Middlesex University London

Lecture 8

Single-Dimensional Arrays

THIS WEEK:

- Single-Dimensional Arrays
- Writing into Array
- Reading from Array
- Operations on Array
- Finding min and max element
- Extras: Searching (linear search and binary search)
- Extras: Sorting (selection sort and insertion sort)

DISCUSSION

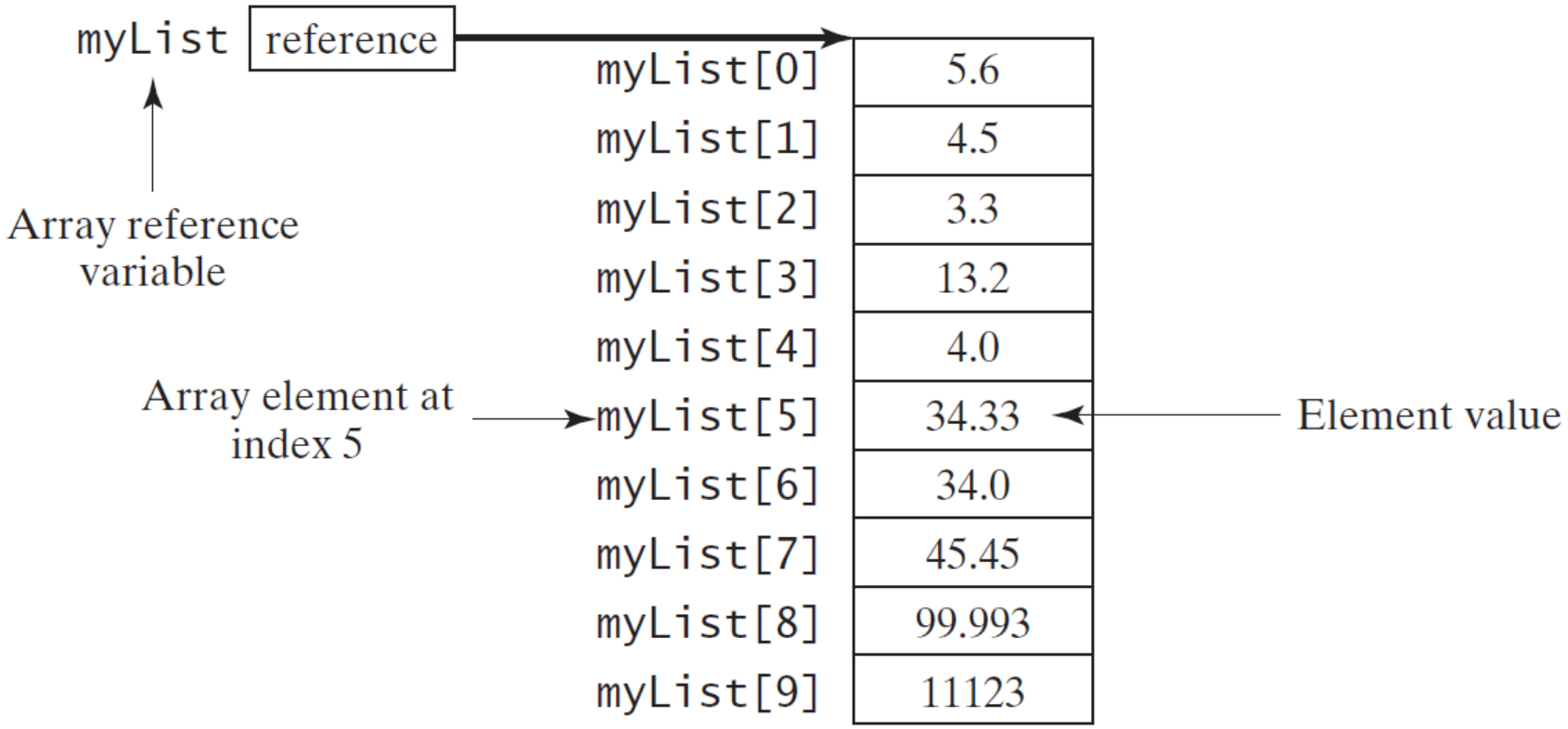
- What's are variables?
- How variables are used in programming ?
- Can the variable store multiple values ?
  - int number

WHAT IS ARRAY

- Array is a **data structure** that represents a collection of the same types of data, holding **fix number** of values
- To store multiple **values of the same type** an array of variables has to be created
- It allows easy access and reduce complexity of the code and readability of the code

INTRODUCING ARRAY

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



DECLARING ARRAY

```
datatype [] arrayRefVar;
```

Example:

```
double [] myList;
```

CREATING ARRAY

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

Example:

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

DECLARING & CREATING ARRAY

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

Example:

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

ARRAY SIZE AND DEFAULT VALUES

- Size of array cannot be change. Once an array is created, its size is fixed. It cannot be changed.
- You can find its size using

```
arrayRefVar.length
```

Example: `myList.length;` Returns: 10

When an array is created, its elements are assigned the default value of: 0 for the numeric primitive data types, '\u0000' for char types, and false for boolean types.

INDEXED VARIABLES

- The array elements are accessed through the index.
- The array indices are *0-based*, i.e., it starts from 0 to arrayRefVar.length-1.
- In the previous example, myList holds ten double values and the indices are from 0 to 9.

Index	0	1	2	3	4	5	6	7	8	9
Value	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

- Each element in the array is represented using the following syntax, known as an *indexed variable*:

```
arrayRefVar[index];
```

INITIALIZE ELEMENTS IN AN ARRAY

```
arrayRefVar[index] = value;
```

Example:

```
myList[0]=10.5;
myList[1]=11.5;
myList[2]=12.2;
myList[3]=10;
myList[4]=10.5;
myList[5]=11.5;
myList[6]=12.2;
myList[7]=10;
myList[8]=10.5;
myList[9]=11.5;
```

myList

Index	0	1	2	3	4	5	6	7	8	9
Value	10.5	11.5	12.2	10.0	10.5	11.5	12.2	10.0	10.5	11.5

INITIALIZE ELEMENTS IN AN ARRAY

```
for (int index=0; index <arrayRefVar.length; index++){
        arrayRefVar[index] = value;
}
```

Examples:

```
Scanner sc = new Scanner (System.in);

for (int i = 0; i < myList.length; i++) {
    System.out.println("Enter an element");
    myList[i] = sc.nextDouble();
}
```

```
Scanner sc = new Scanner (System.in);
System.out.print("Enter " + myList.length + " values: ");
for (int i = 0; i < myList.length; i++) {
        myList[i] = sc.nextDouble();
    }
```

ARRAY INITIALIZER

- A shorthand notation:

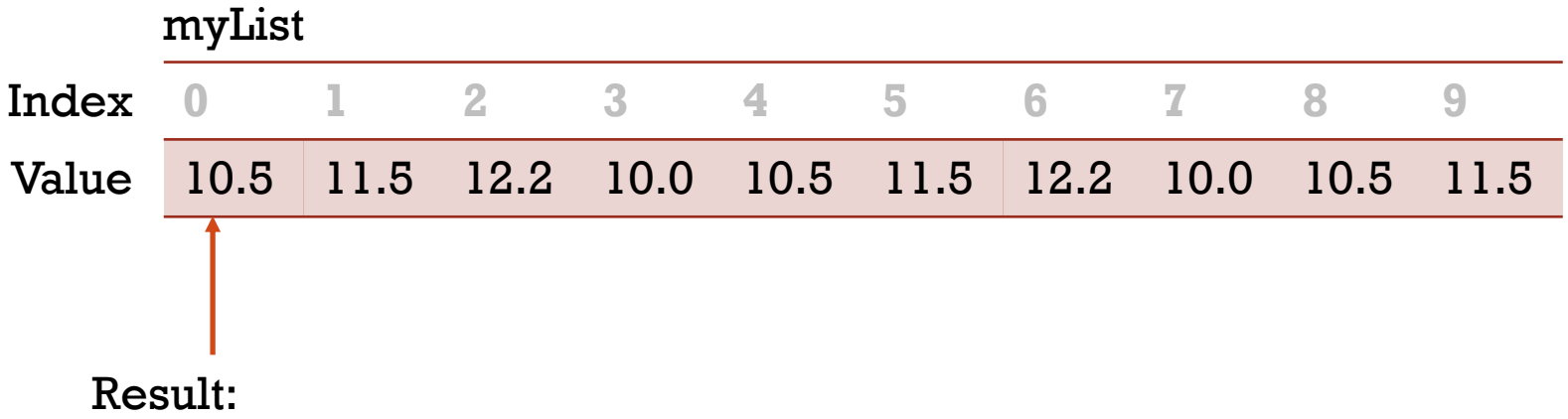
```
datatype [] arrayRefVar = {value, value, value, vale};
```

Example:

```
double [] myList = {10.5, 11.5, 12.2, 10.0,10.5, 11.5, 12.2, 10.0, 10.5, 11.5};
```

RETRIEVE AN ELEMENT FROM AN ARRAY

```
arrayRefVar[index];
```



Example:

```
System.out.println(myList[0]);
```

RETRIEVE ELEMENTS FROM AN ARRAY

```
for (int index=0; index <arrayRefVar.length; index++){
        System.out.println(arrayRefVar[index]);
}
```

Example:

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

}
```

FOREACH LOOP AS ANOTHER WAY OF RETRIEVING ELEMENTS FROM AN ARRAY:

Forech loop allow traveling the array sequentially without using an index

```
for (datatype element :arrayRefVar ){
// process the element
}
```

Example:

```
for (double l : myList){
    System.out.println(l);
}
```

TASK

- Create an array of students ages sitting around your table.

PASSING ARRAY TO METHOD

When passing array to a method the reference of the array is pass to the method

```
public static void printArray (int [] array){
    for (int n : array){
        System.out.println(n);
    }
}
```

PASSING AN ARRAY TO METHOD

```
public static void main(String[] args) {
    int [] n = {100,6,8,76,3,9,5,7};
    largestElement(n);

}

public static void largestElement(int []ListOfNumbers){
    int max = ListOfNumbers[0];
    for (int i = 1; i < ListOfNumbers.length; i++) {
        if (ListOfNumbers[i]>max)
            max=ListOfNumbers[i];
    }
    System.out.println("The largest element is: "+max);
}
```

TASK

- Create a method which would be receiving an array as a parameters and calculating the smallest element in it.

RETURNING AN ARRAY FROM METHOD

```
public static void main(String[] args) {
    int [] n = {100,6,8,76,3,9,5,7};

    for (int e : reverse(n)){
        System.out.println(e);
    }

}

public static int[]  reverse(int []list){
    int [] result = new int [list.length];
    for (int i = 0, j=result.length-1; i < result.length; i++,j--) {
        result[j]= list[i];
    }
    return result;
}
```

KEY THINGS YOU SHOULD UNDERSTAND AFTER WEEK 6

- What is an array
- How to declare and initialize elements of array
- How to get elements from an array
- Passing and returning array from a method

MASTERING YOUR SKILLS

- Read chapter 7 and do all the exercises from the end of the book