Chapter 6 Arrays

ประโยชน์ของ Array

บ่อยครั้งที่เราอยากจะเก็บข้อมูล (ชนิดเดียวกัน) จำนวนมาก ๆ

- เช่น อ่านค่าจำนวนทศนิยมมา 100 ค่า แล้วหาค่าเฉลี่ย แล้ว ดูว่า มีกี่ค่าที่มีค่ามากกว่าค่าเฉลี่ยนั้น
- ค่าต่าง ๆ จะต้องถูกเก็บไว้ก่อน เพื่อที่เราจะได้นำมาใช้หาค่าเฉลี่ยทีหลัง
- ถ้าเราเกิดประกาศตัวแปรมา 100 ตัวเพื่อรับค่า แล้วมาหาค่าเฉลี่ย code จะยาวมาก และทำไม่ได้ทางปฏิบัติ

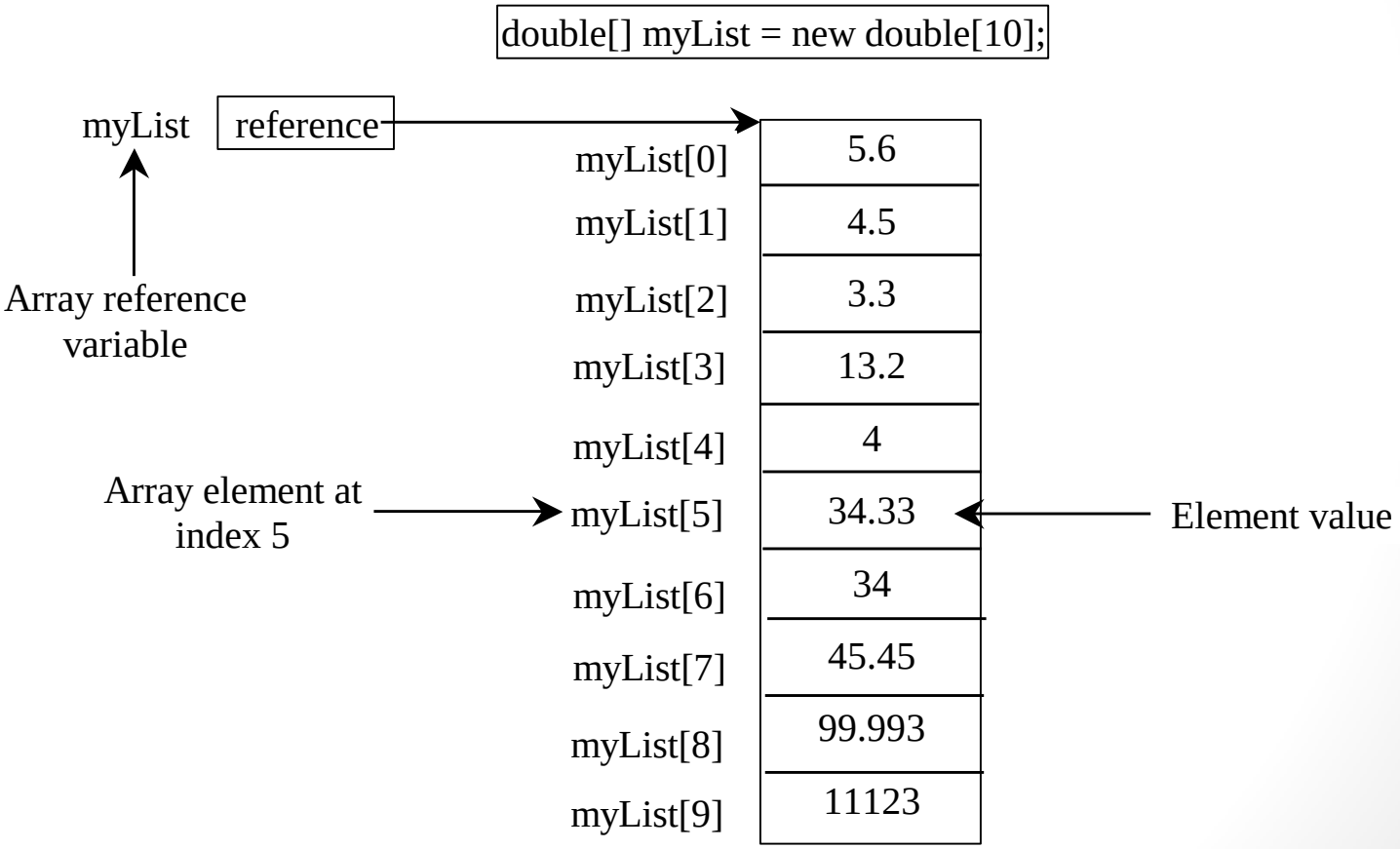
  → Array ช่วยแก้ปัญหานี้ได้

วัตถุประสงค์

- รู้ว่า array สำคัญอย่างไร
- รู้ว่าใช้ array อย่างไร: การประกาศ array, การอ้างถึงตัวแปรใน array, การตั้งค่าเริ่มต้นให้สมาชิกใน array
- การเข้าถึงสมาชิกใน array โดยใช้ตัวแปร
- รู้จักการใช้ array initializer
- สามารถคัดลอกข้อมูลจาก array หนึ่งไปยัง array หนึ่งได้
- ส่ง array ไปให้ method ได้
- ใช้ array หลายมิติในการแก้ปัญหา

Introducing Arrays

Array คือ โครงสร้างข้อมูลชนิดหนึ่ง ที่ใช้เก็บข้อมูลชนิดเดียวกัน



การประกาศตัวแปร Array

- `datatype[] arrayRefVar;`

  Example:

  ```
  double[] myList;
  ```

- `datatype arrayRefVar[];` // This style is allowed, but not preferred

  Example:

  ```
  double myList[];
  ```

การสร้าง Arrays

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

Example:

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

`myList[0]` อ้างไปยังสมาชิกแรกใน array myList

`myList[9]` อ้างไปยังสมาชิกสุดท้ายใน array myList

ประกาศตัวแปร array และ สร้างใน step เดียวกัน

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

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

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

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

ขนาดของ Array

ตอนที่ array ถูกสร้างขึ้นมา มันจะมีขนาดคงที่ และเปลี่ยนไม่ได้ เราจะหาขนาดของ array โดยใช้ field (ตัวแปรของ Array) ข้างล่าง

```
arrayRefVar.length
```

For example,

```
myList.length returns 10
```

ค่า Default (ค่าโดยปริยาย)

ตอนที่ array ถูกสร้างขึ้นมา สมาชิกทุกตัวใน array จะมีค่า default ดังนี้:-

0 for the numeric primitive data types,

'\u0000' สำหรับชนิด char, and

false สำหรับชนิด boolean

ตัวแปรดัชนี (Indexed Variable)

สมาชิกใน array จะถูกเข้าถึง (ถูกอ่าน หรือ ถูกกำหนดค่า) ได้โดยการใช้ดัชนีในการอ้างถึง ดัชนีจะเริ่มจาก 0 ถึง arrayRefVar.length-1

เราจะใช้ syntax ด้านล่างแทนสมาชิกแต่ละตัวใน array

```
arrayRefVar[index];
```

ซึ่งเราจะเรียกตัวแปรที่อยู่ในลักษณะนี้ว่า Indexed Variable

การใช้ Indexed Variables

หลังจากที่เราสร้าง array แล้ว, เราสามารถใช้ indexed variable แบบเดียวกับที่เราใช้ตัวแปรแบบธรรมดา เช่น

```
myList[2] = myList[0] + myList[1];
```

การกำหนดค่าตั้งต้นให้ Array

- ประกาศ, สร้าง, ตั้งค่าตั้งต้น ในขั้นตอนเดียว:

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

การประกาศ, สร้าง, ตั้งค่าตั้งต้นให้ array ในรูปแบบย่อ

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

เหมือนกับ

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

คำเตือน

การประกาศ, สร้าง, ตั้งค่าตั้งต้นให้ array ในรูปแบบย่อต้องทำในหนึ่ง statement

ห้ามทำแยกแบบข้างล่าง

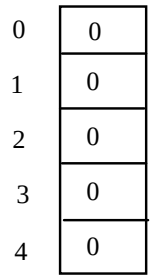
```
double[] myList;

myList = {1.9, 2.9, 3.4, 3.5};
```

Trace Program with Arrays

ประกาศ ตัวแปร variable ชื่อ values, สร้าง array, และ ตั้งค่าการอ้างอิง (ค่า reference) ให้ values

```
public class Test {
  public static void main(Stri[] args) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the array is created



Trace Program with Arrays

i เป็น 1

```
public class Test {
  public static void main(String[] args) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the array is created

0	0
1	0
2	0
3	0
4	0

Trace Program with Arrays

i (=1) is less than 5

```
public class Test {
  public static void main(String[] args) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the array is created

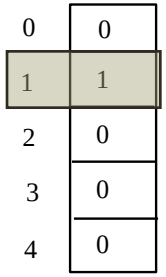
0	0
1	0
2	0
3	0
4	0

Trace Program with Arrays

After this line is executed, value[1] is 1

```
public class Test {
  public static void main(Strin      gs) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the first iteration



Trace Program with Arrays

After i++, i becomes 2

```
public class Test {
  public static void main(String[] args) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the first iteration

0	0
1	1
2	0
3	0
4	0

Trace Program with Arrays

```
public class Test {
  public static void main(String[] ar
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

i (= 2) is less than 5

After the first iteration

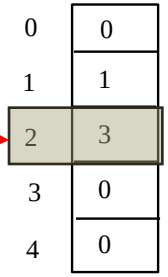
0	0
1	1
2	0
3	0
4	0

Trace Program with Arrays

After this line is executed,
values[2] is 3 (2 + 1)

```
public class Test {
  public static void main(Strin      gs) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the second iteration



Trace Program with Arrays

After this, i becomes 3.

```
public class Test {
  public static void main(String[] args) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the second iteration

0	0
1	1
2	3
3	0
4	0

Trace Program with Arrays

i (=3) is still less than 5.

```
public class Test {
  public static void main(String[] args) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the second iteration

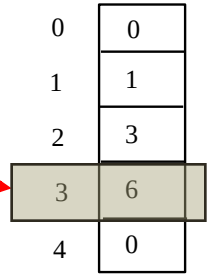
0	0
1	1
2	3
3	0
4	0

Trace Program with Arrays

After this line, values[3] becomes 6 (3 + 3)

```
public class Test {
  public static void main(String[] args)
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the third iteration



Trace Program with Arrays

After this, i becomes 4

```
public class Test {
  public static void main(String[] args) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the third iteration

0	0
1	1
2	3
3	6
4	0

Trace Program with Arrays

i (=4) is still less than 5

```
public class Test {
  public static void main(String[] args) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the third iteration

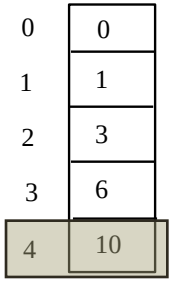
0	0
1	1
2	3
3	6
4	0

Trace Program with Arrays

After this, values[4] becomes 10 (4 + 6)

```
public class Test {
  public static void main(Strir  rgs) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the fourth iteration



Trace Program with Arrays

After i++, i becomes 5

```
public class Test {
  public static void main(String[] args)
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

After the fourth iteration

0	0
1	1
2	3
3	6
4	10

Trace Program with Arrays

i (=5) < 5 is false. Exit the loop

```
public class Test {
  public static void main(String[] args) {
    int[] values = new int[5];
    for (int i = 1; i < 5; i++) {
      values[i] = i + values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```

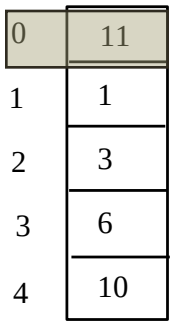
After the fourth iteration

0	0
1	1
2	3
3	6
4	10

Trace Program with Arrays

After this line, values[0] is 11 (1 + 10)

```
public class Test {
  public static void ma      ing[] args) {
    int[] values = nev      [5];
    for (int i = 1; i <      i++) {
      values[i] = i      values[i-1];
    }
    values[0] = values[1] + values[4];
  }
}
```



0	11
1	1
2	3
3	6
4	10

Processing Arrays

See the examples in the text.

1. (Initializing arrays)
2. (Printing arrays)
3. (Summing all elements)
4. (Finding the largest element)
5. (Finding the smallest index of the largest element)
6. (*Shifting elements*)

Enhanced for Loop (for-each loop)

JDK 1.5 มี loop ชนิดใหม่ ที่ช่วยให้เรา traverse (ท่อง) ไปทั้ง array ตามลำดับได้โดยไม่ต้องใช้ตัวแปรดัชนี ตัวอย่างเช่น code ข้างล่างจะ print ค่าในสมาชิกแต่ละตัวออกมา

```
for (double value: myList)
  System.out.println(value);
```

โดยทั่ว ๆ ไป syntax คือ

```
for (elementType value: arrayRefVar) {
  // Process the value
}
```

เรายังคงต้องใช้ ตัวแปร index ถ้าเราต้องการเข้าถึง array แต่ไม่เป็นไปตามลำดับหรือเราต้องการไปเปลี่ยนค่าใน array

ตัวอย่าง: วิเคราะห์สมาชิกใน array

- รับค่า 6 ค่าจากผู้ใช้ แล้วหาค่าที่มากที่สุด และ นับว่า มีตัวเลขที่มีค่ามากที่สุดถูกป้อนเข้ามากี่ตัวเช่น ถ้าผู้ใช้ป้อน 3, 5, 2, 5, 5, and 5, ค่าที่มากที่สุดคือ 5 และมี 5 อยู่ 4 ตัว

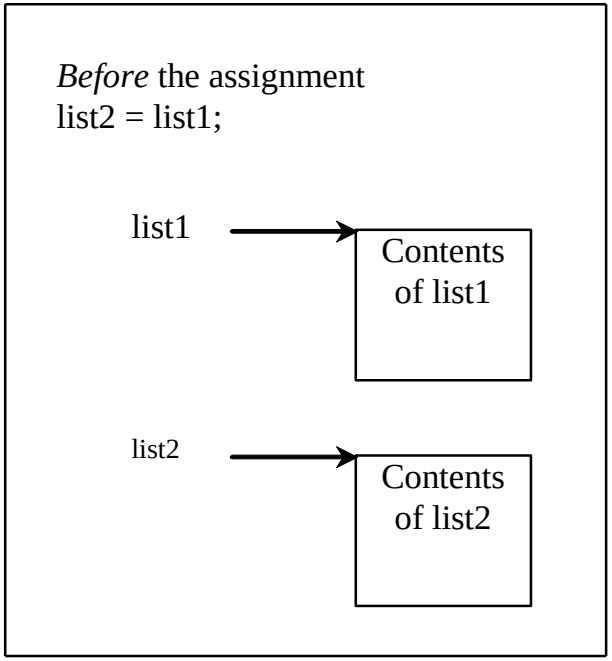
ตัวอย่าง: การให้ Grades นิสิต

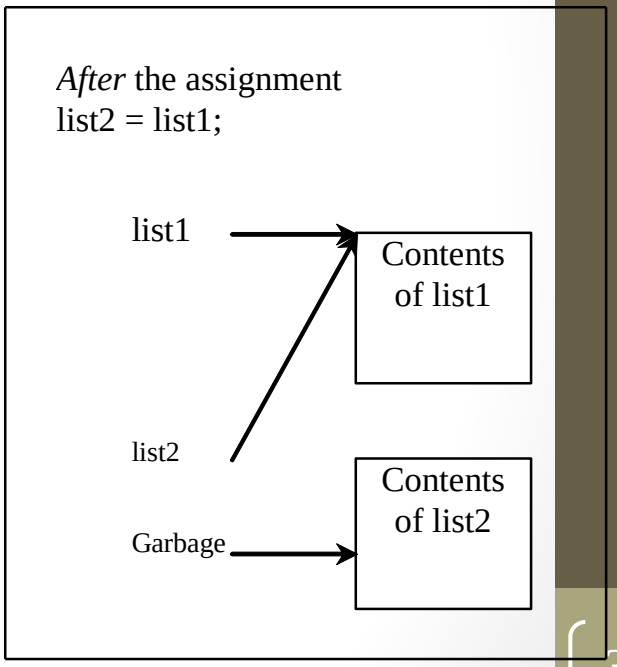
- Objective: อ่านคะแนนของนิสิต (int), หาคะแนนที่ดีที่สุด, and then แล้วให้เกรดตามรายละเอียดดังนี้:
  - Grade is A if score is >= best–10;
  - Grade is B if score is >= best–20;
  - Grade is C if score is >= best–30;
  - Grade is D if score is >= best–40;
  - Grade is F otherwise.

การคัดลอก Arrays

บ่อยครั้ง เราต้องการคัดลอกข้อมูลใน array ทั้งหมดหรือบางส่วนของมัน แต่ถ้าเราใช้ statement แบบข้างล่าง การคัดลอกจะไม่ได้เกิดขึ้นจริง ๆ

```
list2 = list1;
```





การคัดลอก Arrays

Using a loop:

```
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];
```

The `arraycopy` Utility

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

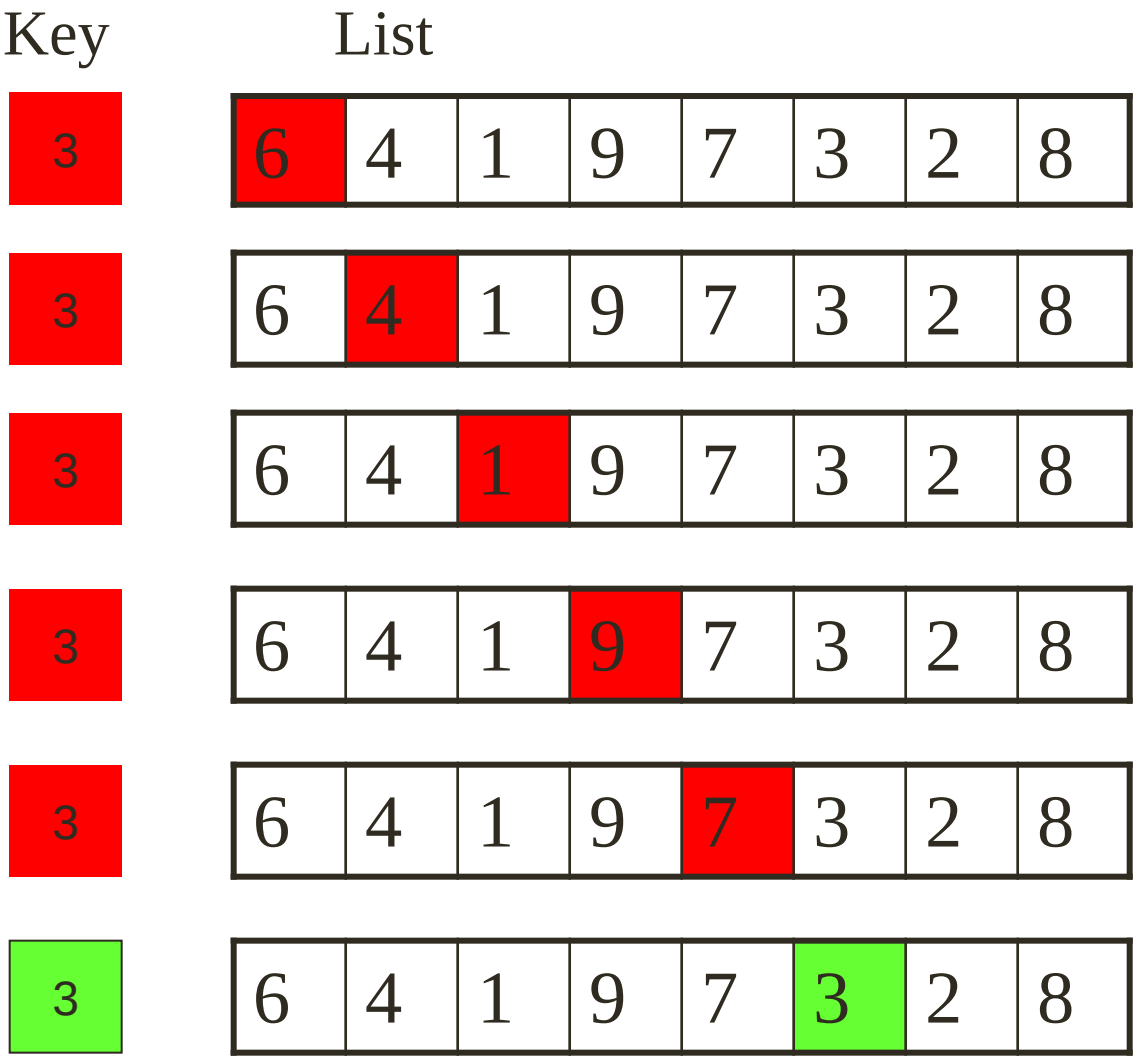
Example:

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

Linear Search

The linear search approach compares the key element, key, *sequentially* with each element in the array list. The method continues to do so until the key matches an element in the list or the list is exhausted without a match being found. If a match is made, the linear search returns the index of the element in the array that matches the key. If no match is found, the search returns -1.

Linear Search Animation



From Idea to Solution

```
/** The method for finding a key in the list */
public static int linearSearch(int[] list, int key) {
  for (int i = 0; i < list.length; i++)
    if (key == list[i])
      return i;
  return -1;
}
```

Trace the method

```
int[] list = {1, 4, 4, 2, 5, -3, 6, 2};
int i = linearSearch(list, 4);  // returns 1
int j = linearSearch(list, -4); // returns -1
int k = linearSearch(list, -3); // returns 5
```

Binary Search

สมาชิกใน array ต้องเรียงลำดับอยู่แล้ว (มากไปน้อย หรือ น้อยไปมากก็ได้)

เช่น 2 4 7 10 11 45 50 59 60 66 69 70 79

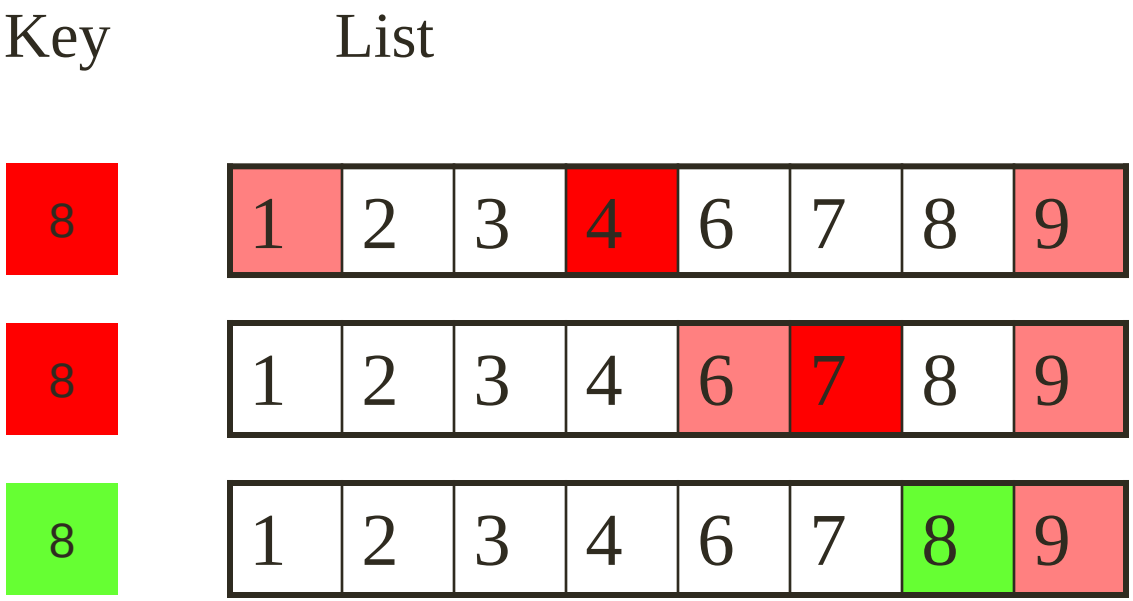
เริ่มต้นที่เปรียบเทียบกับสมาชิกตัวที่อยู่ตรงกลางก่อน

Binary Search, cont.

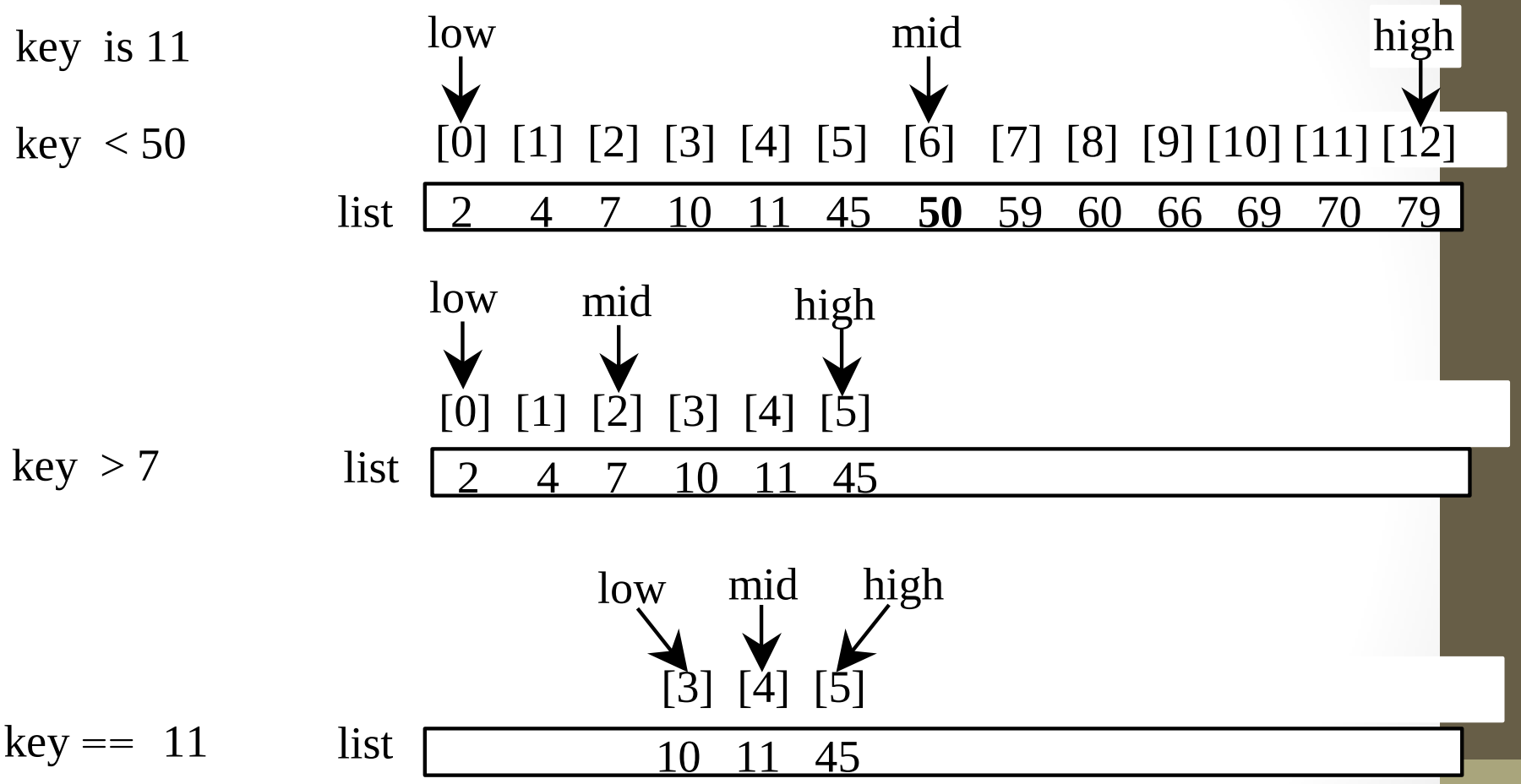
พิจารณา 3 กรณี:

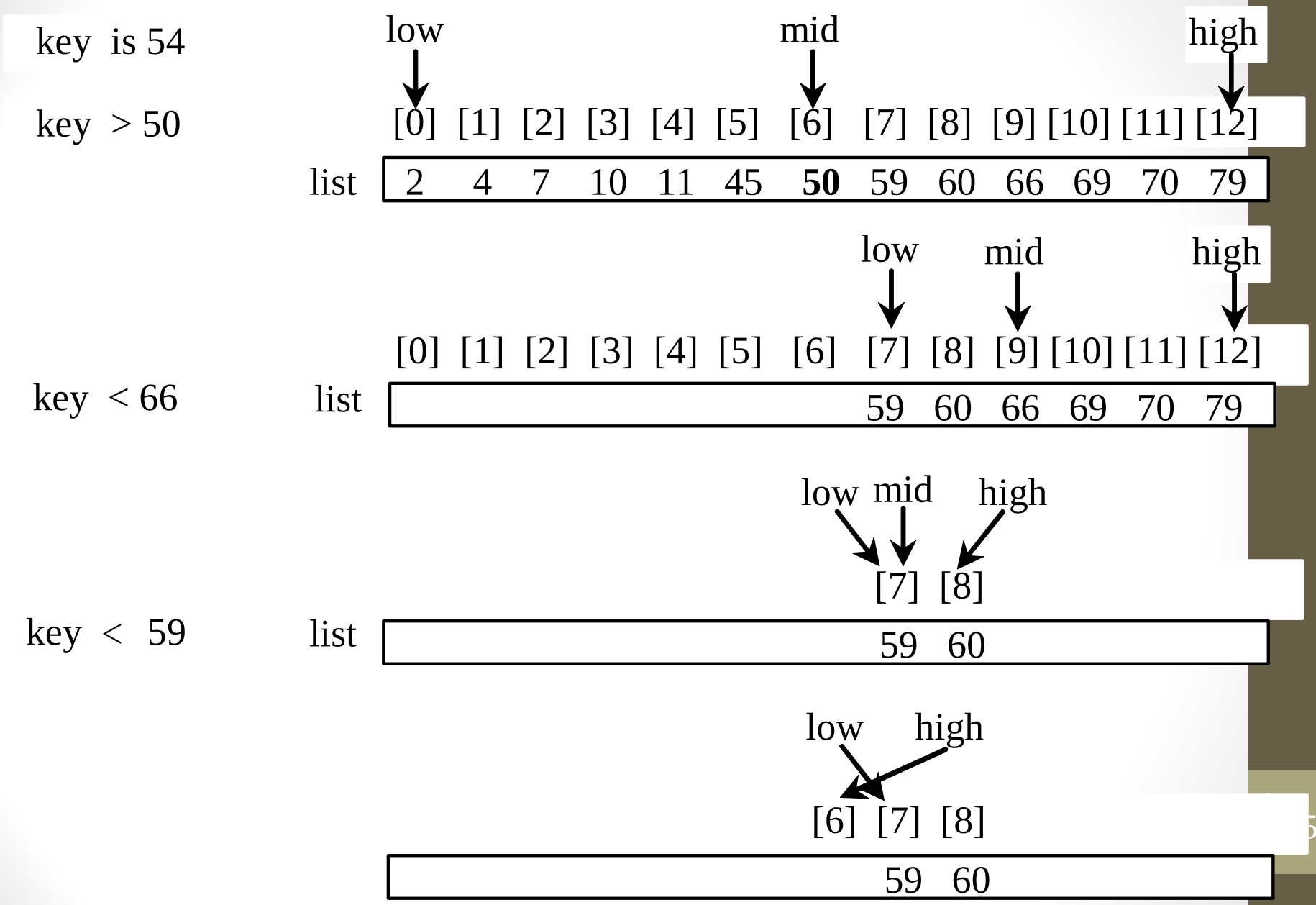
- key < middle element, ค้นหาในครึ่งแรกของ array
- key == middle element, หยุดหาต่อเพราะเจอแล้ว
- Key > middle element , ค้นหาในครึ่งหลังของ array

Binary Search



Binary Search, cont.



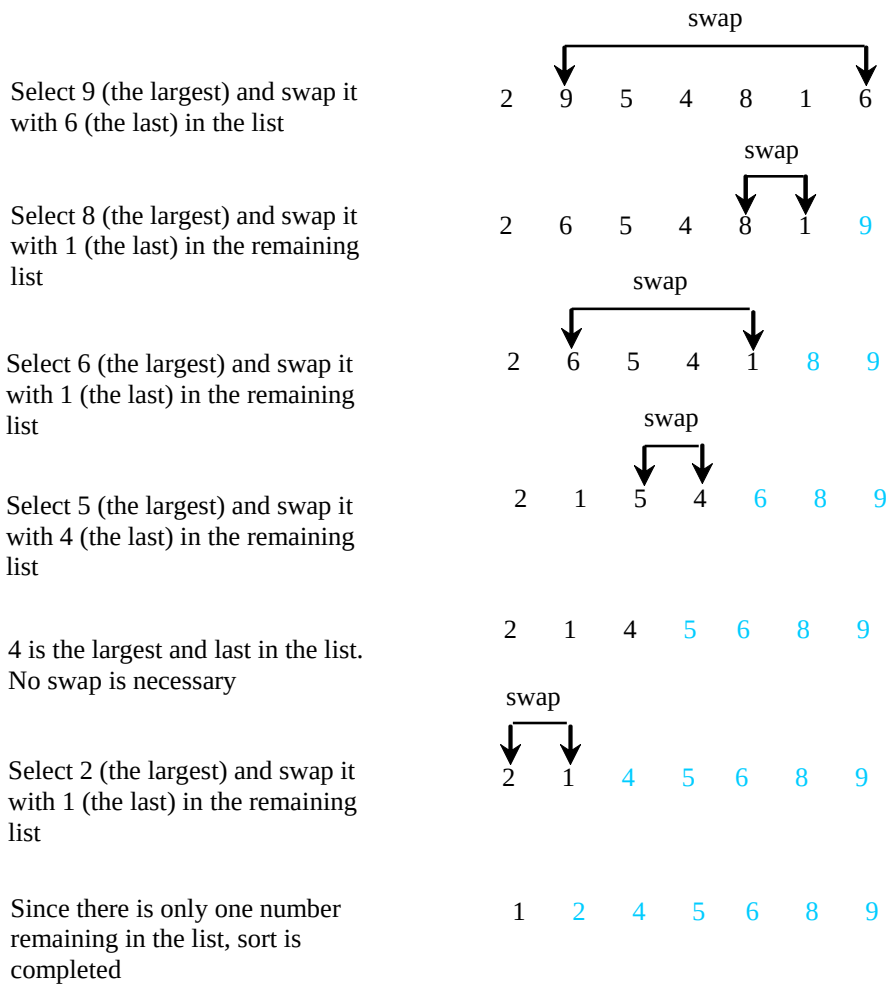

key is 54
low
mid
high
key > 50
[0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12]
list
2 4 7 10 11 45 50 59 60 66 69 70 79
low
mid
high
[0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12]
key < 66
list
59 60 66 69 70 79
low mid high
[7] [8]
key < 59
list
59 60
low high
[6] [7] [8]
59 60

Sorting Arrays

Sorting, like searching, is also a common task in computer programming. It would be used, for instance, if you wanted to display the grades from Listing 6.2, “Assigning Grades,” in alphabetical order. Many different algorithms have been developed for sorting. This section introduces two simple, intuitive sorting algorithms: *selection sort* and *insertion sort*.

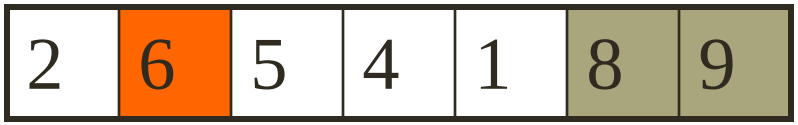
Selection Sort

Selection sort finds the largest number in the list and places it last. It then finds the largest number remaining and places it next to last, and so on until the list contains only a single number. Figure 6.17 shows how to sort the list {2, 9, 5, 4, 8, 1, 6} using selection sort.



Selection Sort

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



Insertion Sort

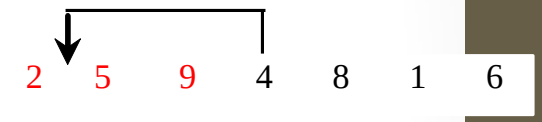
The insertion sort algorithm sorts a list of values by repeatedly inserting an unsorted element into a sorted sublist until the whole list is sorted.

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

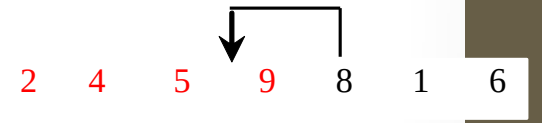
Step 1: Initially, the sorted sublist contains the first element in the list. Insert 9 to the sublist.

Step2: The sorted sublist is {2, 9}. Insert 5 to the sublist.

Step 3: The sorted sublist is {2, 5, 9}. Insert 4 to the sublist.

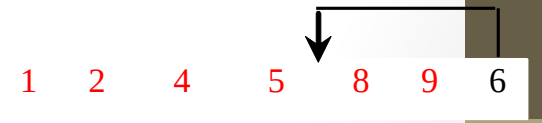


Step 4: The sorted sublist is {2, 4, 5, 9}. Insert 8 to the sublist.



Step 5: The sorted sublist is {2, 4, 5, 8, 9}. Insert 1 to the sublist.

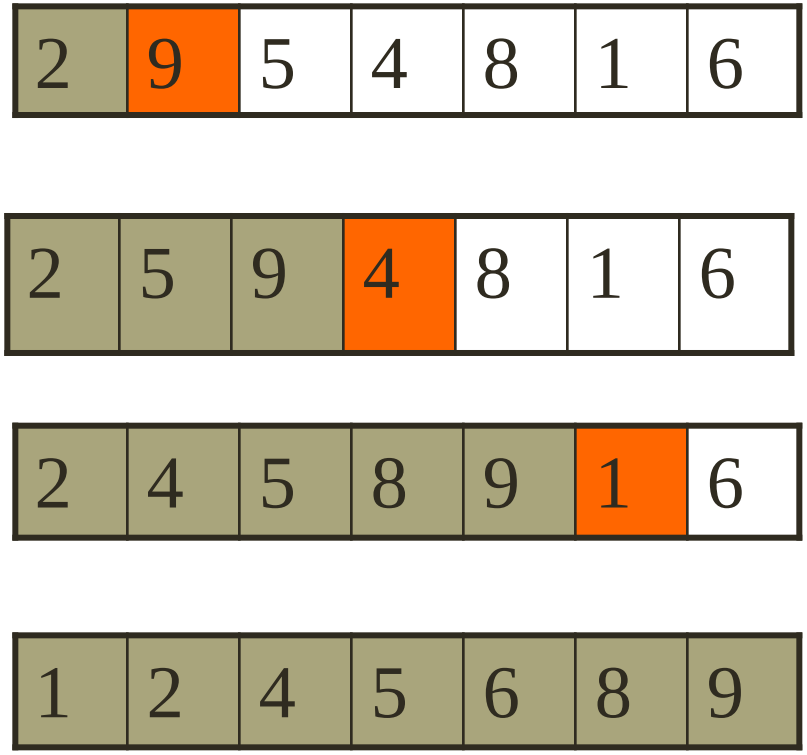
Step 6: The sorted sublist is {1, 2, 4, 5, 8, 9}. Insert 6 to the sublist.

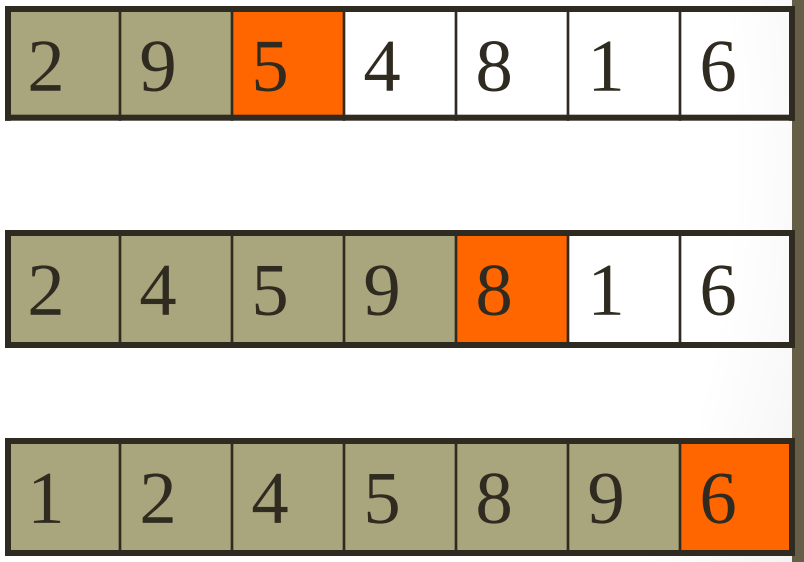


Step 7: The entire list is now sorted

Insertion Sort

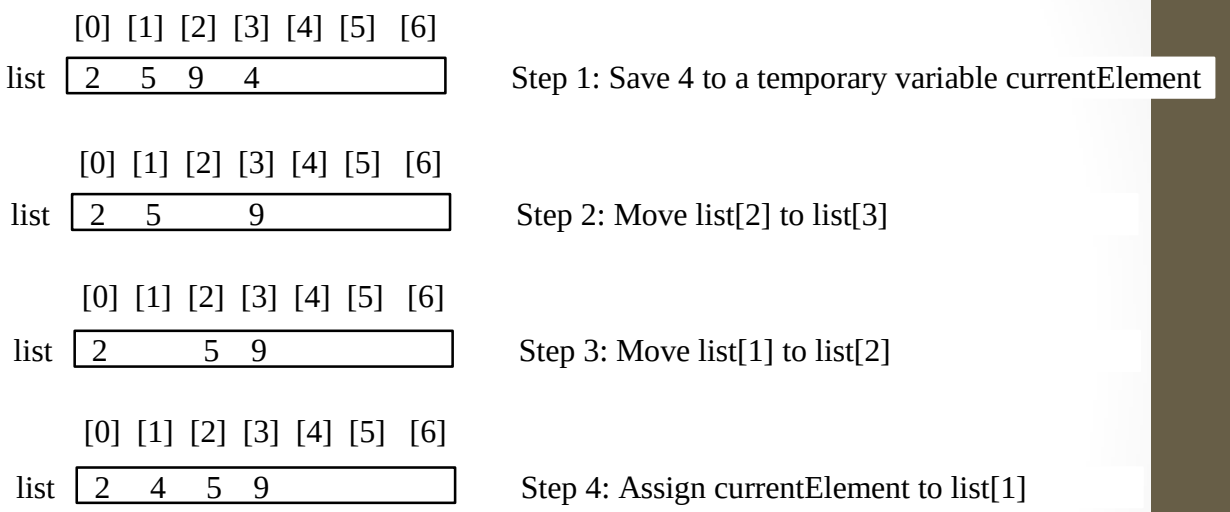
int[] myList = {2, 9, 5, 4, 8, 1, 6}; // Unsorted





How to Insert?

The insertion sort algorithm sorts a list of values by repeatedly inserting an unsorted element into a sorted sublist until the whole list is sorted.



From Idea to Solution

```
for (int i = 1; 1; i < list,length; i++) {
  insert list[i] into a sorted sublist list[0..i-1] so that
  list[0..i] is sorted
}
```

```
list[0]
```

```
list[0] list[1]
```

```
list[0] list[1] list[2]
```

```
list[0] list[1] list[2] list[3]
```

```
list[0] list[1] list[2] list[3] ...
```

The Arrays.sort Method

Since sorting is frequently used in programming, Java provides several overloaded sort methods for sorting an array of int, double, char, short, long, and float in the java.util.Arrays class. For example, the following code sorts an array of numbers and an array of characters.

```
double[] numbers = {6.0, 4.4, 1.9, 2.9, 3.4, 3.5};
java.util.Arrays.sort(numbers);

char[] chars = {'a', 'A', '4', 'F', 'D', 'P'};
java.util.Arrays.sort(chars);
```

Arrays 2 มิติ

บางครั้งเราอยากมีโครงสร้างเก็บข้อมูล 2 มิติ เช่น ตารางด้านล่างเก็บระยะทางจากเมืองหนึ่งไปยังอีกเมืองหนึ่ง

Distance Table (in miles)

	Chicago	Boston	New York	Atlanta	Miami	Dallas	Houston
Chicago	0	983	787	714	1375	967	1087
Boston	983	0	214	1102	1763	1723	1842
New York	787	214	0	888	1549	1548	1627
Atlanta	714	1102	888	0	661	781	810
Miami	1375	1763	1549	661	0	1426	1187
Dallas	967	1723	1548	781	1426	0	239
Houston	1087	1842	1627	810	1187	239	0

Declare/Create Two-dimensional Array

```
// Declare array ref var
dataType[][] refVar;

// Create array and assign its reference to variable
refVar = new dataType[10][10];

// Combine declaration and creation in one statement
dataType[][] refVar = new dataType[10][10];

// Alternative syntax
dataType refVar[][] = new dataType[10][10];
```

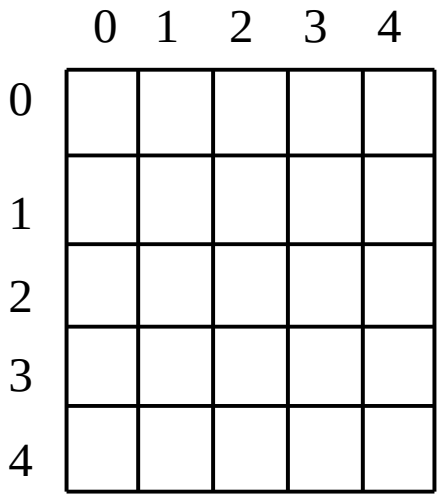
Declaring Variables of Two-dimensional Arrays and Creating Two-dimensional Arrays

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

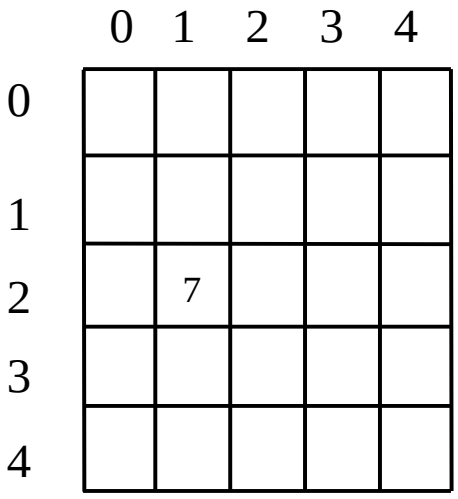
or

```
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;
```

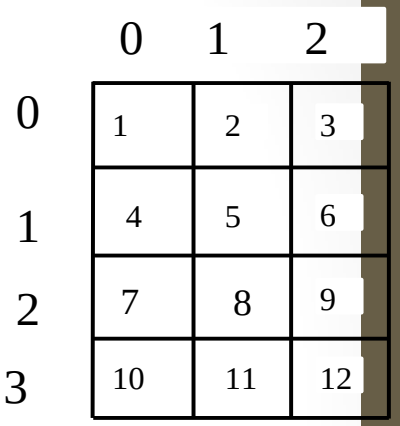
Two-dimensional Array Illustration




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




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




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

matrix.length? 5

matrix[0].length? 5

array.length? 4

array[0].length? 3

Declaring, Creating, and Initializing Using Shorthan Notations

You can also use an array initializer to declare, create and initialize a two-dimensional array. For example,

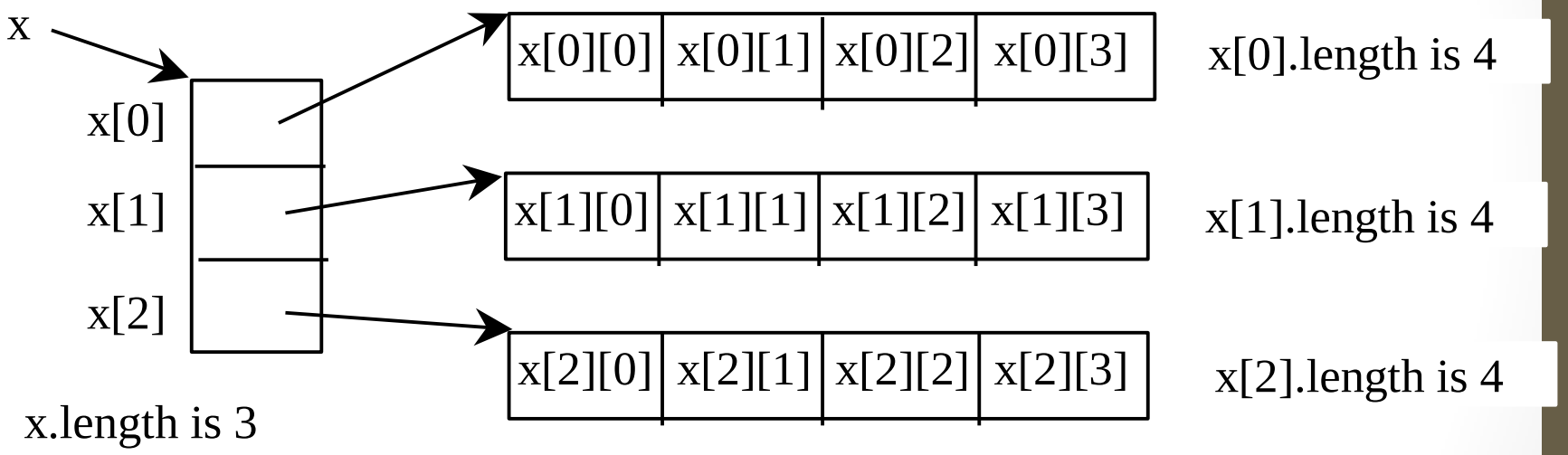
```
int[][] array = {
  {1, 2, 3},
  {4, 5, 6},
  {7, 8, 9},
  {10, 11, 12}
};
```

Same as

```
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;
```

Lengths of Two-dimensional Arrays

```
int[][] x = new int[3][4];
```



Lengths of Two-dimensional Arrays, cont.

```
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

array[3].length

array[4].length ArrayIndexOutOfBoundsException

Ragged Arrays

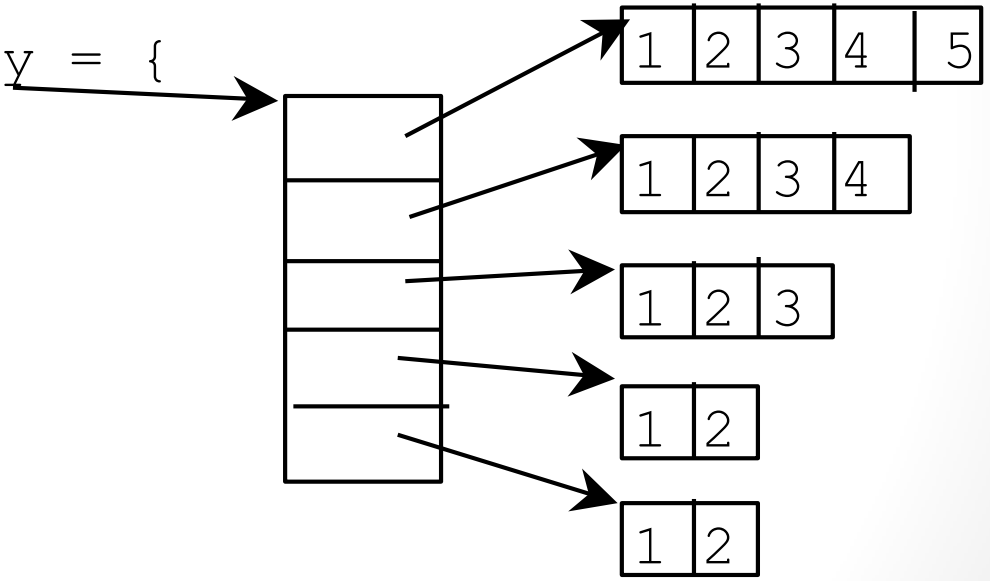
แต่ละแถวของ array ก็เป็น array ด้วย ซึ่งขนาดของแต่ละแถว
อาจไม่เหมือนกันก็ได้ (*ragged array*)

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

matrix.length is 5
matrix[0].length is 5
matrix[1].length is 4
matrix[2].length is 3
matrix[3].length is 2
matrix[4].length is 1

Ragged Arrays, cont.

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



Problem: Grading Multiple-Choice Test

Students' Answers to the Questions:

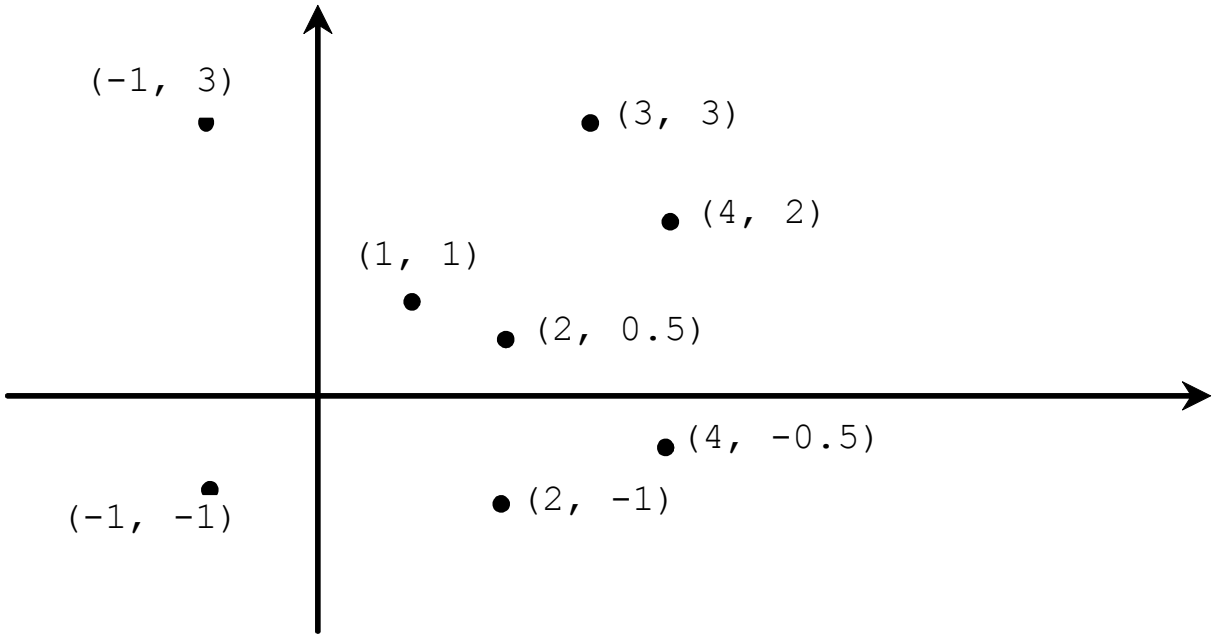
	0	1	2	3	4	5	6	7	8	9
Student 0	A	B	A	C	C	D	E	E	A	D
Student 1	D	B	A	B	C	A	E	E	A	D
Student 2	E	D	D	A	C	B	E	E	A	D
Student 3	C	B	A	E	D	C	E	E	A	D
Student 4	A	B	D	C	C	D	E	E	A	D
Student 5	B	B	E	C	C	D	E	E	A	D
Student 6	B	B	A	C	C	D	E	E	A	D
Student 7	E	B	E	C	C	D	E	E	A	D

- Objective: write a program that grades multiple-choice test.

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

Problem: Finding Two Points Nearest to Each Other



	x	y
0	-1	3
1	-1	-1
2	1	1
3	2	0.5
4	2	-1
5	3	3
6	4	2
7	4	-0.5

Arrays n มิติ

เช่น ถ้าเราจะประกาศ array 3 มิติสามารถทำได้ดังนี้.

```
double[][][] scores = new double[10][5][2];
```

ดัชนีแรกใน scores อ้างถึงนักเรียน

ดัชนีที่สองอ้างถึงการสอบครั้งที่

ดัชนีที่สามอ้างถึงคะแนนสอบ

Refine the table

Tax rate	Single filers	Married filing jointly or qualifying widow/widower	Married filing separately	Head of household
10%	Up to $6,000	Up to $12,000	Up to $6,000	Up to $10,000
15%	$6,001 - $27,950	$12,001 - $46,700	$6,001 - $23,350	$10,001 - $37,450
27%	$27,951 - $67,700	$46,701 - $112,850	$23,351 - $56,425	$37,451 - $96,700
30%	$67,701 - $141,250	$112,851 - $171,950	$56,426 - $85,975	$96,701 - $156,600
35%	$141,251- $307,050	$171,951 - $307,050	$85,976 - $153,525	$156,601 - $307,050
38.6%	$307,051 or more	$307,051 or more	$153,526 or more	$307,051 or more

10%	6000	12000	6000	10000
15%	27950	46700	23350	37450
27%	67700	112850	56425	96745
30%	141250	171950	85975	156600
35%	307050	307050	153525	307050
38.6%				

Reorganize the table

6000	12000	6000	10000
27950	46700	23350	37450
67700	112850	56425	96745
141250	171950	85975	156600
307050	307050	153525	307050

Rotate

6000	27950	67700	141250	307050
12000	46700	112850	171950	307050
6000	23350	56425	85975	153525
10000	37450	96745	156600	307050

Single filer

Married jointly

Married separately

Head of household

Declare Two Arrays

6000	27950	67700	141250	307050
12000	46700	112850	171950	307050
6000	23350	56425	85975	153525
10000	37450	96745	156600	307050

Single filer

Married jointly

Married separately

Head of household

10%
15%
27%
30%
35%
38.6%

```
int[][] brackets = {
  {6000, 27950, 67700, 141250, 307050}, // Single filer
  {12000, 46700, 112850, 171950, 307050}, // Married jointly
  {6000, 23350, 56425, 85975, 153525}, // Married separately
  {10000, 37450, 96700, 156600, 307050} // Head of household
};
```

```
double[] rates = {0.10, 0.15, 0.27, 0.30, 0.35, 0.386};
```