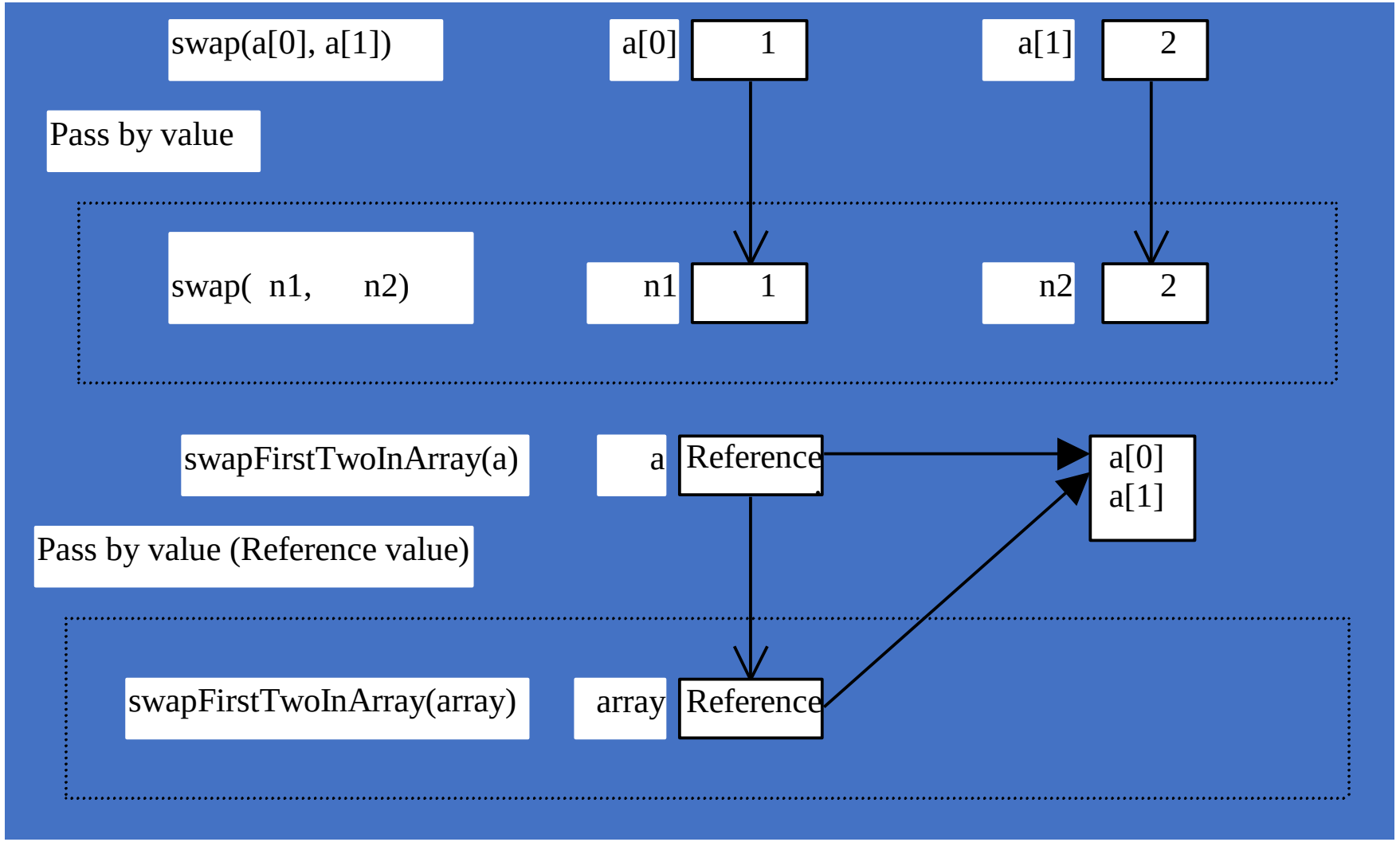
Ví dụ 3: Truyền tham số kiểu mảng

- Mục tiêu: Minh họa sự khác nhau giữa truyền tham số kiểu dữ liệu cơ sở và kiểu mảng.

Ví dụ 3: Truyền tham số kiểu mảng

```
328 public class TestArray2 {
329     static void min(int arr[]) {
330         int min = arr[0];
331         for (int i = 1; i < arr.length; i++)
332             if (min > arr[i]) {
333                 min = arr[i];
334             }
335         System.out.println(min);
336     }
337 
338     public static void main(String args[]) {
339         int a[] = { 33, 3, 4, 5 };
340         min(a);// truyền mảng tới phương thức
341     }
342 }
```

Ví dụ 3: Truyền tham số kiểu mảng (tiếp)



Ví dụ 4: Sử dụng mảng tính độ lệch

$$mean = \frac{\sum_{i=1}^{n} x_i}{n}$$

$$deviation = \sqrt{\frac{\sum_{i=1}^{n} (x_i - mean)^2}{n-1}}$$

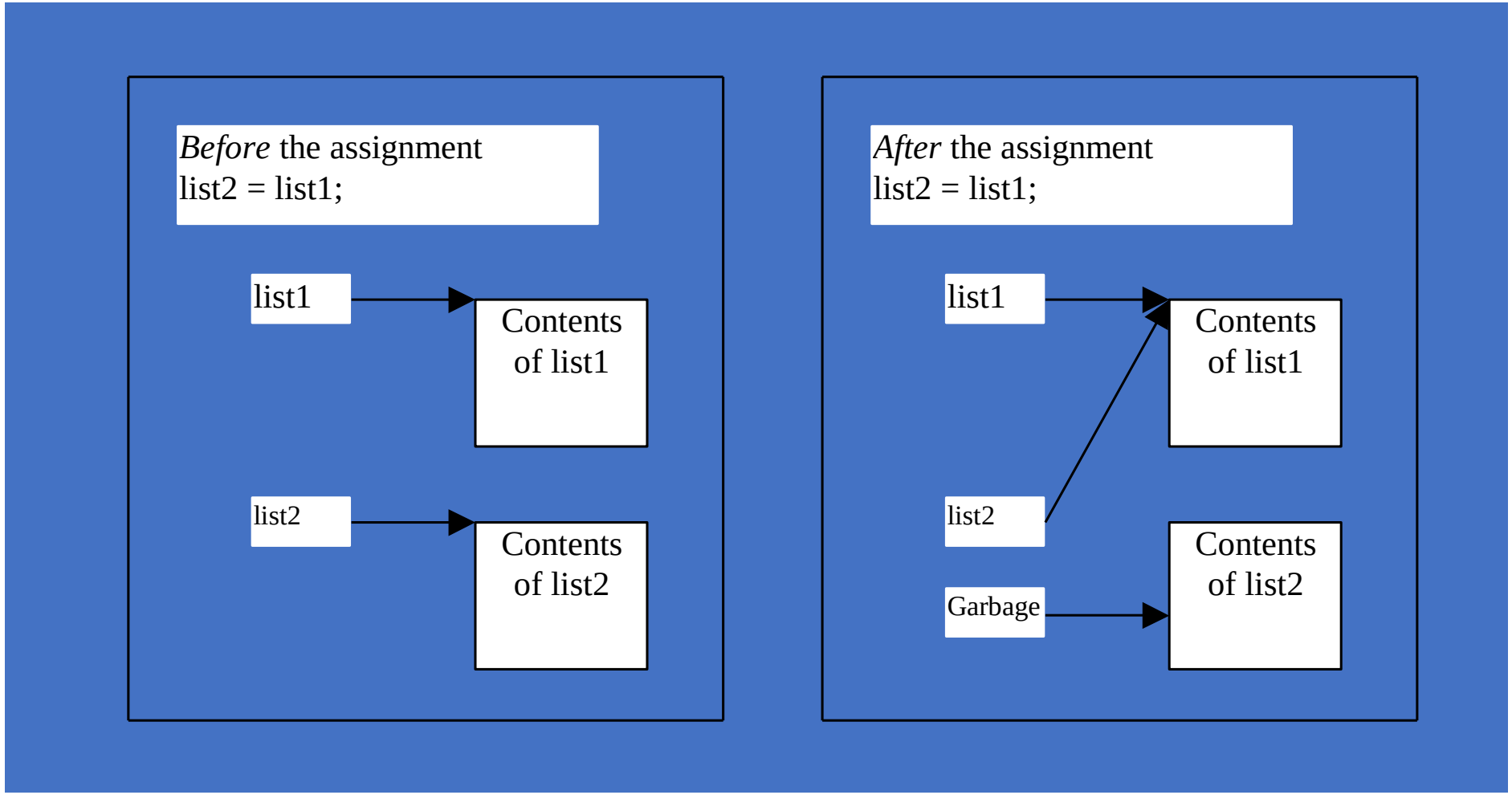
Ví dụ 5: Đếm tần số xuất hiện của các ký tự

- Sinh ngẫu nhiên 100 ký tự chữ thường và gán cho 1 mảng ký tự.
- Đếm tần số xuất hiện của mỗi ký tự trong mảng.
- Tìm trung bình (mean) và độ lệch chuẩn (standard deviation) của các lần đếm.

Ví dụ 6: Copy mảng

Trong ví dụ này, bạn sẽ thấy rằng một phép gán đơn giản không thể copy mảng. Chương trình chỉ đơn giản tạo 2 mảng và định copy từ mảng này sang mảng kia sử dụng một câu lệnh gán.

Ví dụ 6: Copy mảng (tiếp)



Ví dụ 6: Copy mảng (tiếp)

Sử dụng một vòng lặp:

```
int[] sourceArray = {2, 3, 1, 5, 10};
int[] targetArray = new int[sourceArray.length];


for (int i = 0; i < sourceArrays.length; i++)
   targetArray[i] = sourceArray[i];
```

Ví dụ 6: Copy mảng (tiếp)

```
public static void arraycopy(
        Object src, int srcPos,Object dest, int destPos, int length
)
```

Ví dụ 6: Copy mảng (tiếp)

```
public class TestArrayCopyDemo {
    public static void main(String[] args) {
        char[] copyFrom = { 'd', 'e', 'c', 'a', 'f',
                'f', 'e', 'i', 'n', 'a', 't', 'e', 'd' };
        char[] copyTo = new char[7];

        System.arraycopy(copyFrom, 2, copyTo, 0, 7);
        System.out.println(new String(copyTo));
    }
```

Tiện ích arraycopy

```
arraycopy(sourceArray, src_pos, targetArray, tar_pos,
  length);
```

Ví dụ:

```
System.arraycopy(sourceArray, 0, targetArray, 0,
  sourceArray.length);
```

Mảng nhiều chiều

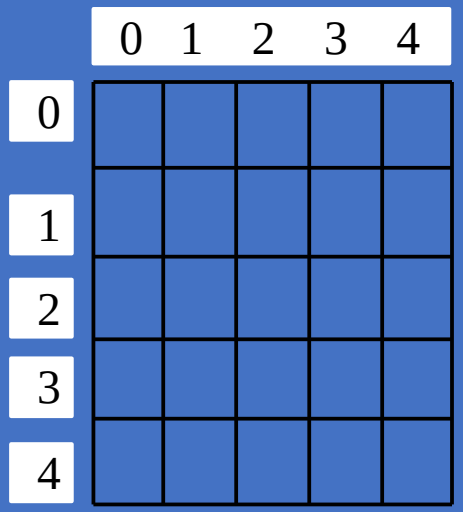
Khai báo và tạo biến mảng nhiều chiều:

```
int[][] matrix = new int[10][10];
```

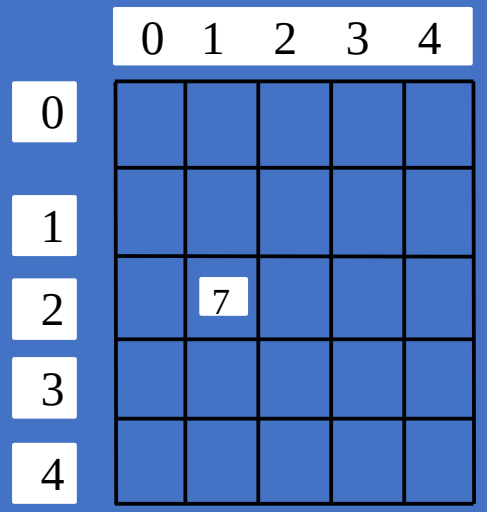
hoặc:

```
int matrix[][] = new int[10][10];

matrix[0][0] = 3;


for (int i=0; i<matrix.length; i++)
  for (int j=0; j<matrix[i].length; j++)
  {
    matrix[i][j] = (int)(Math.random()*1000);
  }


double[][] x;
```

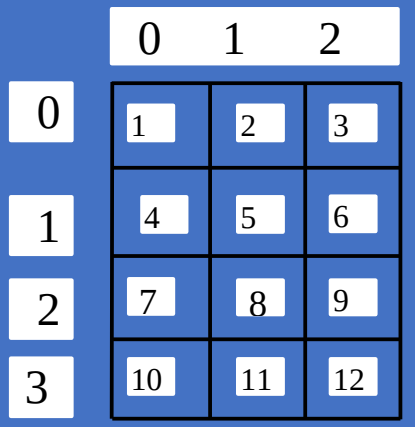
Minh họa mảng nhiều chiều




```
matrix = new int[5][5];
```




```
matrix[2][1] = 7;
```




```
int[][] array = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9},
    {10, 11, 12}
};
```

Khai báo, tạo, khởi tạo trong 1 lệnh

Ví dụ khai báo, tạo và khởi tạo một mảng 2 chiều:

```
int[][] array = {
  {1, 2, 3},
  {4, 5, 6},
  {7, 8, 9},
  {10, 11, 12}
};
```

Câu lệnh trên tương đương với các câu lệnh sau:

```
int[][] array = new int[4][3];
array[0][0] = 1;  array[0][1] = 2;  array[0][2] = 3;
array[1][0] = 4;  array[1][1] = 5;  array[1][2] = 6;
array[2][0] = 7;  array[2][1] = 8;  array[2][2] = 9;
array[3][0] = 10; array[3][1] = 11; array[3][2] = 12;
```

Độ dài của mảng nhiều chiều

```
int[][] array = {
   {1, 2, 3},
   {4, 5, 6},
   {7, 8, 9},
   {10, 11, 12}
};

array.length
array[0].length
array[1].length
array[2].length
```

Mảng gồ ghề

- Mỗi hàng của một mảng 2 chiều là một mảng đơn. Vì vậy, mỗi hàng có thể có độ dài khác nhau. Một mảng như vậy được gọi là *mảng gồ ghề (ragged array)*. Ví dụ:

```
int[][] matrix = {
   {1, 2, 3, 4, 5},
   {2, 3, 4, 5},
   {3, 4, 5},
   {4, 5},
   {5}
};
```

Ví dụ 7: Cộng và nhân 2 ma trận

- Mục tiêu: Sử dụng mảng 2 chiều để tạo 2 ma trận A, B rồi tính ma trận tổng A+B và ma trận tích A*B.

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{12} & b_{13} & b_{14} & b_{15} \\ b_{21} & b_{22} & b_{23} & b_{24} & b_{25} \\ b_{31} & b_{32} & b_{33} & b_{34} & b_{35} \\ b_{41} & b_{42} & b_{43} & b_{44} & b_{45} \\ b_{51} & b_{52} & b_{53} & b_{54} & b_{55} \end{pmatrix} = \begin{pmatrix} a_{11}+b_{11} & a_{12}+b_{12} & a_{13}+b_{13} & a_{14}+b_{14} & a_{15}+b_{15} \\ a_{21}+b_{21} & a_{22}+b_{22} & a_{23}+b_{23} & a_{24}+b_{24} & a_{25}+b_{25} \\ a_{31}+b_{31} & a_{32}+b_{32} & a_{33}+b_{33} & a_{34}+b_{34} & a_{35}+b_{35} \\ a_{41}+b_{41} & a_{42}+b_{42} & a_{43}+b_{43} & a_{44}+b_{44} & a_{45}+b_{45} \\ a_{51}+b_{51} & a_{52}+b_{52} & a_{53}+b_{53} & a_{54}+b_{54} & a_{55}+b_{55} \end{pmatrix}$$

Ví dụ 7: Cộng và nhân 2 ma trận (tiếp)

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{pmatrix} \times \begin{pmatrix} b_{11} & b_{12} & b_{13} & b_{14} & b_{15} \\ b_{21} & b_{22} & b_{23} & b_{24} & b_{25} \\ b_{31} & b_{32} & b_{33} & b_{34} & b_{35} \\ b_{41} & b_{42} & b_{43} & b_{44} & b_{45} \\ b_{51} & b_{52} & b_{53} & b_{54} & b_{55} \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} & c_{13} & c_{14} & c_{15} \\ c_{21} & c_{22} & c_{23} & c_{24} & c_{25} \\ c_{31} & c_{32} & c_{33} & c_{34} & c_{35} \\ c_{41} & c_{42} & c_{43} & c_{44} & c_{45} \\ c_{51} & c_{52} & c_{53} & c_{54} & c_{55} \end{pmatrix}$$

$$c_{ij} = a_{i1} \times b_{1j} + a_{i2} \times b_{2j} + a_{i3} \times b_{3j} + a_{i4} \times b_{4j} + a_{i5} \times b_{5j}$$

Ví dụ 7: Cộng và nhân 2 ma trận (tiếp)

```
361 public class CongMaTran {
362     public static void main(String[] args) {
363         // tao hai ma tran
364         int a[][] = { { 1, 3, 4 }, { 3, 4, 5 } };
365         int b[][] = { { 1, 3, 4 }, { 3, 4, 5 } };
366 
367         // tao ma tran khac de luu giu ket qua phep cong hai ma tran
368         int c[][] = new int[2][3];
369 
370         // cong va in tong hai ma tran
371         for (int i = 0; i < 2; i++) {
372             for (int j = 0; j < 3; j++) {
373                 c[i][j] = a[i][j] + b[i][j];
374                 System.out.print(c[i][j] + " ");
375             }
376             System.out.println();
377         }
378     }
379 }
```

Ví dụ 8: Chấm điểm Multiple-Choice Test

Students' Answers to the Questions:

	0	1	2	3	4	5	6	7	8	9
Student 0	A	B	A	C	C	D	E	E	A	D
Student 1	D	B	A	B	C	A	E	E	A	D
Student 2	E	D	D	A	C	B	E	E	A	D
Student 3	C	B	A	E	D	C	E	E	A	D
Student 4	A	B	D	C	C	D	E	E	A	D
Student 5	B	B	E	C	C	D	E	E	A	D
Student 6	B	B	A	C	C	D	E	E	A	D
Student 7	E	B	E	C	C	D	E	E	A	D

- Mục tiêu: viết chương trình tính điểm bài thi trắc nghiệm dạng multiple-choice.

Key to the Questions:

	0	1	2	3	4	5	6	7	8	9
Key	D	B	D	C	C	D	A	E	A	D

Ví dụ 9: Tính tổng điểm

- Mục tiêu: viết một chương trình tính tổng điểm cho sinh viên trong 1 lớp. Giả sử điểm được lưu trong một mảng 3 chiều có tên scores. Chiều thứ nhất ứng với một sinh viên, chiều thứ hai ứng với một kỳ thi, chiều thứ ba ứng với các điểm của kỳ thi. Giả sử có 7 sinh viên, 5 kỳ thi, và mỗi kỳ thi có 2 điểm: điểm trắc nghiệm và điểm lập trình. Do đó scores[i][j][0] biểu diễn điểm trắc nghiệm trong kỳ thi j của sinh viên i. Chương trình sẽ hiển thị tổng điểm của mỗi sinh viên.

Tìm kiếm trong mảng

- Tìm kiếm là quá trình tìm một phần tử xác định trong mảng, ví dụ tìm điểm nào đó trong danh sách điểm.
- Giống như sắp xếp, tìm kiếm là một tác vụ thường xuyên trong lập trình máy tính.
- Có nhiều giải thuật và cấu trúc dữ liệu để tìm kiếm. Chương này sẽ thảo luận 2 phương pháp phổ biến là: tìm kiếm tuyến tính (*linear search)* và tìm kiếm nhị phân (*binary search)*.

Tìm kiếm tuyến tính

- Phương pháp tìm kiếm tuyến tính so sánh phần tử khóa key với mỗi phần tử trong mảng list[].
- Việc tìm kiếm sẽ kết thúc khi tìm thấy một phần tử mảng bằng key hoặc khi duyệt hết mảng mà không tìm thấy.
- Nếu tìm thấy, tìm kiếm tuyến tính sẽ trả về chỉ số của phần tử bằng key.
- Nếu không tìm thấy, kết quả bằng -1.

Ví dụ 10: Tìm kiếm tuyến tính

- Mục tiêu: Tạo ngẫu nhiên 10 phần tử mảng kiểu `int` rồi hiển thị. Nhận key từ user rồi thực hiện tìm kiếm tuyến tính.

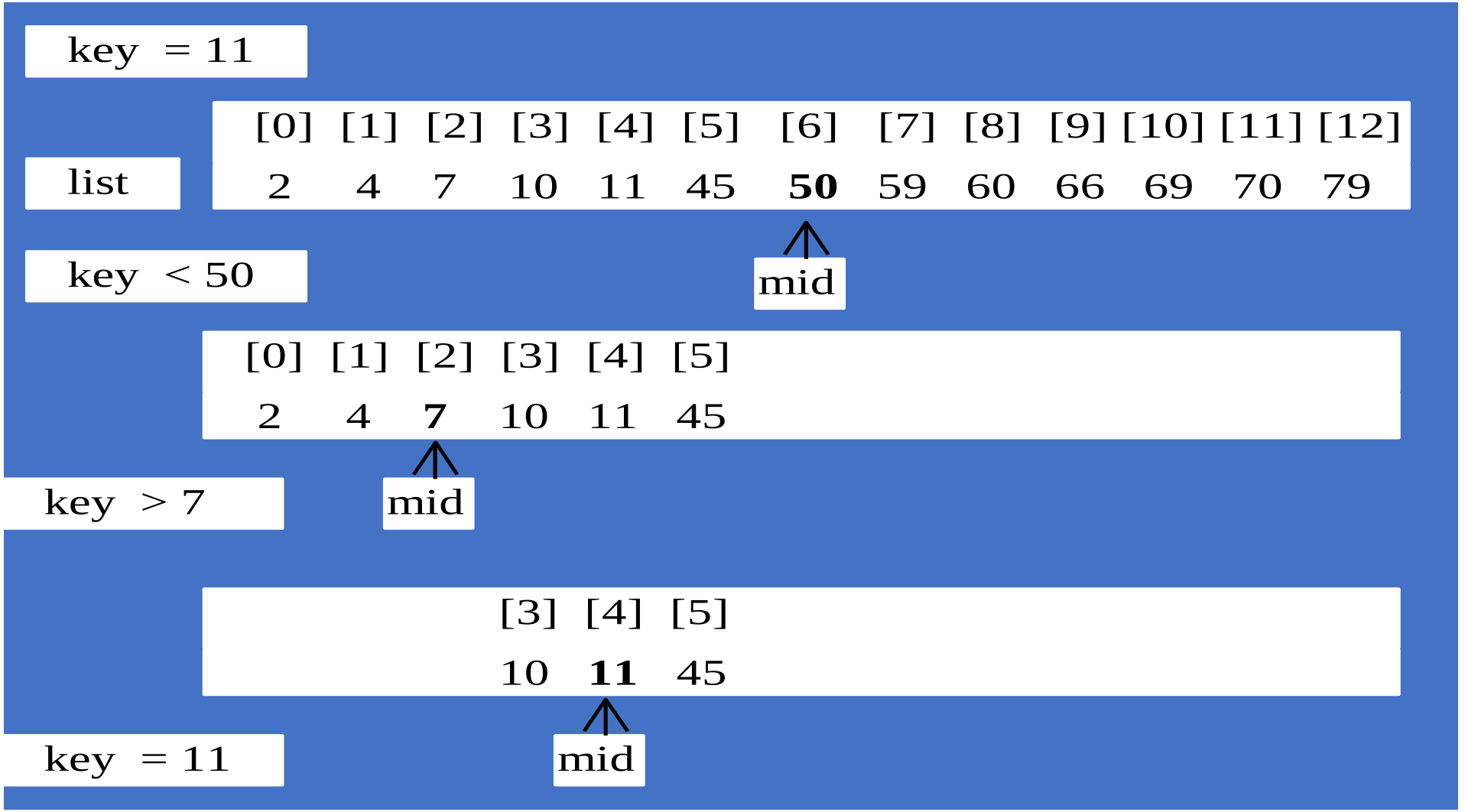
Tìm kiếm nhị phân - Binary Search

- Để thực hiện được tìm kiếm nhị phân, các phần tử mảng phải được sắp xếp theo thứ tự. Không làm mất tính tổng quát, giả sử mảng được sắp xếp tăng dần.
- vd: 2 4 7 10 11 45 50 59 60 66 69 70 79
- Trước tiên so sánh key với phần tử nằm giữa mảng. Có thể xảy ra 3 trường hợp sau:

Tìm kiếm nhị phân (tiếp)

- Nếu key bằng phần tử giữa, việc tìm kiếm kết thúc vì tìm thấy.
- Nếu key nhỏ hơn phần tử giữa, bạn chỉ cần tìm key trong nửa đầu của mảng theo phương pháp nhị phân .
- Nếu key lớn hơn phần tử giữa, bạn chỉ cần tìm key trong nửa cuối của mảng theo phương pháp nhị phân.

Binary Search, cont.



Ví dụ 11: Tìm kiếm nhị phân

- Mục tiêu: Tạo mảng 10 phần tử kiểu int rồi hiển thị. Nhận key từ user rồi thực hiện tìm kiếm nhị phân.

Ví dụ 12: Sắp xếp mảng

- Mục tiêu: Viết chương trình sử dụng phương pháp *Sắp xếp chọn* (`selectionSort`) để sắp xếp dãy số thực.

int[] myList = {2, 9, 5, 4, 8, 1, 6}; // chưa sắp xếp

Sắp xếp tăng dần: 1, 2, 4, 5, 6, 8, 9

Ví dụ 12: Sắp xếp chọn

int[] myList = {2, 9, 5, 4, 8, 1, 6}; // chưa sắp xếp

Tìm phần tử lớn nhất trong myList và đổi chỗ với phần tử cuối cùng của myList.

2, 9, 5, 4, 8, 1, 6 => 2, 6, 5, 4, 8, 1, 9 *(size = 7)*

2, 6, 5, 4, 8, 1 => 2, 6, 5, 4, 1, 8 *(size = 6)*

2, 6, 5, 4, 1 => 2, 1, 5, 4, 6 *(size = 5)*

2, 1, 5, 4 => 2, 1, 4, 5

2, 1, 4 => 2, 1, 4,

2, 1 => 1, 2

Kết quả: 1, 2, 4, 5, 6, 8, 9

Bài tập: Sắp xếp nổi bọt - Bubble Sort

```
int[] myList = {2, 9, 5, 4, 8, 1, 6};
```

Lần 1: 2, 5, 4, 8, 1, 6, 9
Lần 2: 2, 4, 5, 1, 6, 8, 9
Lần 3: 2, 4, 1, 5, 6, 8, 9
Lần 4: 2, 1, 4, 5, 6, 8, 9
Lần 5: 1, 2, 4, 5, 6, 8, 9
Lần 6: 1, 2, 4, 5, 6, 8, 9

Q&A

BKACAD®

HỌC VIỆN CÔNG NGHỆ BKACAD

THANK YOU