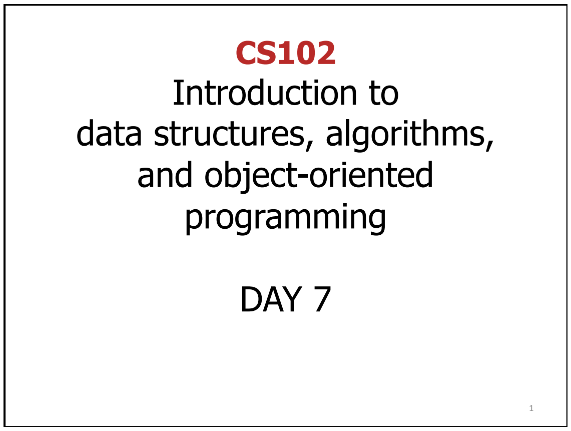
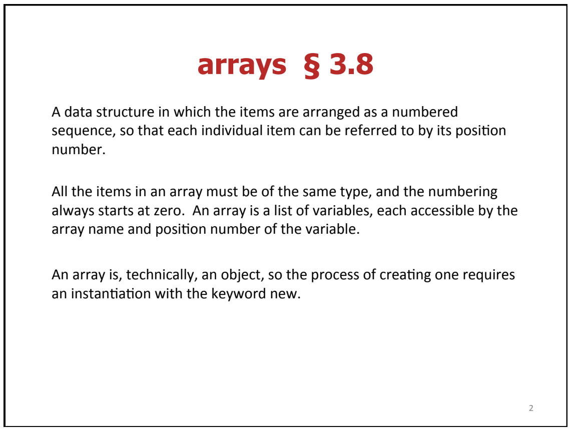


CS102
Introduction to
data structures, algorithms,
and object-oriented
programming

DAY 7

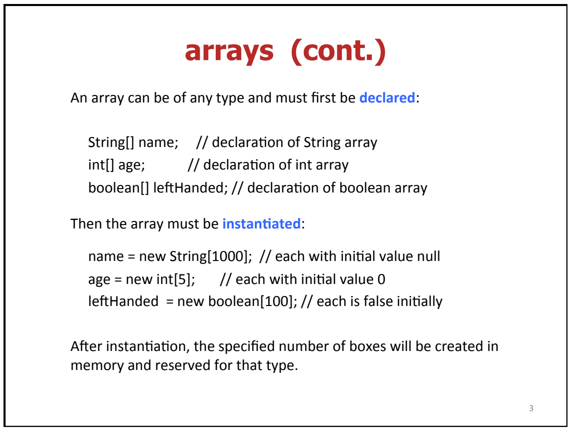


arrays § 3.8

A data structure in which the items are arranged as a numbered sequence, so that each individual item can be referred to by its position number.

All the items in an array must be of the same type, and the numbering always starts at zero. An array is a list of variables, each accessible by the array name and position number of the variable.

An array is, technically, an object, so the process of creating one requires an instantiation with the keyword new.



arrays (cont.)

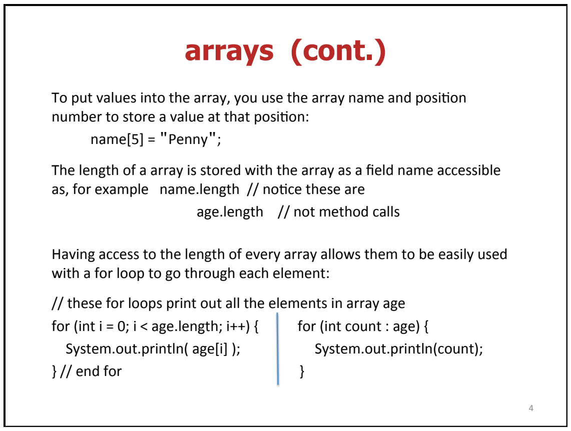
An array can be of any type and must first be **declared**:

```
String[] name;    // declaration of String array
int[] age;        // declaration of int array
boolean[] leftHanded; // declaration of boolean array
```

Then the array must be **instantiated**:

```
name = new String[1000];  // each with initial value null
age = new int[5];      // each with initial value 0
leftHanded  = new boolean[100]; // each is false initially
```

After instantiation, the specified number of boxes will be created in memory and reserved for that type.



arrays (cont.)

To put values into the array, you use the array name and position number to store a value at that position:

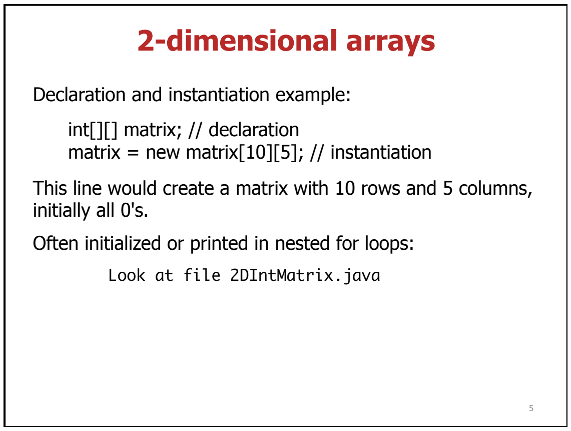
```
name[5] = "Penny";
```

The length of a array is stored with the array as a field name accessible as, for example name.length // notice these are
age.length // not method calls

Having access to the length of every array allows them to be easily used with a for loop to go through each element:

```
// these for loops print out all the elements in array age
for (int i = 0; i < age.length; i++) {
   System.out.println( age[i] );
} // end for
```

```
for (int count : age) {
    System.out.println(count);
}
```



2-dimensional arrays

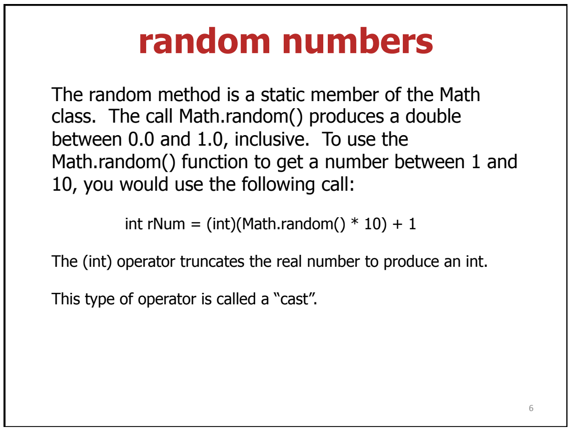
Declaration and instantiation example:

```
int[][] matrix; // declaration
matrix = new matrix[10][5]; // instantiation
```

This line would create a matrix with 10 rows and 5 columns, initially all 0's.

Often initialized or printed in nested for loops:

```
Look at file 2DIntMatrix.java
```



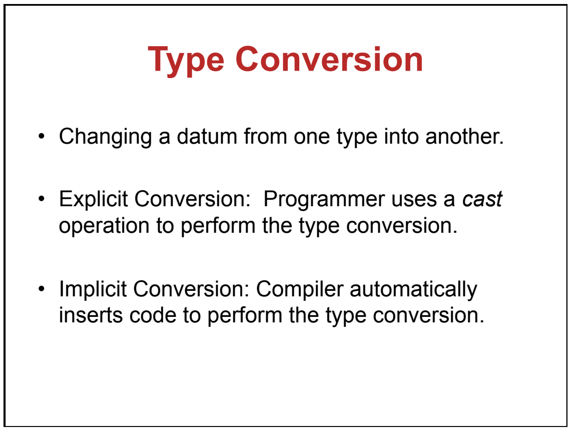
random numbers

The random method is a static member of the Math class. The call Math.random() produces a double between 0.0 and 1.0, inclusive. To use the Math.random() function to get a number between 1 and 10, you would use the following call:

```
int rNum = (int)(Math.random() * 10) + 1
```

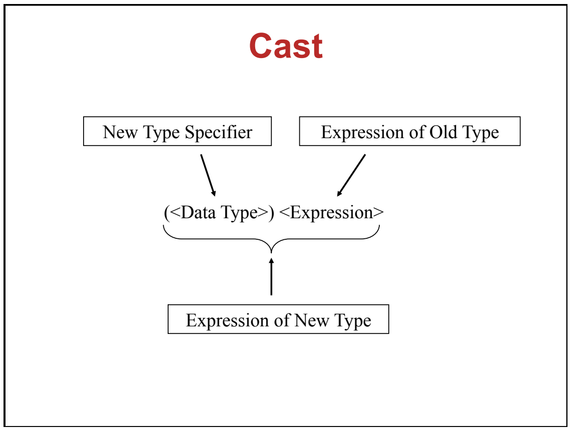
The (int) operator truncates the real number to produce an int.

This type of operator is called a "cast".

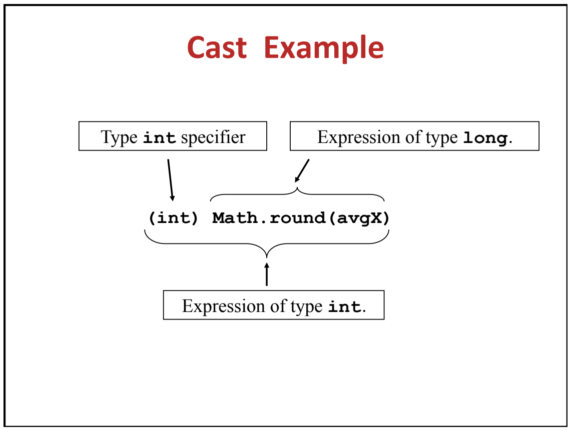


Type Conversion

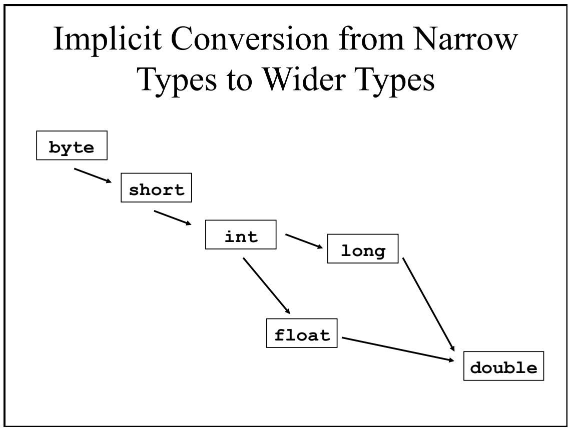
- Changing a datum from one type into another.
- Explicit Conversion: Programmer uses a *cast* operation to perform the type conversion.
- Implicit Conversion: Compiler automatically inserts code to perform the type conversion.



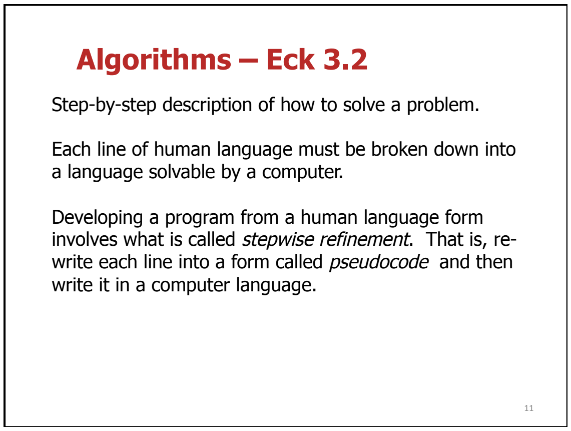
Cast



Cast Example



Implicit Conversion from Narrow Types to Wider Types

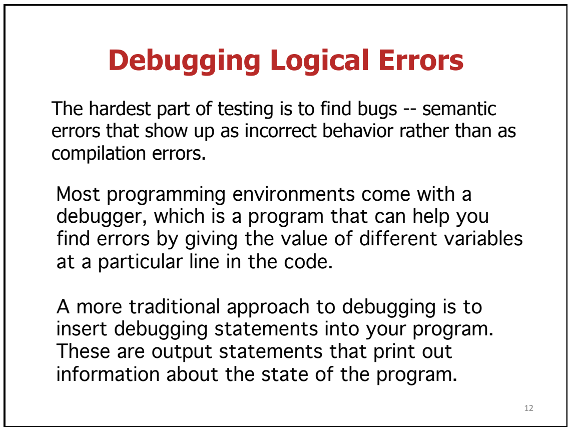


Algorithms – Eck 3.2

Step-by-step description of how to solve a problem.

Each line of human language must be broken down into a language solvable by a computer.

Developing a program from a human language form involves what is called *stepwise refinement*. That is, re-write each line into a form called *pseudocode* and then write it in a computer language.



Debugging Logical Errors

The hardest part of testing is to find bugs -- semantic errors that show up as incorrect behavior rather than as compilation errors.

Most programming environments come with a debugger, which is a program that can help you find errors by giving the value of different variables at a particular line in the code.

A more traditional approach to debugging is to insert debugging statements into your program. These are output statements that print out information about the state of the program.