Advanced Programming

1 Hello World

As it is almost a tradition in many programming tutorials, we would like to start with a first "Hello World" application to show how a very simple Java source code looks like:

```
1 public class HelloWorld
  {
3     public static void main(String [ ] args)
      {
5         System.out.println("Hello World");
      }
7 }
```

2 Ticket Machine

In the lecture you have seen a "naïve ticket machine" example. Write a similar application with the following specification:

1. Provide a class `TicketMachine` with

- Fields:
  - `price` - The price of a single ticket
  - `balance` - How much money has been inserted
  - `total` - The total amount of money in the machine

All three fields should be **floating point** variables and declared as private.

- Methods:
  - `TicketMachine(...)` (constructor) - Initialize the fields. `balance` and `total` shall be 0, for `price` use a method argument as value.
  - `getPrice()` - Return value of `price`

- `getTotal()` - Return value of `total`
- `insertMoney(...)` - Check if the inserted amount is valid (> 0). Increase `balance` by the amount passed in the argument. As soon as `balance` $\geq$ `price`, print a ticket and return change. Use helper methods `printTicket` and `returnChange` to do so. Reset `balance`.
- `printTicket` - Write some information about the ticket, like name or price, on the screen.
- `returnChange` - Check if `balance` $>$ `price` and print the amount of change on the screen. Reduce `balance`.

Note that the helper methods `printTicket` and `returnChange` should be declared as `private`.

2. Provide another class `Application` which contains the Main-method. Create two instances of `TicketMachine`, each with a different price. Call the method `insertMoney` of both objects several times with different amounts of money. Finally, call `getTotal` of both objects and check if the output matches your expectations. Store the command line output to a text file and attach it to your source code.