(1)Black-Scholes : **Option_BlackScholes.py**

```
def bs(S0, K, T, r, q, sigma, call_put):...

# main
S0 = 50
K = 50
r = 0.1
q = 0.05
sigma = 0.4
T = 0.5

testCall = bs(S0, K, T, r, q, sigma, "call")
testPut = bs(S0, K, T, r, q, sigma, "put")
print('=======================================')
print(testCall)
print(testPut)
```

直接調整 **#main** 下面的變數，呼叫函式存入變數，再印出即可(所有參數以及函式都已預先輸入好，直接執行即可以看到精美的結果)。

(2)Monte-Carlo Simulation : **Option_MonteCarlo2.py**

```
def monte_carlo_European(S0, K, T, r, q, sigma, call_put, sims, rep):..

# main
S0 = 50
K = 50
r = 0.1
q = 0.05
sigma = 0.4
T = 0.5
sims = 10000
rep = 20
monte_carlo_European(S0, K, T, r, q, sigma, "call", sims, rep)
monte_carlo_European(S0, K, T, r, q, sigma, "put", sims, rep)
```

直接調整 **#main** 下面的變數，再呼叫函式即可(所有參數都以及函式已預先輸入好，直接執行即可以看到精美的結果)。

(3)CRR Binomial Tree : **Option_BinomialTree.py**

***Bonus2已包含在 Option_BinomialTree.py 中**

```
from math import log, sqrt, exp
# from scipy.stats import binom
def binomial_prob(n, j, p):...
```

```
def binomial_European(S0, K, T, r, q, sigma, layers, call_put):...
def binomial_American(S0, K, T, r, q, sigma, layers, call_put):...
def combinatorial_European(S0, K, T, r, q, sigma, layers, call_put):...

# main
S0 = 50
K = 50
r = 0.1
q = 0.05
sigma = 0.4
T = 0.5

# n = 100 (layers = 100)
layers = 100
# European Call
...
```

直接調整 **#main** 下面的變數，再呼叫函式即可(所有測資(不同的n)、參數以及函式都已預先輸入好，直接執行即可以看到相當精美的結果)。

直接執行，輸出看起來是這樣 ：

```
============================================================
n = 100
(CRR Binomial Tree) Price of European call : 6.026046
(CRR Binomial Tree) Price of American call : 6.026232
(CRR Binomial Tree) Price of European put : 4.822022
(CRR Binomial Tree) Price of American put : 4.97949
============================================================
n = 500
...
```

關於Bonus2 ：

直接呼叫 **Option_BinomialTree.py** 中
combinatorial_European(S0, K, T, r, q, sigma, layers, call_put)
即可 (已經幫老師/助教呼叫好，直接執行即可)。
最後，由**CRR Binomial Tree**與**Monte-Carlo Simulation**所計算出來的價格都已四捨五入至第六位