Exercise 1

This exercise might be done during the Tutorium

Let $S^1 = (S^1_t)_{t\in[0,T]}$ and $S^2 = (S^2_t)_{t\in[0,T]}$ be two assets following the risk-neutral dynamics

$$dS^1_t = rS^1_t dt + \sigma_1 S^1_t dW^1_t, \quad 0 \le t \le T, \tag{1}$$
$$dS^2_t = rS^2_t dt + \sigma_2 S^2_t dW^2_t, \quad 0 \le t \le T, \tag{2}$$

for risk-free rate $r > 0$, constant volatilities $\sigma_1, \sigma_2 > 0$ and correlated Brownian motions $\langle W^1, W^2\rangle_t = \rho t$, $\rho \in [-1, 1]$.

An exchange option for S^1 to S^2 with maturity T is a product that pays $(S^1_T - S^2_T)^+$ at time T. It can be seen that the value at time 0 of the exchange option is

$$C^{BS}(r, S^1_0, S^2_0, \sigma, T), \tag{3}$$

where $\sigma = \sqrt{\sigma_1^2 - 2\rho\sigma_1\sigma_2 + \sigma_2^2}$ and $C^{BS}(r, S, K, \sigma, T)$ is the discounted Black Scholes value of a Call option of risk-free rate r, spot price S, strike K, volatility σ and maturity T.

Give the implementation of a class `ExchangeOption` that implements

`net.finmath.montecarlo.assetderivativevaluation.products.AbstractAssetMonteCarloProduct`.

In particular, the method

```
getValue(double evaluationTime, AssetModelMonteCarloSimulationModel model)
```

has to be implemented in such a way that returns the (discounted) payoff of an exchange option.

Hint: in order to give a general implementation, you can suppose the argument `model` of type `AssetModelMonteCarloSimulationModel` to represent an n-dimensional process, for general n, and that the processes in (1) and (2) are the i-th and j-th component of the process represented by `model`, identified by their index. Look at the methods of `AssetModelMonteCarloSimulationModel` that you can use for your scope.

Exercise 2

This exercise might be done during the Tutorium

Write a `JUnit` test class with the following three methods:

- A method which checks if the value of the exchange option with an underlying constructed with a seed at your choice approximates (3) up to a tolerance of 2%. The value in (3) should be computed analytically.
- A method which checks that, out of 500 computations of the value of an exchange option whose underlying is constructed with a random seed, the price approximates (3) up to a tolerance of 2% at least in the 90% of cases.
- A method where you check if the price of the exchange option increases or decreases with respect to the correlation ρ: what do you expect?

Use parameters' values of your choice, as long as you think they make sense.

Hint: the main point of this exercise is how to construct the object that you have to pass to `getValue`. A *possible* choice is to use a constructor of

`net.finmath.montecarlo.assetderivativevaluation.MonteCarloMultiAssetBlackScholesModel`.

In this case, you have to focus in particular on how to construct the object representing the Brownian motions driving the model and the correlation matrix.

Exercise 3

Implement a class `GeneralOption` which extends `AbstractAssetMonteCarloProduct`. In this class we want to compute the value of a European option whose payoff is a general function of the terminal value S_T of a one-dimensional underlying.

Write a `JUnit` test class where you construct an object of type `GeneralOption` specifying a payoff function $f(x) = (x - K)^+$ for a given value of K, you make it call the method `getValue` and test if the value you obtain is the same, up to a (very small) tolerance, as the one you get when you value the option by using the class

`net.finmath.montecarlo.assetderivativevaluation.products.EuropeanOption`.

for the same underlying.