Exercise 1

This exercise is meant to provide a first look at the `Finmath` library implementation for the simulation of a LIBOR Market Model $L_i := L(T_i, T_{i+1})$, $0 \le i \le n-1$, with

$$dL_i(t) = L_i(t)\sigma_i(t)dW_i(t), \quad 0 \le t \le T_i, \quad i = 0, \dots, n-1, \tag{1}$$

where $d\langle W_i, W_j \rangle(t) = \rho_{i,j}(t)dt$. You can find an already implemented class

`com.andreamazzon.handout7.LIBORMarketModelConstruction`,

with a method `createLIBORMarketModel` which returns an object of type

`net.finmath.montecarlo.interestrate.TermStructureMonteCarloSimulationModel`.

In our construction, object of a class implementing this latter interface is obtained by linking together an object of type `IndependentIncrements` (for example Brownian motion) with one of type `net.finmath.montecarlo.interestrate.LIBORMarketModel`. This can be done by using the constructor of the class

`net.finmath.montecarlo.process.EulerSchemeFromProcessModel`.

As you can see, the method `createLIBORMarketModel` is mainly devoted to construct an object of type `LIBORMarketModel` (in particular, using the constructor of the class `LIBORMarketModelStandard`). Have a look at the code and at the `Finmath` library classes which it involves, in order to get what is needed to implement the LIBOR market model. To have a further help understanding it, have a look also at the PDF file *Libor market model simulation* you find in `com.andreamazzon.handout7` (this will be commented during the exercise class). Note that in our case, the term $\sigma_i(t)$ in (1) is given by a volatility structure

$$\sigma_i(t) := (a + b(T_i - t)) \exp(-c(T_i - t)) + d, \quad t \ge 0, \quad i = 0, \dots, n-1,$$

$a, d \in \mathbb{R}$, $b, c > 0$. Moreover, we define a correlation

$$\rho_{i,j}(t) := \exp(-\alpha |T_i - T_j|), \quad t \ge 0, \quad i, j = 0, \dots, n-1,$$

$\alpha > 0$. Do then the following:

- Taking inspiration for example from `net.finmath.montecarlo.interestrate.products.Caplet`, write a class `myDigitalCaplet` implementing

  `net.finmath.montecarlo.interestrate.products.AbstractLIBORMonteCarloProduct`.

  The method `getValue`, taking as inputs the evaluation time and an object of type `TermStructureMonteCarloSimulationModel`, must in this case return the discounted payoff of a digital caplet with underlying $L(T_i, T_{i+1})$. The dates T_i and T_{i+1}, or one of those and the period length, must be given in the constructor of the class.

- Complete where needed (look at the capitalized instructions) the implementation of the class `LMMDigitalCapletTest`, that you find in `com.andreamazzon.handout7`, under tests.

  Here we use our `createLIBORMarketModel` method in order to construct and simulate a LIBOR Market Model with tenure structure

  $$T_0 = 0 < T_i = 0.5 < T_{i+1} = 1 < \cdots < T_{20} = 10,$$

  correlation decay parameter $\alpha = 0.5$, volatility parameters $a = 0.2$, $b = 0.1$, $c = 0.15$, $d = 0.3$, and initial forwards $L_i = 0.05$ (note that, thanks to the method

`ForwardCurveInterpolation.createForwardCurveFromForwards`, we don't have to provide all the initial forwards as the missing ones are interpolated).

In particular, for every T_i, we consider the digital caplet with underlying L_i, notional $N = 10000$, strike $K = 0.05$ and maturity T_i, and compare its Monte Carlo price to the analytical one. Here your duty is to compute the Monte Carlo prices.