Software Engineering and Programming Basics

Arrays, Strings, Static Key Word

Authors of slides:
Prof. Dr.-Ing. Janet Siegmund
Prof. Dr.-Ing. Norbert Siegmund
Prof. Christian Lengauer
Partly extracted from script of PD Dr. Christian Bachmaier

Catching Up I

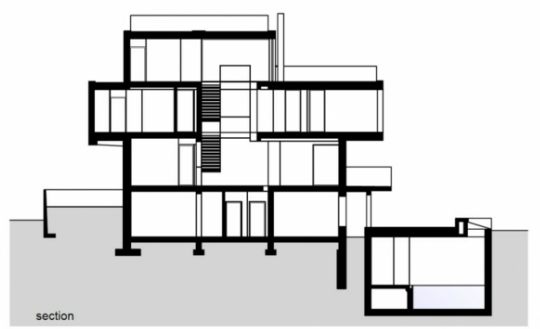
- Class consists of…
  - Attributes (structure)
  - Methods (behavior)
- Object is…
  - an instance of a class
  - attributes have a concrete value

Catching Up II

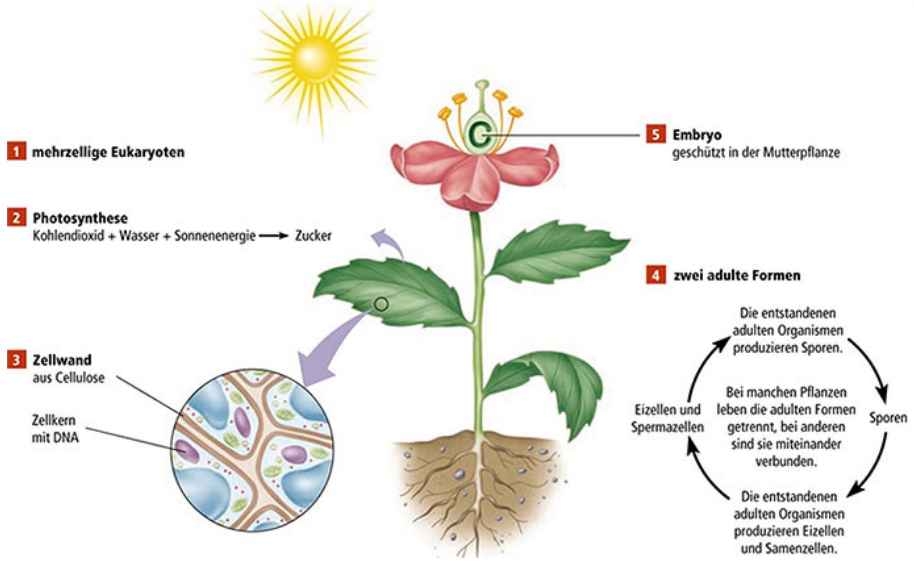
Object

Object or class?

Class



Class



Class

Name: _____

Given Name: _____

Age: _____

Address: _____

Object

Catching Up III

- There are special methods, a.k.a **constructors**, that create an object of a class
  - Constructors have no return type
  - Need to have the same name as the class

- Constructors are called via **`new`**
- Variables save a reference (pointer) to a data of a complex data type in memory

Catching Up IV

- What has precedence within a method: A local variable or an attribute in a class that has the same name as the local variable?
  - Local variable
- How can I access instance variable instead?
  - Keyword `this`

Learning Goals

- Getting to know the difference between static and instance methods and attributes
- Getting to know the simplest of the complex data types: Array
- Getting to know the class String

Static Attributs and Methods

Methods of Objects

- Example:

```
class Person {
  String givenName;
  String name;
  int age;
  Residence address;

  void setName(String name) {
    this.name = name;
  }

  String getName() {
    return this. name;
  }

  void isBirthday () {
    this.age = this.age + 1;
  }

  void movesHouse(Residence newAddress) {
    this.address = newAddress;
  }
}
```

- Instance methods are defined within a class, yet they apply to objects (instances) of this class
- Thus, you need to first create an object, before you can use an instance method

- In other words:
- Before I can ask about age, I need a concrete person

`static` Keyword

- Problem: Objects are not always necessary to execute some action
  - Mathematical computations
  - Physical formulas
  - In general: utility functions
  - Management variables to get some statistics about the objects of a class (e.g., the number of objects)

- Idea: `static` denotes methods and attributes of classes that can be called without creating an object first

Static vs. Instance: Methods

- Instance:
  - Methods can only be applied to objects of a class (need to be instantiated first)
- Static:
  - Methods can be used without creating an object
  - Statics methods cannot be applied to instance (non-static) variables and methods within the same class (because this would require an object)

Static vs. Instance: Variables

- Instance:
  - Variables are valid only per object
- Static:
  - Variables are independent of the objects of a class
  - Each object of a class sees **exactly the same variable**

Using Static

- Definition with keyword **`static`**
- Usage:
  - className.Attribute
  - className.Method
  - Hints: CLASSNAME, not name of object!
  - `Person.numPeople = 20;` (**not:** `kathryn.numPeople = 20`)
  - `MyMath.sum(4);`
- Constants
  - `final double PI = 3.14159265359; // stored per object (not good)`
  - `final static double PI = 3.14159265359; // stored only once (good)`

Static vs. Instance

- Static oder Instance… How would you implement the method/attribute?

Methode / Attribut:	Static (A)	Instance (B)
M: Change age of a person		
A: Number of all created persons		
M: Multiplying two numbers		
M: Compare a person with another person		
A: Friends of a person		
A: Books of a library		
A: Value-added tax (VAT)		

3 to 5 minutes

Arrays

Arrays I

- What do we do, when we want to store the 9 best friends (also persons) of a person?

```
class Person {
  Person freund1;
  Person freund2;
  Person freund3;
  Person freund4;
  Person freund5;
  Person freund6;
  Person freund7;
  Person freund8;
  Person freund9;
 …
}
```

- What do we do when we want to store our Twiiter-followers? Add 100 persons?
- Songs of a music collection?

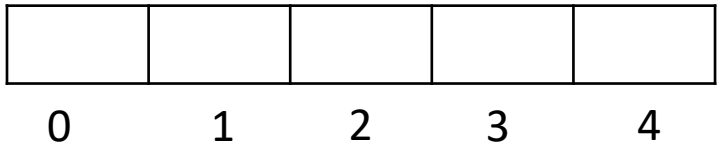
Arrays II

- Answer: **Array** as composed complex data structure
- Several values/objects of **the same type** will be summarized in **one variable**

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

Type Array Name Reserve memory for five int values

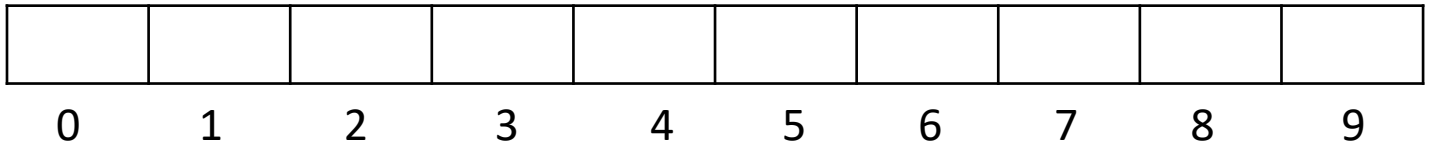
- Declaration via „[ ]“ and size
- **Index in arrays start with 0!**



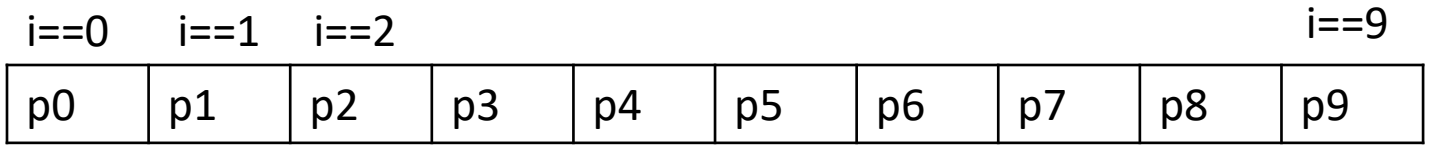
Arrays III

- When creating an object, **no object is initialized**
- Only memory is **reserved**!

```
Person[] friends = new Person[10];
```




```
for (int i = 0; i < friends.length; ++i) {
  friends[i] = new Person("Mein", "Freund");
}
```



Arrays IV

- Initialization

```
int[] fiveInts = new int[5];
int[] fourInts;
fourInts = new int[4];
```

Declaration

Initialization (Memory is reserved)

```
char[] c = new char[3];
c[0]= 'a'; c[1] = 'b'; c[2] = 'c';
char[] c = new char[] {'a', 'b', 'c'};
double[] d = {1.2, 3.5, 2.1};
```

- Size of an array via variablename.length

```
int[] fiveInts = new int[5];
int size = fiveInts.length; //returns 5
```

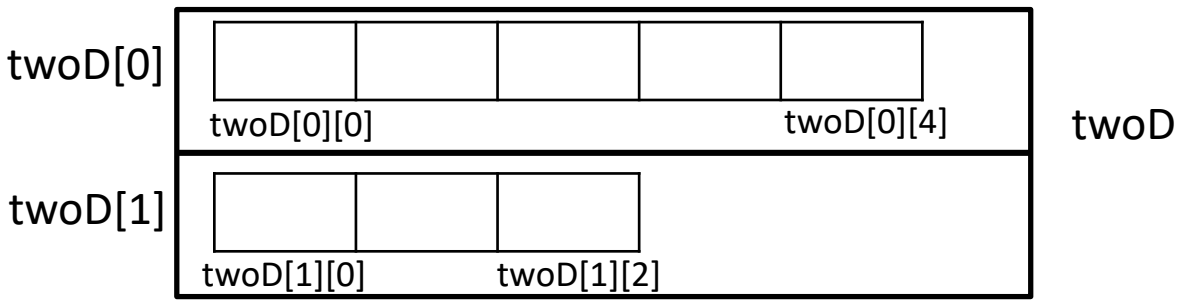
- Again: Access to array starting from 0 to size - 1

Multidimensional Arrays

- Arrays are stored in arrays (e.g. matrix)
- Declaration via additional „[ ]“

```
int [][] twoD = new int[2][];
twoD[0] = new int[5];
twoD[1] = new int[3];
```

Here, it is possible without defining the size, because not all rows need to have the same length




```
int [][] uniform = new int[5][8];
```
All rows have the same length

```
int [][] initWithElements = new int[][] {{2,4},{4,4,5,6,12}};
```

Some Background Information on Arrays

- Connection to Computer Architecture
  - Direct model of main memory
  - Elements in array correspond to consecutive memory cells
  - Number of elements is fixed

- Complications in Java
  - Only innermost dimension of arrays is stored in main memory
  - Java arrays are **not necessarily rectangular**
  - Careful when sub arrays have different size

Strings I

- String is an array of chars!
- So it is a complex data type (composed of primitive data types)

- String is initialized with "…"

```
String s = "Hello!";
```

H	e	l	l	o	!

```
char[] cs = {'H','e','l','l','o','!'};
```

Objects of class String

```
String s2 = new String("Hello");
String s3 = new String(cs);
```

Strings II

- Comparison of Strings happens with **equals**
- == returns **unexpected** result!

Result: Not Same! →

```
String h = new String("Hi");
String t = new String("Hi");
if(h == t){
  System.out.println("Same!");
}
else {
  System.out.println("Not Same!");
}
```

Result : Same! →

```
if(h.equals(t)){
  System.out.println("Same!");
}
else{
  System.out.println("Not Same!");
}
```

- Background on this: Next time (it has to do with objects being references) ☺

Quiz!!!

- Write a function to compute the mean of a one-dimensional array

```
static int average(int[] data){?}
```

- What would be printed out for this loop?

```
for(int i = 0; i < 10; i ++) {
  if (i % 3 == 0)
    continue;
  if (i % 2 == 0)
    System.out.print("*");
  else
    System.out.print("+");
}
```

- Where are the errors?

```
public int errors(int i) {
  boolean yesNo = new boolean[] {true, false,
        false, true};
  for(int i = 0; i <= yesNo; ++i) {
    yesNo = 1.0;
  }
}
```

Take Aways I

- An array is a simple data structure that stores several values of objects of the same type
  - Is initialized with size or directly with the data
  - Index starts with 0
  - With loops, we can iterate over all elements until length – 1
- Reminder:
  - Implement a static method in Java for the following operation:
    - **Input**: 2-dimensional array with **`int`** values
    - **Output**: An **`int`** value, representing the sum of the largest values of all columns
  - Example:

5	2	7	8
3	3	5	4

is 23, since: 5 + 3 + 7 + 8 = 23

Take Aways II

- A string is an array of chars
- Static methods and attributes can be used without instantiating objects

Coming Up Next

- Internal Memory Usage in Java (Heap vs. Stack)
- References in Java
- Ways to pass parameters to functions
- Cloning of objects