

Casting

```
int x;  
double y;  
  
x = 10;  
y = (double)x;
```

Arrays

“Lists” of Variables

Sometimes, it becomes necessary to order variables in a list:

```
Temperature on Sept 1 = 90  
Temperature on Sept 2 = 87  
Temperature on Sept 3 = 84
```

in order to spot trends.

Lists of variables

- All of the variables and values are of the same type – integers, strings, whatever.
- Their order is important.
- Their place in the list is important
- May be shortened as...

Temp1 = 90
Temp2 = 87
Temp3 = 84

Track the order as a separate variable

Called an “index”:

Temp[1] = 90
Temp[2] = 87
Temp[3] = 84

x = 2
Temp[x] = ?

Name [index] = value.... index can change!

Arrays

- A difficult concept at first : `int z[5];`
- Means: `z[0], z[1], z[2], z[3], z[4]` are separate integer variables
- Starts at 0.
- An array with 5 elements is *indexed* from 0 to 4.
- in Java: `int z[] = new int[5];`

Array example

```
int x;  
int z[ ] = new int[10]; // integer too complicated,  
                        // needs "new", z[0] to z[9]  
for (x = 0; x < 10; x = x+1)  
{  
    z[x] = x;  
}
```

What is the value of z[4]? 4!

in fact...

z[0] = 0 z[1] = 1 z[2] = 2 ... to z[9] = 9

Arrays do not have to be numbers

```
int x;  
String myArray[ ] = new String[5];  
myArray[0] = "hello";  
myArray[1] = "we";  
myArray[2] = "have";  
myArray[3] = "a quiz";  
myArray[4] = "on Monday";  
for (x=0; x <= 4; x++)  
{  
    System.out.println( myArray[x] );  
}
```

What are arrays good for? –

- They are confusing just to read...

```
int a[ ] = new int[ 5 ];  
    EmployeeInfo[ 1021 ]  
        Temperature[ x ]
```

What are arrays good for? –

- They are hard to track... “context awareness” is difficult
- What's going on in this code?

```
int x, RunningTotal = 0;  
int Sum[ ] = new int[ 5 ];  
for (x = 0; x <= 4; x++)  
{  
    RunningTotal = RunningTotal + x  
    Sum[ x ] = RunningTotal;  
}
```

- Sum[0] = 0
- Sum[1] = 0 + 1
- Sum[2] = 0 + 1 + 2
- Sum[3] = 0 + 1 + 2 + 3
- Sum[4] = 0 + 1 + 2 + 3 + 4 = 10

What are arrays good for?

- Think of the use of lists, that are
 - ordinal (ordered in sequence)
 - item positions are important
 - lists can be anything: strings, names, numbers, ages, costs, addresses

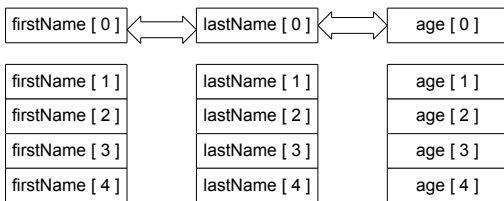
Using three arrays to keep information

```
String lastName[ ] = new String[ 5 ];  
String firstName[ ] = new String[ 5 ];  
int age[ ] = new int[ 5 ];
```

e.g.

```
lastName[ 0 ] = "Truman";  
firstName[ 0 ] = "Harry";  
age[ 0 ] = 175;
```

array items are correlated by index



continued

```
lastName[ 1 ] = "Garcia";  
firstName[ 1 ] = "Jerry";  
age[ 1 ] = 55;
```

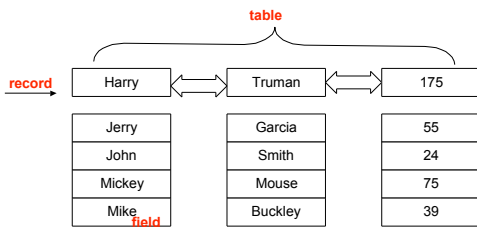
```
lastName[ 2 ] = "Smith";  
firstName[ 2 ] = "John";  
age[ 2 ] = 24;
```

continued

```
lastName[ 3 ] = "Mouse";  
firstName[ 3 ] = "Mickey";  
age[ 3 ] = 75;
```

```
lastName[ 4 ] = "Buckley";  
firstName[ 4 ] = "Mike";  
age[ 4 ] = 39;
```

array items are correlated by index



What is this?

- Information that is grouped by index
- Kept in arrays

e.g. lastName[1], firstName[1], and age[1] go together to form one profile for person #1

Is a **DATABASE**.

We're creating our own database.

printing out the info – note *index*

```
int index = 0;

userInput = JOptionPane.showInputDialog("Enter index: ");
index = Integer.parseInt( userInput );

if (index >=0) && (index <=4)
{
    System.out.println(" Item: " + index );
    System.out.println("First Name: " + firstName[ index ] );
    System.out.println("Last Name: " + lastName[ index ] );
    System.out.println("Age: " + age[ index ] );
}
```

just imagine...

- Each array contains thousands of items
- More arrays to hold soc. sec. #, birthdate, pay scale, years of service
- Program ways to enter data as well as display.
- A full-scale Database Management program.

Changing array values - age

```
public static void changeAge( int indexToChange )
{
    int newAge;
    userInput = JOptionPane.showInputDialog("Enter age: ");
    newAge = Integer.parseInt( userInput );

    age[ indexToChange ] = newAge;
}
```

note:

- array index is a parameter *passed into the method*
- nothing is returned, array is changed directly *within the method* (why? because the arrays are *public*)

Changing array values – first name

```
public static void changeFirstName( int indexToChange )
{
    userInput = JOptionPane.showInputDialog("Enter First: ");

    firstName[ indexToChange ] = userInput;
}
```

Changing array values – last name

```
public static void changeLastName( int indexToChange )
{
    userInput = JOptionPane.showInputDialog("Enter Last: ");

    lastName[ indexToChange ] = userInput;
}
```

printing out the info – note *index*

```
// get user request, exit after user enters 4 or greater
int index = 0;
do
{
    userInput = JOptionPane.showInputDialog("Enter index: ");
    index = Integer.parseInt( userInput );

    System.out.println("First Name: " + firstName[ index ] );
    System.out.println("Last Name: " + lastName[ index ] );
    System.out.println("Age: " + age[ index ] + "for index " + x);

} while (index < 4 )
```
