Exercise 1

This exercise might be done during the Tutorium

Write two classes implementing `AbstractAssetMonteCarloProduct` whose method

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

returns the Monte-Carlo approximation of the *Delta* of a European call option written on a Black-Scholes model represented by `underlyingSimulation`, by the Pathwise Differentiation Method and the Likelihood Ratio Method, respectively.

Compare the errors you get when approximating the *Delta* of the option using the two classes above and the one you have written for Exercise 1 Handout 12.

Hints: you can find a description of the Pathwise and Likelihood ratio methods from pages 530 and 537 of the script, respectively.

Above, I mentioned explicitly that now you have to take into consideration a Black-Scholes model. This is one of the major drawbacks of approximating the derivative with the Pathwise Differentiation and with the Likelihood Ratio methods: they are model dependent. So, writing the implementation of `getValue`, you should check that `underlyingSimulation` represents the simulation of a Black-Scholes model. For example, you can check if the object returned by letting call `getModel()` by `underlyingSimulation` (note: you might have to downcast it before doing that!) is of type `BlackScholesModel`.

Exercise 2

The *Vega* of a European call option is the option's price sensitivity with respect to changes in the volatility. Suppose that the option is written with respect to an underlying that follows a Black-Scholes model. Derive the calculation of the *Vega* of the European call option by using the Pathwise method described at page 530 of the script, just writing down the computations. In our case, θ is of course the volatility σ.