Arrays and Basic Algorithms

Roman Kontchakov

Birkbeck, University of London
Outline

- Arrays
- Common Array Algorithms
- Enhanced for Loop
- Using Arrays with Methods
  - Sections 6.1–6.4
- slides are available at www.dcs.bbk.ac.uk/~roman/sp1
Arrays

- **Array** collects a sequence of values of the same type.

1 // empty array of 5 integers
2 int[] data = new int[5];
3 // list of initial values
4 double[] data2 = { 32, 54, 67.5, 29 };
Array Elements

- individual elements in an array are accessed by an integer index $i$, using the notation $\text{data}[i]$
- an array element can be used in expressions like any other variable
- the elements of arrays are numbered starting at 0
- use the expression $\text{data.length}$ to find the number of elements in an array

1. int minidx = 0;
2. for (int i = 0; i < data.length; i++)
3.     if (data[i] < data[minidx])
4.         minidx = i;
Arrays: References are Copied

an **array variable** specifies the location of an array
**copying** the reference yields another reference to the **same** array

```java
1 int[] scores = { 10, 9, 7, 4, 5};
2 int[] values = scores;
3 scores[3] = 10;
4 // prints 10, not 4 !
5 System.out.println(values[3]);
```

**NB:** just like object references!
Arrays: Common Errors

- array data has elements with index values 0 to data.length - 1
  run-time exception: java.lang.ArrayIndexOutOfBoundsException

- arrays must be initialised (otherwise the reference is null)
  run-time exception: java.lang.NullPointerException
double total = 0;
for (int i = 0; i < data.length; i++) {
    total += data[i];
}
double average = total / data.length;

what if data.length == 0?

double average = 0;
if (data.length > 0)
    average = total / data.length;
for (int i = 0; i < data.length; i++) {
    // print comma unless at the first element
    if (i > 0)
    System.out.print(",");
    System.out.print(data[i]);
}

Exercise: modify the code so that the comma is printed after the current element (i.e., not before as in the code above)
int searchedValue = 100;
int pos = 0;
boolean found = false;
while (pos < data.length && !found) {
    if (data[pos] == searchedValue)
        found = true;
    else
        pos++;
}
if (found)
    System.out.println("Found at position: " + pos);
else
    System.out.println("Not found");

NB: not optimal if the array is sorted
Arrays and Methods

Arrays can occur as method parameters...

```java
public static double sum(double[] data) {
    double total = 0;
    for (int i = 0; i < data.length; i++) {
        total += data[i];
    }
    return total;
}
```

**NB:** brackets can also be placed after the array’s name
```
double data[]
```

This, however, is not recommended — the brackets identify the array type and should appear with the type designation.
use the enhanced `for` loop to visit all elements of an array

```java
public static double sum(double[] data) {
    double total = 0;
    // for(int i = 0; i < data.length; i++)
    for (double e: data) {
        // total += data[i];
        total += e; // e stands for data[i]
    }
    return total;
}
```
Arrays and Methods (2)

... and return values

```java
public static int[] squares(int n) {
    int[] result = new int[n];
    for (int i = 0; i < result.length; i++)
        result[i] = i * i;
    return result;
}
```

**NB.** the enhanced for loop cannot be used here

```java
e = i * i;  // will not modify the array element
```
arrays (and objects) are passed “by reference”

```java
public static void multiply(double[] data, double factor) {
    for (int i = 0; i < data.length; i++) {
        // changes the contents of data
        data[i] *= factor;
    }
}
```
Take Home Messages

- An array is a sequence of values of the same type.
- An array index in an array `data` must be \( \geq 0 \) and \( < \text{data.length} \).
- An array variable specifies the location of an array; copying the reference yields a second reference to the same array.
- Arrays can occur as method parameters and return values (passing “by reference”).