ARCTICTIGER

Chapter 5
More Data Types and Operators

Based on the course literature:

Java: A beginner’s guide

Fifth Edition

Herbert Schildt

What we’ll cover

- Arrays, normal, multidimensional, irregular
- For-each loop
- Strings
- Command line arguments
- Bitwise operators
- Ternary operator ?

Arrays

- An array is a collection or list of variables all of which have the same type.
- Arrays can have one or more dimension.
- Like with JavaScript Arrays are implemented as objects in Java.

One dimensional array

A list
Account numbers, ages, userids etc.

```
type array-name[] = new type[size];

int sample[] = new int[10];
// or
int sample[]; // Creates a reference
sample = new int[10]; // Allocates memory for array and updates reference

sample[0]; // 0 indexed arrays. So this is the first item.
sample[9]; // 9 is the last in a array with a length of 10.
```

Demo1 – 2

Multidimensional arrays

- We use an array of one or more arrays to represent dimensions.
- A two dimensional array is used to represent a table.

0,0	0,1	0,2	0,3
1,0	1,1	1,2	1,3
2,0	2,1	2,2	2,3
3,0	3,1	3,2	3,3
4,0	4,1	4,2	4,3

```
int table[] [] = new int[4] [5];
table[1][1] = 5;
```

0,0	0,1	0,2	0,3
1,0	1,1	1,2	1,3
2,0	2,1	2,2	2,3
3,0	3,1	3,2	3,3
4,0	4,1	4,2	4,3

Demo 3

Irregular Arrays

```
int table[][] = new int[3][];
table[0] = new int[4];
table[1] = new int[2];
table[2] = new int[10];
```

More dimensions

```
int multidim [][][][] = new int[5][4][3][7];

//North, East, Height, Time;
```

Array declarations

```
int counter[] = new int[3];
int[] counter = new int[3];
int []counter = new int[3];

char table[][] = new char[3][4];
char[][] table = new char[3][4];
char [][]table = new char[3][4];
```

```
int[] nums, nums2, nums3; // 3 arrays
int nums[], num2, num3; // 1 array, 2 int
```

Returning array from a method

```
int[] someMethod() { …
```

For-Each Style Loop

```
for(type itr-var: collection) statement;

int[] nums = {2,4,5,8};
for(int val: nums){
        if(val > 4){
                break;
        }
        val = 200; // This has doesn’t change nums
}
```

Demo 4

Strings

- Strings are objects in Java, not an array of characters.

```
“This is a string literal”
String str = new String(“Hello”);
String str2 = “Hello world”;
```

Useful String methods

boolean equals(str)	Returns true if the string contains the same char sequence as str.
int length()	Obtains the number of chars in the string.
char charAt(index)	Obtains the char at a given index.
int compareTo(str)	Returns 0 if they are the same. Returns > 0 if the string is greater than str.Returns < 0 otherwise
int indexOf(str)	Returns the first index of a match. -1 if it doesn't contain str.
int lastIndexOf(str)	Returns the last index of any matches. -1 it doesn't contain str.

Demo 5

More string stuff

- To concatenate strings we use the +

- Why not == to compare strings.
- Strings a re objects and == checks to see if the references point to the same object.

Even more String stuff

- Strings are immutable (non changeable).
- String methods often generate new strings.

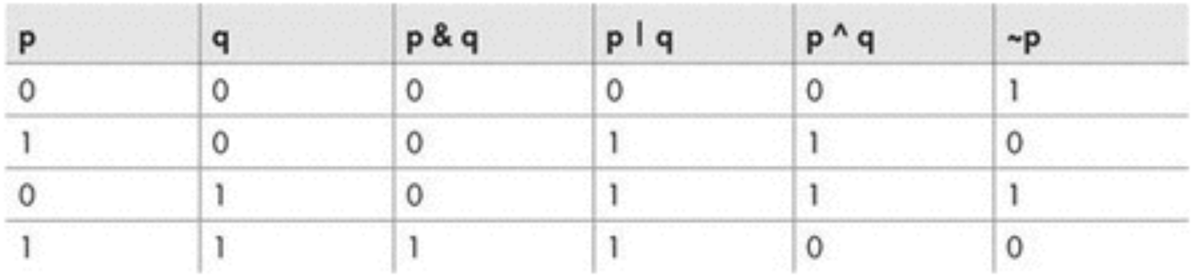
Bitwise operators

- Bitwise operators work only with long, int, short,char or byte.
- They are used to work with the actual bits that make up a number.

Bitwise operators

Operator	Result
&	Bitwise and
|	Bitwise or
^	Bitwise xor
>>	Shift right
>>>	Unsigned shift right
<<	Shift left
~	Uniary not

Bitwise operators



p	q	p & q	p | q	p ^ q	~p
0	0	0	0	0	1
1	0	0	1	1	0
0	1	0	1	1	1
1	1	1	1	0	0

$$\begin{array}{rl} & 1101\ 0011 \\ \& & \underline{1010\ 1010} \\ & 1000\ 0010 \end{array}$$

&

- & can be used to switch bits off

$$\begin{array}{rr} & 1101\ 0011 \\ \& & \underline{1010\ 1010} \\ & 1000\ 0010 \end{array}$$

Demo 6

|

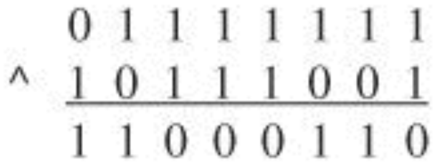
- | (or) can be used to switch bits on.

```
  1101 0011
| 1010 1010
  ---------
  1111 1011
```

Demo 7

^

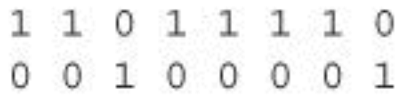
- ^ (XOR, exclusive or)



$$\begin{array}{cc} & 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\ \wedge & \underline{1\ 0\ 1\ 1\ 1\ 0\ 0\ 1} \\ & 1\ 1\ 0\ 0\ 0\ 1\ 1\ 0 \end{array}$$

~

- ~ (Unary Not) Inverses the switches
- Off becomes On and On becomes Off



```
1 1 0 1 1 1 1 0
0 0 1 0 0 0 0 1
```

Demo 8

ARCTICTIGER

Shift operators

>>	Shift right
>>>	Unsigned shift right
<<	Shift left

?

- ? or ternary operator

```
if(val < 0){
        absVal = -val;
} else {
        absVal = val;
}
absVal = (val < 0) ? -val : val;
```