i1black_scholes.py | i0binomial_treesII.py | i0binomial_trees.py

```python
import math


def EuropeanBSTree(K, T, S, sig, r, N, PorC):
    dt = T / N
    nu = r - 0.5 * sig * sig
    x = math.sqrt(sig * sig * dt + (nu * dt) ** 2)
    dxu = math.exp(x)
    dxd = 1 / dxu
    pu = 0.5 + 0.5 * (nu * dt / x)
    pd = 1 - pu
    disc = math.exp(-r * dt)

    St = [0] * (N + 1)
    C = [0] * (N + 1)

    St[0] = S * dxd ** N

    for j in range(1, N + 1):
        St[j] = St[j - 1] * dxu / dxd

    for j in range(1, N + 1):
        if PorC == "put":
            C[j] = max(K - St[j], 0)
        elif PorC == "call":
            C[j] = max(St[j] - K, 0)

    for i in range(N, 0, -1):
        for j in range(0, i):
            C[j] = disc * (pu * C[j + 1] + pd * C[j])
        print("-" * 10, "\n", C)
    return C[0]
```

```python
def AmericanBSTree(K, T, S, sig, r, N, PorC):
    dt = T / N
    dxu = math.exp(sig * math.sqrt(dt))
    dxd = math.exp(-sig * math.sqrt(dt))
    pu = ((math.exp(r * dt)) - dxd) / (dxu - dxd)
    pd = 1 - pu
    disc = math.exp(-r * dt)

    St = [0] * (N + 1)
    C = [0] * (N + 1)

    St[0] = S * dxd ** N

    for j in range(1, N + 1):
        St[j] = St[j - 1] * dxu / dxd

    for j in range(1, N + 1):
        if PorC == "put":
            C[j] = max(K - St[j], 0)
        elif PorC == "call":
            C[j] = max(St[j] - K, 0)

    for i in range(N, 0, -1):
        for j in range(0, i):
            C[j] = disc * (pu * C[j + 1] + pd * C[j])
        print("-" * 10, "\n", C)
    return C[0]


if __name__ == "__main__":
    price = EuropeanBSTree(60, 10, 70, 0.2, 0.05, 10, "call")
    print("\n EuropeanBSTree call price", format(price, ".2f"))

    price = AmericanBSTree(60, 10, 70, 0.2, 0.05, 10, "call")
    print("\n AmericanBSTree call price", format(price, ".2f"))

    price = EuropeanBSTree(95, 5, 100, 0.1, 0.04, 10, "put")
    print("\n EuropeanBSTree put price", format(price, ".2f"))

    price = AmericanBSTree(95, 5, 100, 0.1, 0.04, 10, "put")
    print("\n AmericanBSTree put price", format(price, ".2f"))

'''''
```

i1black_scholes.py | i0binomial_treesII.py

```python
import math


# math erf implementation
def my_fnor_func(x):
    y = 0.5 * (1 + math.erf(x / math.sqrt(2)))
    return y


def Black_Scholes(t, St, K, T, r, sig, PorC):
    Tmt = T - t
    ATmt = sig * math.sqrt(Tmt)
    logo = math.log(St / K)
    Ap = (logo + (r + 0.5 * sig ** 2) * Tmt) / ATmt
    An = Ap - ATmt

    if PorC == "call":
        p = St * my_fnor_func(Ap) - K * math.exp(-r * Tmt) * my_fnor_func(An)
    elif PorC == "put":
        p = K * math.exp(-r * Tmt) * my_fnor_func(-An) - St * my_fnor_func(-Ap)
    return p


if __name__ == "__main__":
    price = Black_Scholes(0, 70, 60, 10, 0.05, 0.2, "call")
    print(
        "1. European call option (S0 = 70, K = 60, T = 10, r = 0.05, σ = 0.2, N = 10)"
    )
    print("Black Scholes call price", format(price, ".2f"))

    price = Black_Scholes(0, 100, 95, 5, 0.04, 0.1, "put")
    print("3. European put option (S0 = 100, K = 95, T = 5, r = 0.04, σ = 0.1, N = 10)")
    print("Black Scholes put price", format(price, ".2f"))

"""
output
1. European call option (S0 = 70, K = 60, T = 10, r = 0.05, σ = 0.2, N = 10)
Black Scholes call price 36.02
3. European put option (S0 = 100, K = 95, T = 5, r = 0.04, σ = 0.1, N = 10)
Black Scholes put price 1.29

"""
```