Lesson 1a and 1b

Mark Eric Dieckmann

January 16, 2022

1 Revision of standard classes and simple own class

We use Lesson 1a to revise concepts we learned in the lectures and to practice them in the form of small code components. The class **Lesson1a** will be used for this purpose (it is not available yet in this package). We will look at the following topics:

- Primitive and reference variables in the stack.
- Objects in the heap and their connection to reference variables.
- What is the difference of an instance and class variable? Do we need an instance of a class to have access to a class variable?
- What is the difference between an instance and class method.
- What purpose does *toString()* have?
- Static arrays.
- Illustrate the difference between a shallow and deep copy for instances of String and for an own class.

Reference variables, overloading. Address.

Objects, own small class, instance variables, class variables. instance and class methods. toString Shallow and deep copies. this. String. String arrays.

2 The complete class Human

We have introduced class diagrams in Lecture 4. Access modifiers public and private have the symbols + and -, respectively. Constants have uppercase letters. Class variables and methods are underlined. Variable and method names come first followed by the type. Argument types go in the method's parentheses.

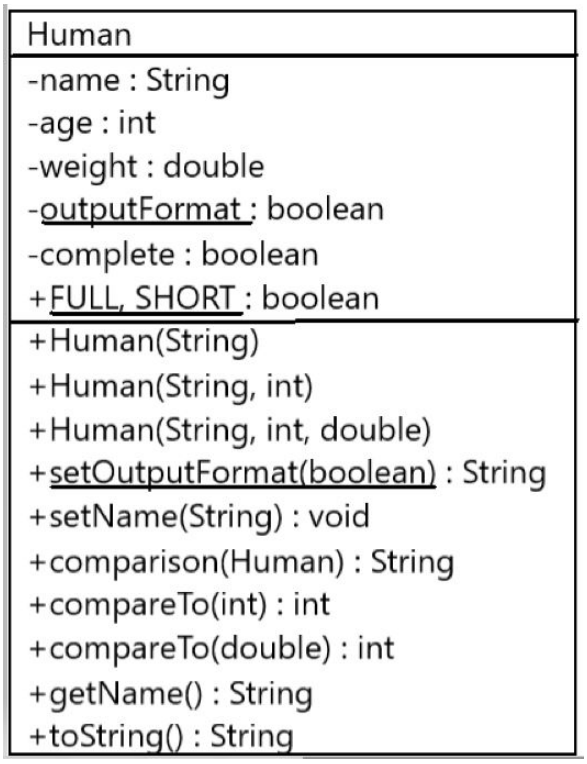
Our complete class **Human** has the following class diagram.

name, age, weight are the name, age and weight of a person. These instance variables are declared private. The instance variable *complete* is a boolean, which is true if all information has been initialized. We initialize it as *false* when we declare it.

outputFormat is a class variable: all instances of **Human** share its value. It is initialized to *false* where it is declared.

FULL=true and *SHORT=false* are class constants. They are initialized when they are declared.

Constructors are methods with the name of the class **Human** and without return value. They are only called when the object is initialized. The constructor is overloaded. The first initializes only *name*. The second initializes *name, age*. The third constructor initializes *name, age, weight* and sets *complete* to *true*.



setOutputFormat(arg) sets *outputFormat* to *arg*. It returns the string "Full format" if *arg=true* and "Short format" otherwise.

setName(arg) sets *name=arg*. Use this method to initialize *name* in the constructors.

comparison(Human) compares two instances of **Human** by their values for *name*, *age* and *weight*. It compares names with the *compareTo(String)* method of **String**. It compares the age and weight by the overloaded method *compareTo(arg)*, which is defined for int's and double's. These methods should return -1, 0 or 1 depending on how the ages and weights of both instances of **Human** compare. *comparison(Human)* should return a formatted string. It should start with a "(" followed by an int with two digits for the return value of the String comparison. This number should be followed by a comma and by a two digit int for the number we get from comparing *age*. Then there should be another comma and the result from the weight comparison followed by ")".

getName() returns the value of *name*.

toString() returns a string that depends on the available information. If *complete=false* or if *outputFormat* equals the value of the class constant *SHORT*, it should just return the string "Name: " followed by the value of "name". If *outputFormat* equals *FULL* and *complete* equals *true*, it should return a formatted string. It should start with "Name:" followed by a string with seven characters (aligned to the right). It should be followed by ", age:" and an integer with three digits. That should be followed by ", weight:" and a float with 5 digits and one after the radix point.

2.1 The class Lesson1b

The class **Lesson1b** contains the main method. We create several instances of **Human** to test the methods we implemented.

We create the human *first* with the name "Mark". The human *second* has the name "Hanna" and age 28. The human *third* has the name "Alice", the age 22 and the weight 52. *fourth* has the name "Connor", the age 24 and the weight 75.

We use *setOutputFormat* to the value of the class constant *SHORT*. We print the return values of the *toString()* methods of all four humans to the console.

Then we set the output format of *toString()* to the full format and write the return values of *toString()* of the four humans to the console.

Finally we write the return value of *comparison(arg)* to the console. We compare *third* and *fourth*, which have the complete information.