University of Southampton

Loops and Arrays

(slides adapted from D. Millard)

Thai Son Hoang

ECS, University of Southampton, U.K.

COMP1202
23rd October 2023

Recap

University of Southampton

- Object-oriented programming
  - Classes / objects
- Variables
  - capturing object properties
- Methods
  - representing object behaviours
- Computational thinking
- Encapsulation
  - **private** vs **public**
  - Constructor

Objectives

University of Southampton

- Looping
  - **while**
  - **do** . . . **while**
  - **for**
- Arrays
  - indexes
- "for each" loop

Readings

- Chapter 4.9 of Barnes and Kölling [2016]

Looping

- Sometimes we want a computer to do something
  over and ...
  over and ...
  over and ...
  over and ..
  over again ...
- We can achieve this with a loop
- A programming structure that performs a series of instructions repeatedly until some specified condition is met

There are different kinds of loops...

- **while**
- **do** . . . **while**
- **for**
- "**for** each" (enhanced **for**)
- Recursion (not really a loop but has repetitive behaviour)

While: The Monarch of Loops

- The fundamental loop
- All of the other loops can be built from this

While

Just like an if statement

```
while ( condition ) {
    code to run;
}
```

```
if (condition) {
    code to run;
}
```

Zero or more

- The condition is checked at the start of the loop
- so the code inside may not run at all

While

Example 1 (1/2)

University of Southampton

Example 1

```
int i = 0;
while (i<10){
    System.out.println("The number is " + i);
    i= i+1;
}
System.out.println("i is now " + i);
```

Vevox (106-493-186)?

What numbers will be printed by the loop and the final statement?

While

Example 1 (2/2)

University of Southampton

Example 1

```
int i = 0;
while (i<10) {
    System.out.println("The number is " + i);
    i= i+1;
}
System.out.println("i is now " + i);
```

Answer

- `i` starts as `0` and the loop stops when `i` becomes equal to `10`.
- So the loop will print the numbers `0, 1, .., 9`
- and the final statement will print number `10`

While

Example 2 (1/2)

University of Southampton

Example 2

```
int i = 10;
while (i<10) {
    System.out.println("The number is " + i);
    i= i+1;
}
System.out.println("i is now " + i);
```

Vevox (106-493-186)?

What numbers will be printed by the loop and the final statement?

While

Example 2 (2/2)

University of Southampton

Example 2

```
int i = 10;
while (i<10) {
    System.out.println("The number is " + i);
    i= i+1;
}
System.out.println("i is now " + i);
```

Answer

- `i` starts as `10` and the loop stops when `i` becomes equal to `10`.
- So the body of the loop will not start at all
- and the final statement will print number `10`

University of Southampton

Example 3

```
int i = 0;
while (i<10) {
    System.out.println("The number is " + i);
    i= i * 1;
}
System.out.println("i is now " + i);
```

Vevox (106-493-186)?

What numbers will be printed by the loop and the final statement?

University of Southampton

Example 3

```
int i = 0;
while (i<10) {
    System.out.println("The number is " + i);
    i= i * 1;
}
System.out.println("i is now " + i);
```

Answer

- This is called an infinite loop. They cause programs to hang and/or crash.
- So the final statement will not start at all
- Always check that your loops will end!

University of Southampton

Blackboard Tests for While Loops

University of Southampton

Do While

```
do {
    code to run;
} while (condition);
```

```
while (condition) {
    code to run;
};
```

Once or more

- The condition is checked at the end of the loop
- so the code inside run at least once

Do While

Example 4 (1/2)

Example 4

```
int i = 0;
do {
    System.out.println("The number is " + i);
    i= i+1;
} while (i<10);
System.out.println("i is now " + i);
```

Vevox (106-493-186)?

What numbers will be printed by the loop and the final statement?

Do While

Example 4 (2/2)

Example 4

```
int i = 0;
do {
    System.out.println("The number is " + i);
    i= i+1;
} while (i<10);
System.out.println("i is now " + i);
```

Answer

- `i` starts as `0` and the loop stops when `i` becomes equal to `10`.
- So the loop will print the numbers `0, 1, .., 9`
- and the final statement will print number `10`

Do While

Example 5 (1/2)

Example 5

```
int i = 10;
do {
    System.out.println("The number is " + i);
    i= i+1;
} while (i<10);
System.out.println("i is now " + i);
```

Vevox (106-493-186)?

What numbers will be printed by the loop and the final statement?

Do While

Example 5 (2/2)

Example 5

```
int i = 10;
do {
    System.out.println("The number is " + i);
    i= i+1;
} while (i<10);
System.out.println("i is now " + i);
```

Answer

- `i` starts as `10` and
  the loop stops when `i` becomes no less than `10`.
- So the loop will print the numbers `10`
- and the final statement will print number `11`

Do While

Relationship with **while** loop

How can a **do** ... **while** loop, i.e.,

```
do {
    code to run;
} while (condition);
```

be represented using a **while** loop?

Self-Checked

Do ... While loops

Blackboard Tests for Do...While Loops

Outline

For Loop: Determinstic Loop

```
for (initialisation; condition; statechange)
{
    code to run;
}
```

Counting

The for loop is a simple way of repeating a block of code when you know in advance how many iterations (times around the loop) that you want.

For Loop

Example 6

Example 6

Initialisation ↓ Condition ↓ StateChange ↓

```
for (int i = 0;    i < 10;    i = i + 1)
{
    System.out.println("The number is " + i);
}
```

Vevox (106-493-186)?

What numbers will be printed by the loop?

For Loop

Example 7 (1/2)

Example 7

```
for (int i = 0;    i < 10;    i = i + 1)
{
    System.out.println("The number is " + i);
}
System.out.println("i is now " + i);
```

Vevox (106-493-186)?

What numbers will be printed by the loop and the final statement?

For Loop

Example 7 (2/2)

Example 7

```
for (int i = 0;    i < 10;    i = i + 1)
{
    System.out.println("The number is " + i);
}
System.out.println("i is now " + i);
```

Answer

- Notice the scope of `i`.
- `i` can only be used within the body of the loop
- The code cannot be compiled

For Loop

University of Southampton

Relationship with **while** loop

How can a **for** loop, i.e.,

```
for (initialisation; condition; statechange)
{
    code to run;
}
```

be represented using a **while** loop?

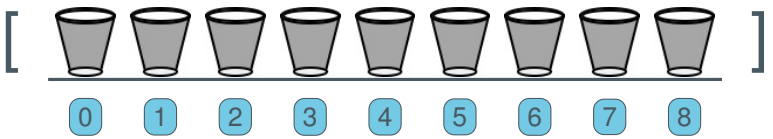
- **for** loop is often neater than **while** loop with the same functionality.

Arrays

University of Southampton

- Sometimes we want to group things together
  - "things" being objects and primitives
- The most fundamental collection is an Array
- If a variable is a cup ...

an array is a shelf of cups



Arrays

University of Southampton

More details (1/2)

- Arrays are objects
- But they have their own special syntax
- To declare an Array:

```
int[] numberStore;
```

or

```
int numberStore[];
```

Both of these are valid.
They do the same thing!

Arrays

University of Southampton

More details (2/2)

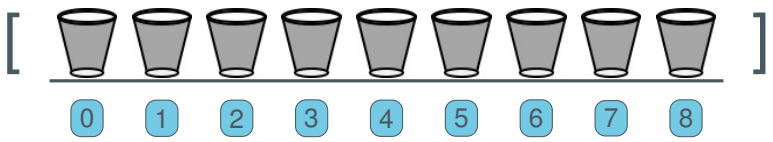
- To instantiate an Array:

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

This is the length you want the array to be (the number of cups)

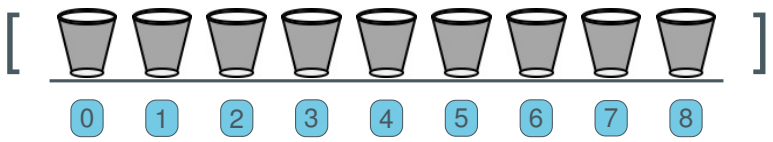
N.B that we are creating an object here (using the **new** keyword), but we don't call a constructor. Arrays have their own special syntax.

Index



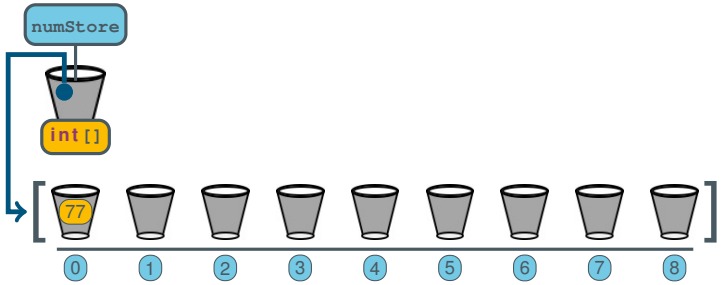
- An index is the position in the array.
- Java arrays start at 0

Length



- The length is the number of indexes
- Here the length is 9

Using indexes (insertion)




```
int[] numStore;
numStore = new int[9];
numStore[0] = 77;
System.out.println(numStore[0]);
```

Declaration
Instantiation
Assignment
Retrieval

Array rules

- An array can hold objects or primitives
- An array is an object, regardless of what it contains
- Arrays care about type
  - if you declare an **int[]** you can only put **int**s in it[1]

[1] (not entirely true, look up "implicit widening" if you are interested)

When you play with arrays...

- .... you will learn to know and hate the
  - `ArrayIndexOutOfBoundsException`
- This happens when you try and access an index that isn't there
- Lets use a **for** loop to expose this

Iterating Over an Array (1/2)

```
int numStore = new int[9];
//some missing code to fill the array with values

for (int i = 0; i < 9; i=i+1) {
    System.out.print("Number at position " + i);
    System.out.println(" is " + numStore[i]);
}
```

Arrays are powerful because we can use a variable as the array index.
This means we can iterate over them in a loop.

Iterating Over an Array (2/2)

```
int numStore = new int[9];
//some missing code to fill the array with values

for (int i = 0; i < 10; i=i+1) {
    System.out.print("Number at position " + i);
    System.out.println(" is " + numStore[i]);
}
```

But make sure the index is always valid (in bounds).
Otherwise the program will crash when it reaches the line that tries to use an index that is out of bounds

For Each Loop (1/2)

- Iterating over an array is so common that ...
- ... Java now includes a loop specifically to do it.
- The "**for** `each`" loop (aka enhanced **for** loop) is designed to work with collections like arrays

```
for (type variableName : collection) {
    code to run;
}
```

For Each Loop (2/2)

```
for (int i = 0; i < 10; i=i+1) {
    System.out.print("Number at position " + i);
    System.out.println(" is " + numStore[i]);
}
```

becomes

```
for (int n : numStore) {
    System.out.println("Number is " + n);
}
```

- Like the **for** loop, the "**for each**" loop is a shortcut, that is a bit neater than writing the code the long way.
- But it can only be used for access (e.g. `n = n + 1` would not increment the value in the array)
- And it hides the current index

Self-Checked

For and For Each loops

Blackboard Tests for For Loops

Blackboard Tests for For Each Loops

More Blackboard Tests

Summary

- Looping
  - **while**
  - **do** ... **while**
  - **for**
- Arrays
  - indexes
  - **for each** loops

YOUR QUESTIONS

References I

- David J. Barnes and Michael Kölling. *Objects First with Java: A Practical Introduction using BlueJ*.
  Pearson, sixth edition edition, 2016 (Chapter 4.9)