Arrays

Definition

A collection of data of the same type[1]

Properties

First element starts at zero

Index determines location of item

Size of array in property `length`

Example of one dimensional array is shown here

index	data
0	416-922-1131
1	416-332-4352
2	416-567-3453
3	416-653-2342
4	416-463-2522
5	905-363-2722
6	905-263-2233

[1]An Introduction to Computer Science using Java, John Carter, p.308

Examples

index	data
0	23
1	73
2	74
3	100
4	865
5	-34
6	0
7	876

index	data
0	John
1	Suzie
2	Jeremy
3	Ronald
4	Philip
5	Chuan

index	data
0	10.34.11.194
1	10.34.18.199
2	10.34.188.1
3	10.34.250.0

Examples of one dimensional arrays

Declaration and Instantiation in Java

Arrays in Java need to be declared like any other variable

Arrays are *objects*. They are not primitives. Therefore, they need to be instantiated. Instantiating reserves memory for the array when the program is running. We may not know the size of the array at compile time

If the elements of the array are primitives, memory is reserved for them when the array is instantiated but, *if the elements of the array are objects, the objects need to be instantiated separately.* We will learn arrays of objects later

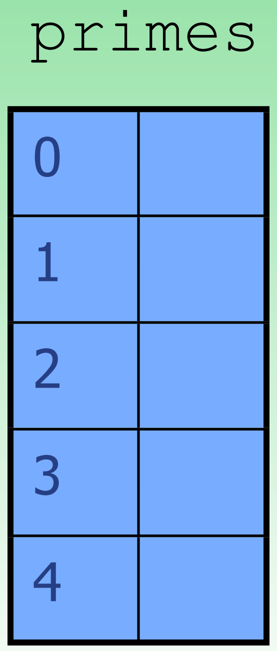
With arrays of primitives, declaration and instantiation can happen in a single statement

Declaration and Instantiation in Java

The statement

```
    int[] primes = new int[5];
```

declares and creates the structure called `primes`



The contents of the array are undefined. In this case, they could contain any integers

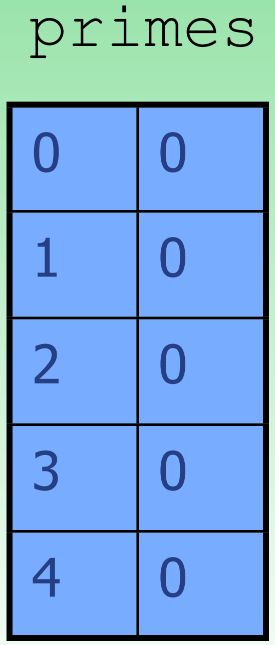
Declaration and Instantiation in Java

$$\overbrace{\texttt{int[] primes}}^{\text{declaration}}\ \texttt{=}\ \overbrace{\texttt{new int[5];}}^{\text{instantiation}}$$

Initialization

We can set every element of the array to zeroes with the code

```
for (int i = 0; i < primes.length; i++)
      primes[i] = 0;
```



When we initialize an array, we are sure of its contents.

Initialization of variables is highly recommended when programming.

Accessing Elements

We can access any element of the array by using the index. If we want to put the value 2 in index location zero, we write

```
primes[0] = 2;
```

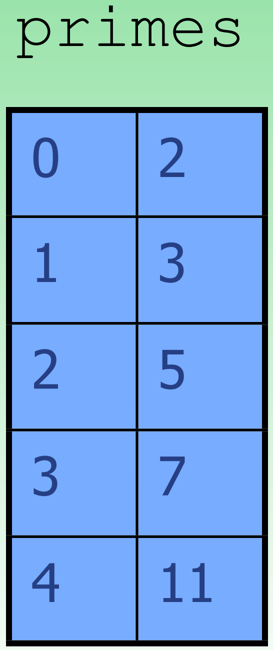
`primes`

0	2
1	0
2	0
3	0
4	0

Accessing Elements

We can put other values in a similar way

```
primes[1] = 3;
primes[2] = 5;
primes[3] = 7;
primes[4] = 11;
```



Accessing Elements

To print the value of an element of the array, such as the element in position 1, we can use

```
System.out.println(primes[1]);
```

primes

0	2
1	3
2	5
3	7
4	11

Two dimensional arrays

The statement

```
    int[][] matrix = new int[2][3];
```

declares and instantiates the structure called `matrix`

`matrix`

	column 0	column 1	column 2
row 0			
row 1			

The contents of the array are undefined

Initialization of a two dimensional array

We can set every element of the previous array to zero with the code

```
for (int row = 0; row < matrix.length; row++) {
   for (int column = 0; column < matrix[0].length; column++) {
      matrix[row][column] = 0;
   }
}
```

`matrix`

	column 0	column 1	column 2
row 0	0	0	0
row 1	0	0	0

Exercises

Exercise 1

Declare and instantiate a one dimensional array of integers of size 12

Initialize each element of the array to -1

Set the third element of the array, i.e. index 2, to the value 5

Print the entire array

Exercise 2

Declare and instantiate a one dimensional array of integers of size 50

Initialize each element of the array to double its index. For example, the element with index 12 should have a value of 24

Print the entire array and check the result

Exercise 3

Declare and instantiate a one dimensional array of 15 integers

Initialize each element of the array to a random number `n` such that

```
0 <= n < 1000
```

The method `Math.random()` generates a random `double`

```
0 <= Math.random() < 1
```

Print the entire array and check that all numbers are within the correct range

Exercise 4

Modify Exercise 3 so that each element of the array is initialized to a random number `n`

```
125 <= n < 750
```

Print the entire array and check that all numbers are within the correct range

Exercise 5

Linear/sequential search:

On a piece of paper, draw an array of 10 integers where each element of the array is a random number in the range [0, 50)

Write an algorithm, using a mixture of English like instructions and Java commands, that searches the array for a number that is given by a person. For example, someone might give the number 17. Your task is to write an algorithm that searches for the number in the array. If the number is there, your algorithm prints the index. If the number is not in the array, your algorithm prints -1

Order of complexity: O(n)

Exercise 6

Linear/sequential search:

Trace your algorithm of Exercise 5 by following your own instructions, showing clearly any variables that your algorithm may have.

Trace the algorithm by searching for the following numbers

```
24
60
10
```

Order of complexity: O(n)

Exercise 7

Every time that we execute the statements of a loop, we have performed an iteration. If a loop executes 10 times around, we say that the loop has performed 10 iterations.

Think about and answer the following:

a) How many iterations did your algorithm perform when searching for the number 24

b) Would everyone's iterations be the same? Why or why not?

c) What is the worst case scenario of your algorithm, i.e., when would it perform the most number of iterations?

d) What is the best case scenario of your algorithm, i.e., when would it perform the least number of iterations?

Order of complexity: O(n)

Exercise 7 (cont’d)

e) What happens to the number of worst-case scenario iterations if the array is doubled in size?

f) What about an array that is tripled in size? Quadrupled in size?

g) What kind of relation is there between the size of the array and the number of iterations?

h) If your array was 10 times larger, would a computer 10 times faster than your computer perform the search task in the same amount of time as your computer?

i) Write some conclusions from this reasoning

Order of complexity: O(n)

Exercise 8

Linear/sequential search:

Implement your algorithm of Exercise 5 in Java. Declare and instantiate a one dimensional array `list[]` of 50 integers

Initialize each element of the array to a random number `n` so that

$$0 <= n < 50$$

Write a Java program that asks the user for a number `n` and searches the array for the number `n`. If the element `n` is found, the program prints the index location. If the element n is not found, the program prints `-1`.

Order of complexity: O(n)

Exercise 9

Linear/sequential search:

Modify your program of Exercise 8 so that the list of numbers has 1000 integers and each element of the array is initialized to a random number `n` so that

```
25 <= n < 4000.
```

Order of complexity: O(n)

Exercise 10

Declare and instantiate a one dimensional array of 10 Strings

Initialize the elements of the array to 10 phone numbers with the dashes.
For example : `416-922-9273`

Write a search program similar to the one you did in exercise 8. For this program, we will be searching for phone numbers. Ask the user for a phone number and if the phone number is in the array, print both the index location and the phone number itself. If the phone number is not in the array, print `-1`

Order of complexity: O(n)

Exercise 11

Modify your program of exercise 10. Ask the user for any text. Using the `indexOf()` method, search your array and print only the items of the array that begin with the text that the user supplied. For example, if the user types

```
416
```

your program prints out all the phone numbers that begin with `416` such as `416-822-2322, 416-462-4744`

Your program also keeps a count of the phone numbers that are printed and at the end of the list prints the total. For example:

```
Total found: 2
```

Order of complexity: O(n)

Exercise 12

Modify your program of exercise 11, so that your program prints out all the phones that contain, anywhere in the phone, the text that the user types. For example, if the user types

```
647
```

the program prints out the phones

```
416-744-6471
647-433-2322
647-392-3392
905-322-0647
Total found: 4
```

Order of complexity: O(n)

Exercise 13

Using the `substring()` method, modify your program of exercise 12, so that when your program prints the phone numbers, the area code and the phone number are separated like the example shown here. If the user types `647` your program would print

```
Area code    Phone
--------------------
416          744-6471
647          433-2322
647          392-3392
905          322-0647
--------------------
Total found: 4
```

Order of complexity: O(n)

Exercise 14a

Download the following files from our Moodle website

```
students.txt
FileIO.java
```

to the very same folder where you have your programs

Order of complexity: O(n)

Exercise 14b

Write a Java program that declares and instantiates a `String` array called `students`. The array is of size 41

Right after your declaration, include this statement in your program which will read all the elements of the file `students.txt` into your array

```
FileIO.readFileIntoArray("students.txt", students);
```

Order of complexity: O(n)

Exercise 14c

To make sure that your array is loading the entries from the file correctly, print the index number of each and every entry of the array. The output of your program should look like this:

```
0 465224154Einstein
1 196754642Beethoven
2 467845721Decartes
3 874857211Plato
4 951654322Newton
5 978456124Copernicus
6 678445784Turing
7 578497877Galileo
8 021546211Aristoteles
9 045266598Kepler
10 947845124Kelvin
.....
.....
.....
38 245454611Edison
39 216777877Hugo
40 999586665Balzac
```

Exercise 14d

If the entries in the previous exercises 14c are being loaded properly, you may remove or comment the section of the code that prints the entries.

Now, after the loading of the entries, add a `do` loop to your program. Inside the loop, ask the user for a text to search. The program keeps asking the user for a text to search until the user types `quit`, at which point the loop ends and the program prints the words `End of Program`

For example,

```
Enter text to search: Hello
Enter text to search: Wow
Enter text to search: Boogie-boogie
Enter text to search: quit
End of Program
```

Exercise 14e

Inside the loop of exercise 14d, include the necessary code so that your program prints out only the entries from the array that contain the text entered by the user. When printing the entries, put a blank space between the student number and the name of the person. For example,

```
Text to search: 23
655233240 Gould
233312442 Fermi
222232112 Boole
221546523 Bell

Enter text to search: quit
End of Program
```

Exercise 14f

Include a counter in your program so that the number of entries is printed at the bottom of the list. For example,

```
Text to search: 25
369258174 Maxwell
876625441 Hilbert
254465884 Jung
Total found: 3

Enter text to search: quit
Total found: 0
End of Program
```

Exercise 14g

Finally, put an `if` statement in the code so that the program does not search the array if the text entered by the user is the word `quit`. The counter should not print the number of entries. The program should just print the words `End of Program`. For example,

```
Text to search: 222
222232112 Boole
Total found: 1

Enter text to search: quit
End of Program
```

Exercise 15

Do a manual trace of the your program of exercise 14. Test your program with the case in slide 14f

Exercise 16

Declare an array of integers called `numbers`. Then, put numbers in the array sorted in ascending order. Then put a number anywhere in the array that is not in order. You array would look something like this: `2 3 4 8 5 10 11`

Write a Java program that finds the number that is out of order

Exercise 17

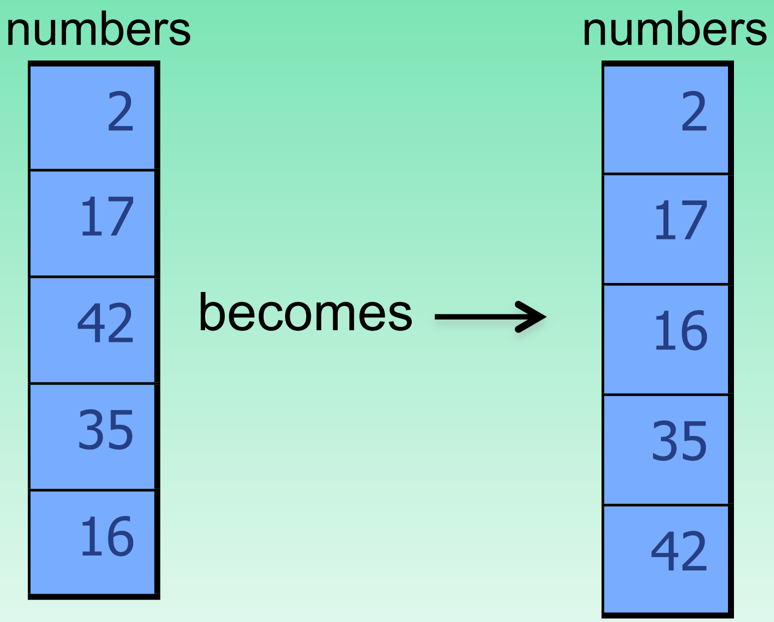
Declare an array called `numbers` of integer of size 5. Put random numbers in the array in the range [0, 50).

Print the array to see what numbers are there. Then, write the necessary code to find the largest element in the array.

Print the location of this number

Exercise 18

Add the necessary code in your program so that once you have found the largest element of the array, you swap it with the last element of the array. In this example, the largest element is `42` and has been swapped with the last element of the array:



Exercise 19

Declare and instantiate a two dimensional array of integers. The dimensions of the array: 5 rows and 4 columns

Initialize each element of the array to `0` (zero)

Set the element at row 3, column 2 to the value `5`

Print the entire array as a readable table of rows and columns as shown:

```
        Col 0    Col 1    Col 2    Col 3
Row 0   0        0        0        0
Row 1   0        0        0        0
Row 2   0        0        0        0
Row 3   0        0        5        0
Row 4   0        0        0        0
```

Exercise 20

Write a Java program that prints out a formatted table like the one of Exercise 9. The program must be able to handle possible future changes to the dimensions of the array.

For example, if `ROWS = 2` and `COLUMNS = 4`, the program outputs

```
        Col 0   Col 1   Col 2   Col 3
Row 0   0       0       0       0
Row 1   0       0       0       0
```

If `ROWS` is changed to `3` and `COLUMNS` is changed to `5`, the program must still print the formatted table, but no other changes can be made to the Java code:

```
        Col 0   Col 1   Col 2   Col 3   Col 4
Row 0   0       0       0       0       0
Row 1   0       0       0       0       0
Row 2   0       0       0       0       0
```

Exercise 21

Declare and instantiate a two dimensional array of integers with 15 rows and 10 columns, initialize each element of the array to a random integer in the range [0, 50) and print the entire array as a readable table of rows and columns

Create a transposition of the array. This means that the rows and the columns are swapped. Here is a 3 by 2 array and its transposed:

`matrix`

12	2
21	46
7	18

$(\texttt{matrix})^{T}$

12	21	7
2	46	18

Print the transposed array as a readable table of rows and columns

Exercise 22

Matrix addition `C = A + B` is defined if and only if both matrices `A` and `B` have the same dimensions. Addition is defined

$$c_{row,\ col} = a_{row,\ col} + b_{row,\ col}$$

Declare and instantiate two two-dimensional arrays of size 5 x 3 of integer. Initialize the arrays to random integers in the range [-6, 10)

Write a Java program that takes two matrices of equal dimensions and performs their addition, places the result in a third matrix, and prints all three matrices clearly. Here is an example that adds two 3 x 2 matrices:

```
         [A]         +          [B]           =         [C]
         ------------------------------------------------------------
         Col 0    Col 1         Col 2    Col 3          Col 4    Col 5
Row 0    2        8             -2       2              0        10
Row 1    1        0             -6       6              -5       6
Row 2    5        7             3        4              8        11
```

Exercise 23

Matrix multiplication `C = A x B` is defined if and only if the number of columns of matrix `A` is the same as the number of rows of matrix `B`. Write a Java program that performs matrix multiplication

Assume that matrices `A` and `B` have the correct dimensions. If `C` is the resulting matrix, the multiplication `C = A x B` is defined as follows

$$c_{row,col} = \sum_{i=0}^{\text{columns}-1 \text{ of A}} a_{row,i} \times b_{i,col}$$

where c is an element of the resulting matrix `C`, a is an element of matrix `A`, and b is an element of matrix `B`. The subscripts *row, col, i* determine the positions of those elements in their respective matrices. To make it simpler to write the program, all subscripts begin at zero.

Exercise 24

Declare and instantiate a two dimensional array of string with 7 rows and 2 columns. Initialize the elements of the array as follows:

`John`	`416-363-2122`
`Anna`	`416-223-2711`
`Peter`	`905-332-2234`
`Susan`	`905-266-0987`
`Andrew`	`416-564-3342`
`Arthur`	`647-362-9992`
`James`	`416-333-2881`

Write a program that asks the user for a name. The program prints out the name and the phone number of the person. If the person is not in the list, the program prints out the message `***name not found`

Exercise 25

Declare and instantiate a two dimensional array of integers with 15 rows and 15 columns, initialize each element of the array to a random integer in the range [0, 50) and print the entire array as a readable table of rows and columns

Create a transposition of the array. This means that the rows and the coumns are swapped. Here is a 3 by 3 array and its transposed:

`matrix`

12	2	32
21	46	9
7	18	31

$(\texttt{matrix})^T$

12	21	7
2	46	18
32	9	31