Dr. Roxana Dumitrescu
Department of Mathematics
King's College London
email: roxana.dumitrescu@kcl.ac.uk

Exercises C++
Sheet 6

Binomial pricer

Problem 1. Write a **Binomial Pricer** which computes the prices of Calls and Puts of European and American style in the Binomial Model. See the lecture for details concerning the design in C++. The main ideas are briefly recalled below:

Design:

1. Write a class **BinModel** which contains

- The initial price $S0$, U, D and R as **private** member variables;
- The **public** functions **SetR**, **SetD**, **SetU**, **SetStock**, **GetR**, **GetD**, **GetU**, **GetStock**, **CheckData** and **RiskNeutProb**.

2. Write a class **Option** which contains as **private** member variable the expiry date N and as **public functions**:

- Functions **GetN** and **SetN**;
- A pure virtual function **payoff**;
- A virtual destructor;

3. Write a class **EurOption** derived from **Option** (with virtual inheritance) which contains as public functions:

- A function **PriceByCRR(const BinModel &)** which computes the price of an option of European type at time 0;
- A destructor;

4. Write a class **AmOption** derived from **Option** (with virtual inheritance) which contains as public functions:

- A function **PriceBySnell(const BinModel &)** which computes the price of an option of American type at time 0;

- A destructor;

5. Write a class **Call** derived from both the **EurOption** and the **AmOption** classes which contains:

- A **private** member variable K representing the strike;
- Public functions: **SetK**, **GetK** and the function **payoff**;
- A destructor;

6. Write a class **Put** derived from both the **EurOption** and the **AmOption** which contains as public functions:

- A **private** member variable K representing the strike;
- Public functions: **SetK**, **GetK** and the function **payoff**;
- A destructor;

Problem 2.[**Approximation of the Black-Sholes model**] The binomial model can be employed to approximate the Black-Scholes model. One of several possible approximation schemes is the following. Divide the time interval $[0, T]$ into N steps of length $h = \frac{T}{n}$, and set the parameters of the binomial model to be

$$U = e^{rh+\sigma\sqrt{h}} - 1, \tag{0.1}$$

$$D = e^{rh-\sigma\sqrt{h}} - 1, \tag{0.2}$$

$$R = e^{rh} - 1, \tag{0.3}$$

where σ is the volatility and r is the continously compunded interest rate in the Black-Sholes model.
Develop code to compute the approximate price for an European call option, European put option, American put option and American call option in the Black-Sholes model by means of this binomial tree approximation. How would you test your results?

Problem 3. Add the ability to price the options **Digital call**, **Digital put**, **Double Digital Option**, **Bull spread**, **Bear spread**, **Strangle** and **Butterfly** by introducing new subclasses **DigitCall**, **DigitPut**, **DoubDigitOpt**, **BullSpread**, **BearSpread**, **Strangle** and **Butterfly** derived from the **EurOption** class. Place any new class in a separate .cpp file, complete with its own header file. The payoffs of the above options are recalled below:

- Digital call:

$$h^{\text{digitcall}}(z) = \begin{cases} 1 & \text{if } K < z \\ 0 & \text{otherwise.} \end{cases} \tag{0.4}$$

- Digital put:

$$h^{\text{digitput}}(z) = \begin{cases} 1 & \text{if } K > z, \\ 0 & \text{otherwise.} \end{cases} \tag{0.5}$$

- Double Digital Option:

$$h^{\text{DoubDigitOpt}}(z) = \begin{cases} 0 & \text{if } z \leq K_1, \\ 1 & \text{if } K_1 < z < K_2, \\ 0 & \text{if } z \geq K_2. \end{cases} \tag{0.6}$$

- Bull spread:

$$h^{\text{BullSpread}}(z) = \begin{cases} 0 & \text{if } z \leq K_1, \\ z - K_1 & \text{if } K_1 < z < K_2, \\ K_2 - K_1, & \text{if } K_2 \leq z. \end{cases} \tag{0.7}$$

- Bear spread:

$$h^{\text{BearSpread}}(z) = \begin{cases} K_2 - K_1 & \text{if } z \leq K_1, \\ K_2 - z & \text{if } K_1 < z < K_2, \\ 0, & \text{if } K_2 \leq z. \end{cases} \tag{0.8}$$

- Strangle:

$$h^{\text{Strangle}}(z) = \begin{cases} K_1 - z & \text{if } z \leq K_1, \\ 0 & \text{if } K_1 < z \leq K_2, \\ z - K_2, & \text{if } K_2 < z. \end{cases} \tag{0.9}$$

- Butterfly:

$$h^{\text{Butterfly}}(z) = \begin{cases} z - K_1 & \text{if } K_1 < z \leq \dfrac{K_1 + K_2}{2}, \\ K_2 - z & \text{if } \dfrac{K_1 + K_2}{2} < z \leq K_2, \\ 0, & \text{otherwise .} \end{cases} \tag{0.10}$$

Problem 3. Add a function **PriceByCRR** (to the class **EurOption**) to compute at any time step the price and the replicating strategy for an European option in the binomial tree, using the **BinLattice**<> class template (more precisely, this is used in order to store the stock and money market account positions in the replicating strategy at the nodes of the binomial tree, as well as the price). Note that the function **PriceByCRR** has to be overloaded (more precisely, we have one function which only computes the price at time 0 and another one which computes the price and the replicating strategy at any time step n and node i).
The portfolio belonging to the **replicating strategy** created at time $n-1$, node i and held during the n th time step, that is, until time n, consists of stock and money market account positions

$$x(n,i) = \frac{H(n,i+1) - H(n,i)}{S(n+1,i) - S(n,i)}, \tag{0.11}$$

$$y(n,i) = \frac{H(n-1,i) - x(n,i)S(n-1,i)}{1+R}, \tag{0.12}$$

for $n = 1, 2, ..., N$ and $i = 0, 1, ..., n-1$ where $S(n,i)$ and $H(n,i)$ denote the stock and option prices at time n, node i.

Problem 4. Add a function **PriceBySnell** (to the class **AmOption**) to compute at any time step n and node i the price and the stopping strategy for an American option in the binomial tree, using the **BinLattice**<> class template (more precisely, this is used in order to store the price, as well as the stopping strategy). Note that the function **PriceBySnell** is overloaded (see the lecture for more details).