# M86 Homework 1

## Alex Craig & Ganqi Li

## Exercise 1

### Wilmott Ch. 1 Questions

**Question 1.1** A company makes a three-for-one stock split. What effect does this have on the share price?

**Answer** The share price will decrease by a factor of three. So if the share price was $90 before the split, it will be $30 after the split. The total value of the shares will remain the same.

**Question 1.2** A company whose stock price is currently $S$ pays out a dividend $DS$, where $0 \leq D \leq 1$. What is the price of the stock just after the dividend date?

**Answer** If the stock price is $100 and the dividend is 0.01, the price of the stock after the dividend date will be $S - DS = \$100 - \$100 \times 0.01 = \$99$.

**Question 1.3** The dollar sterling exchange rate (colloquially known as 'cable') is 1.83, £1 = $1.83. The sterling euro exchange rate is 1.41, £1 = €1.41. The dollar euro exchange rate is 0.77, $1 = €0.77. Is there an arbitrage, and if so, how does it work?

**Answer** Let us test if there is an arbitrage opportunity. Let's start with $1.

$\$1 \to £\frac{1}{1.83} \to €\frac{1.41}{1.83} \to \$\frac{1.41}{1.83 \times 0.77} = \$\frac{1.41}{1.4091} = \$1.0006387$

So we have made a profit of $0.0006387. Very small, but still an arbitrage opportunity. The exchange fees would probably eat up the profit, but if we had a large amount of money, we might make a profit.

Another way to represent the conversion is:

$\$1 \times \frac{£1}{\$1.83} \times \frac{€1.41}{£1} \times \frac{\$1}{€0.77} = \$1.0006387$

**Question 1.5** A spot exchange rate is currently 2.350. The one-month forward is 2.362. What is the one-month interest rate assuming there is no arbitrage?

**Answer** The one-month interest rate is $\frac{2.362-2.350}{2.350} = 0.005106383$ or $0.5106383\%$.

Another way to calculate is the following:

$$2.362 = 2.350 \times e^r \Rightarrow r = \ln(\frac{2.362}{2.350}) = 0.0050933896191$$

This gives us approximately the same answer. The difference is due to one being a continuous rate and the other being a discrete rate.

**Question 1.6 (Dropped from HW 1)** A particular forward contract costs nothing to enter into at time $t$ and obliges the holder to buy the asset for an amount $F$ at expiry $T$. The asset pays a dividend $DS$ at time $t_d$, where $0 \leq D \leq 1$ and $t \leq t_d \leq T$. Use an arbitrage argument to find the forward price $F(t)$.

Hint: Consider the point of view of the writer of the contract when the dividend is reinvested immediately in the asset.

**Answer** The value of the dividend including interest rate can be represented as $DS(t_d) \times e^{r(T-t_d)}$. For no-arbitrage, the party entering the forward must be compensated for this opportunity cost, so the forward price will be the interest-adjusted price of the underlying stock minus the interest-adjusted price of the dividend, or $F(t) = S(t) \times e^{r(T-t)} - DS(t_d) \times e^{r(T-t_d)}$.

Alternatively, if the party buying the contract holds $\frac{1}{1+D}$ share of the asset, they will get $\frac{1}{1+D} \cdot D$ share at time $t_d$ if they invest their dividend immediately into the asset. (This assumes $S(t_d)$ represents the dividend-adjusted price of the asset.) In total, they will have $\frac{1}{1+D} + \frac{D}{1+D} = 1$ share of the asset at $T$. Since a $\frac{1}{1+D}$ share of the underlying achieves the result of a forward contract for 1 share of the asset at time $T$, the value of the forward can also be $F(t) = \frac{1}{1+D} \cdot S(t) \cdot e^{r(T-t)}$.

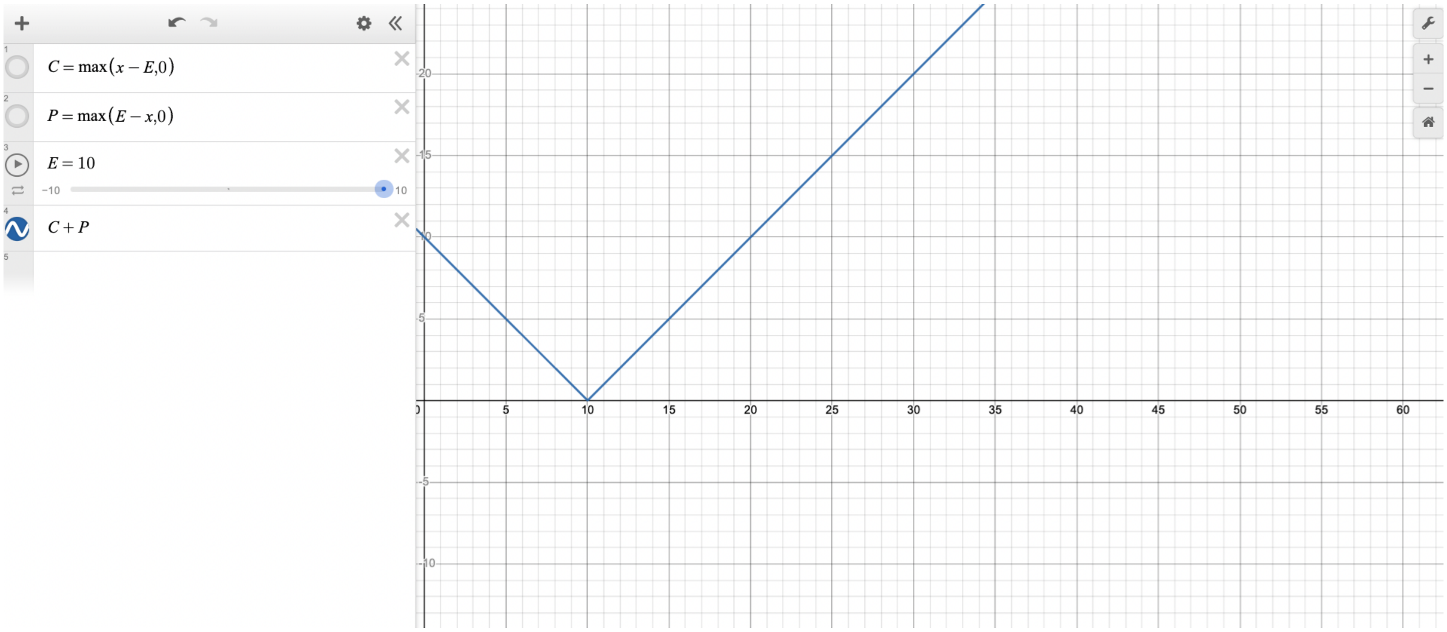
### Exercise 2

**Wilmott Ch. 2 Questions**

**Question 2.1** Find the value of the following portfolios of options at expiry, as a function of the share price:

**2.1b** Long one call and one put, both with exercise price $E$.

**Answer** This is a straddle. The value of the call is $C = \max(S - E, 0)$ and the value of the put is $P = \max(E - S, 0)$. The value of the portfolio is $C + P = \max(S - E, 0) + \max(E - S, 0) = \max(S - E, E - S, 0) = |S - E|$.



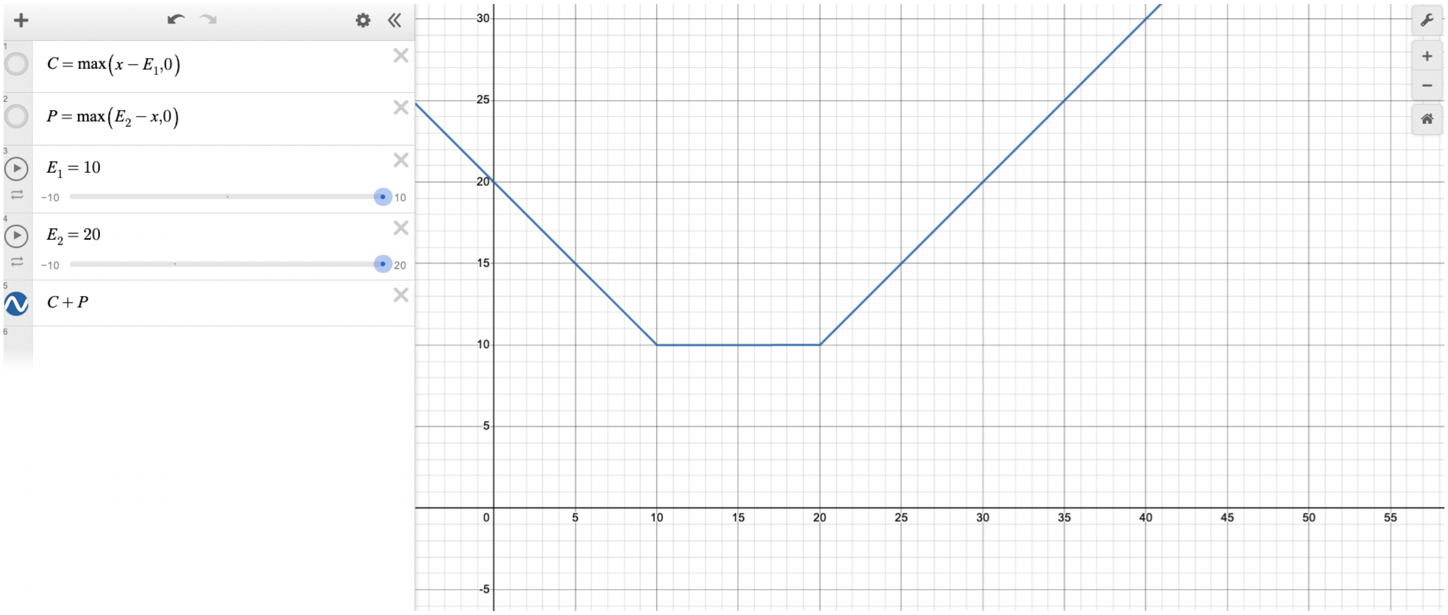

**2.1c** Long one call, exercise price $E_1$, short one call, exercise price $E_2$, where $E_1 < E_2$.

**Answer** This is a bull spread. The value of the portfolio is $C_1 - C_2 = \max(S - E_1, 0) - \max(S - E_2, 0)$.

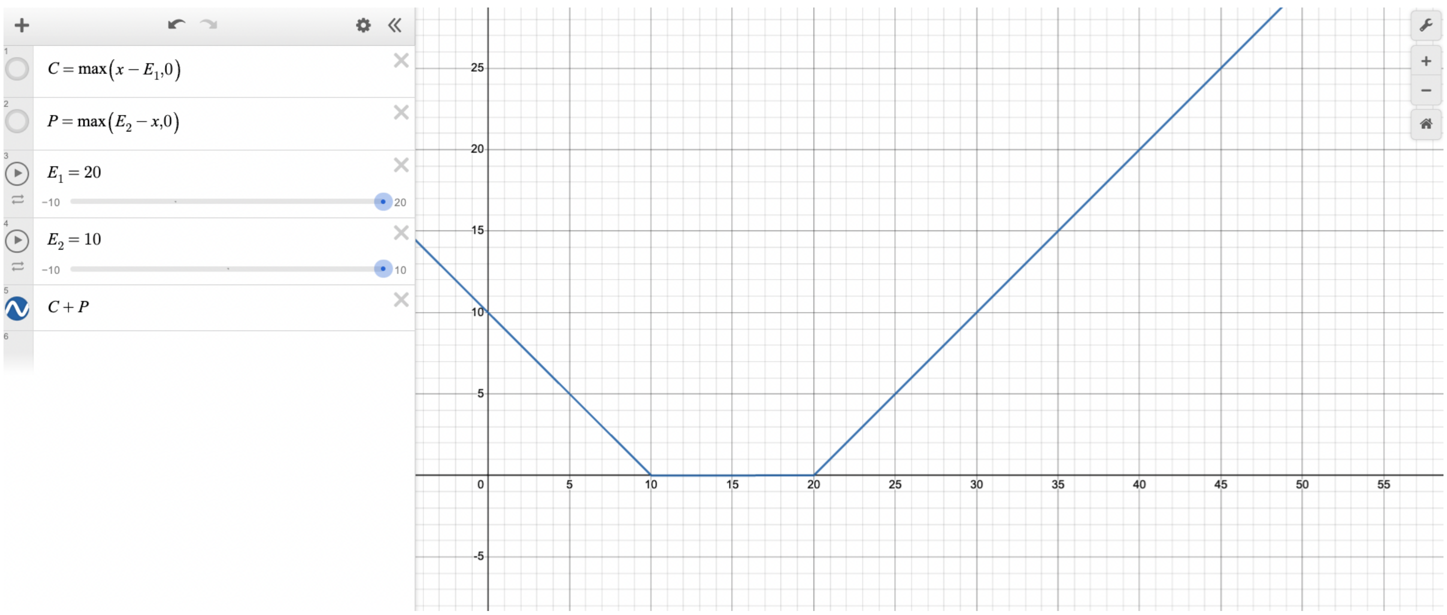

**2.1d** Long one call at exercise price $E_1$, long one put at exercise price $E_2$ . There are three cases to consider.

**Answer** In all three cases, the value of the portfolio is $C + P = \max(S - E_1, 0) + \max(E_2 - S, 0)$.

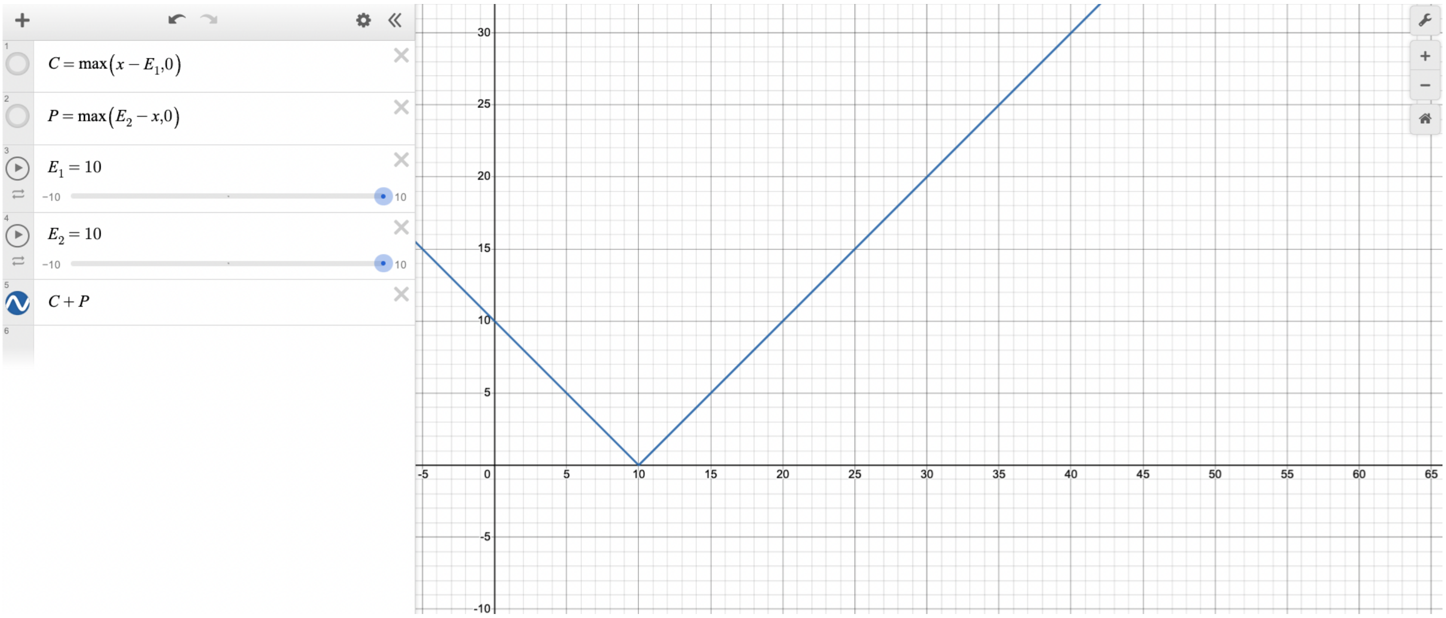
Case 1: $E_1 < E_2$




Case 2: $E_1 > E_2$

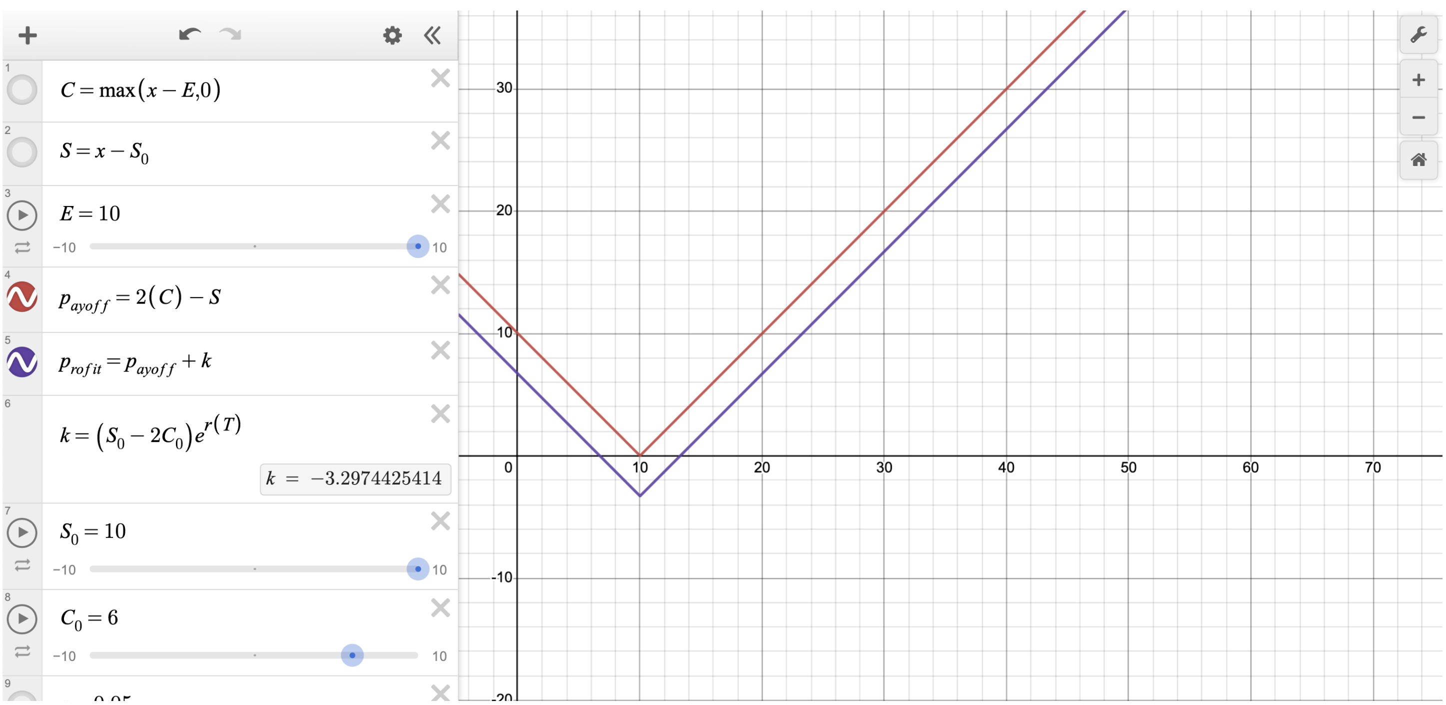



Case 3: $E_1 = E_2$




**Question 2.2** What is the difference between a payoff diagram and a profit diagram? Illustrate with a portfolio of short one share, long two calls with exercise price $E$.

**Answer** The payoff diagram is simply the payoff of being short one share and long two calls. The profit diagram takes into account the interest-adjusted value of the cash from selling the share at $t = 0$ and the value of buying two calls at $t = 0$. In the illustration, we used strike price $E = 10$, initial stock from $S_0 = 10$, call price $C_0 = 6$, interest rate $r = 0.05$, and time to maturity $T = 10$. The red line is the payoff and the purple line is the profit.




**Question 2.3** A share currently trades at \$60. A European call with exercise price \$58 and expiry in three months trades at \$3. The three month default-free discount rate is 5%. A put is offered on the market, with exercise price \$58 and expiry in three months, for \$1.50. Do any arbitrage opportunities now exist? If there is a possible arbitrage, then construct a portfolio that will take advantage of it. (This is an application of put-call parity.)

**Answer** Under the **put-call parity**, if we long a share, long a put and short a call, we can simulate the performance of cash plus interest that has an expected value of the strike price. So $S(t) + P(t) - C(t) = PV_t(E)$, where $E$ is the strike price, $S$ is the stock, $P$ is the put, $C$ is the call, and $PV_t$ is the discounted value of the strike price at time $t$.

On the left-hand-side, $S(t) + P(t) - C(t) = 60 + 1.5 - 3 = \$58.5$. On the right-hand-side, given $r = 0.05$, we have $PV_t(E) = PV_t(58) = 58 \cdot \frac{1}{e^{0.05}} \approx \$55.17$. Since $\$58.5 > \$55.17$, an arbitrage opportunity exists because the LHS is overpriced.

We can carry out a **"reverse conversion"** and take advantage of the arbitrage if we short a share and a put, and long a call.

**Question 2.4** A three-month, 80 strike, European call option is worth \$11.91. The 90 call is \$4.52 and the 100 call is \$1.03. How much is the butterfly spread?

**Answer** We can construct a butterfly spread if we buy an 80 and a 100 call, and write two 90 calls. So the price will be $11.91 + 1.03 - 2 \cdot 4.52 = \$3.9$

## Exercise 3

Question 3 from your document is as follows:

In lecture, we showed how to calibrate a binomial tree using parameters:

$$u = e^{\sqrt{\sigma \Delta t}}$$
$$d = e^{-\sqrt{\sigma \Delta t}}$$
$$p = \frac{1}{2} + \frac{\mu\sqrt{\Delta t}}{2\sigma}$$

where $\sigma$ is volatility and $\mu$ is the drift of $\log S(t)$ (cf. Van Erp lecture 1). Wilmott uses instead:

$$u = 1 + \sqrt{\sigma \Delta t}$$
$$d = 1 - \sqrt{\sigma \Delta t}$$
$$p = \frac{1}{2} + \frac{\alpha\sqrt{\Delta t}}{2\sigma}$$

where $\alpha$ is the mean rate-of-return i.e. \$E(S(T)) = S_0 e^{\alpha T} \$.

(a) Show that with Wilmott's parameters we get \$ E(S_1) \approx S_0 e^{\alpha \Delta t} \$

(b) Show that if we do not assume ud = 1 then we must require \$ \mu \Delta t = p \log u + (1 - p) \log d \$ and \$ \sigma \Delta t = (\log u - \log d) \sqrt{p - p^2}\$

(c) Show that with Wilmott's parameters we have approximately \$ \mu \Delta t \approx p \log u + (1 - p) \log d \$ and \$ \sigma \Delta t \approx (\log u - \log d) \sqrt{p - p^2} \$

Hint: Use a Taylor expansion to approximate log u and log d.

### (a)

$E(S_1)$ is the expected value of the stock price at time $t = \Delta t$. We can calculate this as follows:

$$E(S_1) = S_0 \cdot p \cdot u + S_0 \cdot (1 - p) \cdot d$$
$$= S_0((\frac{1}{2} + \frac{\alpha\sqrt{\Delta t}}{2\sigma})(1 + \sigma\sqrt{\Delta t}) + (\frac{1}{2} - \frac{\alpha\sqrt{\Delta t}}{2\sigma})(1 - \sigma\sqrt{\Delta t}))$$

Let $a = \frac{1}{2}$, $b = \frac{\alpha\sqrt{\Delta t}}{2\sigma}$, $c = 1$, and $d = \sigma\sqrt{\Delta t}$.

We then have a polynomial of the form $(a+b)(c+d)+(a-b)(c-d)$. Expanding this out, we get $ac+ad+bc+bd+ac-ad-bc+bd = 2ac + 2bd = 2 \cdot \frac{1}{2} \cdot 1 + 2 \cdot \frac{\alpha\sqrt{\Delta t}}{2\sigma} \cdot \sigma\sqrt{\Delta t} = 1 + \alpha\Delta t$. For small $\Delta t$, this is approximately $e^{\alpha \Delta t}$.

So we have $E(S_1) = S_0(1 + \alpha\Delta t) \approx S_0 e^{\alpha \Delta t}$.

**(b)**

Because $\mu$ is the drift of $\log S(t)$, by definition $\mu\Delta t$ is the expected change in $\log S(t)$ over the time period $\Delta t$. We can calculate this as follows:

$$E(\log S(\Delta t)) = \mu\Delta t = p\log u + (1-p)\log d$$

We took log of $u$ and $d$ because we are dealing with $\log S(t)$ and we know that $E(S(\Delta t)) = pu + (1-p)d$

We can also calculate the variance of $\log S(t)$ over the time period $\Delta t$ as follows:

$$\begin{gathered}
\mathrm{Var}(\log S(\Delta t)) = \sigma^2\Delta t = E((\log S(\Delta t))^2) - (E(\log S(\Delta t)))^2 \\
= (p\log(u)^2 + (1-p)\log(d)^2) - (p\log(u) + (1-p)\log(d))^2 \\
= p\log(u)^2 + (1-p)\log(d)^2 - (p^2\log(u)^2 + (1-p)^2\log(d)^2 + 2p(1-p)\log(u)\log(d)) \\
= p(1-p)\log(u)^2 + (1-p)(p)\log(d)^2 - 2p(1-p)\log(u)\log(d) \\
= p(1-p)(\log(u)^2 + \log(d)^2 - 2\log(u)\log(d)) \\
= p(1-p)(\log(u) - \log(d))^2
\end{gathered}$$

Thus, it follows that the standard deviation of the return to one time-step is:

$$\mathrm{SD}(\log(S(\Delta t))) = \sigma\sqrt{\Delta t} = \sqrt{p-p^2}(\log(u) - \log(d))$$

**(c)**

Let us consider the function $f(x) = \log(1+x)$. The Taylor series expansion of $f(x)$ around $x = 0$ is given by:

$$f(x) = f(0) + f'(0)x + \frac{f''(0)x^2}{2!} + \frac{f'''(0)x^3}{3!} + \cdots$$

For the logarithmic function $\log(1+x)$, the first derivative at 0 is 1, and higher-order derivatives at 0 are increasingly smaller. Therefore, for small values of $x$, the Taylor series can be approximated by just the first term:

$$\log(1+x) \approx x$$

Applying this to Wilmott's parameters of $u = 1 + \sqrt{\sigma\Delta t}$ and $d = 1 - \sqrt{\sigma\Delta t}$, we get:

- $\log u = \log(1 + \sqrt{\sigma\Delta t}) \approx \sqrt{\sigma\Delta t}$
- $\log d = \log(1 - \sqrt{\sigma\Delta t}) \approx -\sqrt{\sigma\Delta t}$

This approximation assumes $\sqrt{\sigma\Delta t}$ is small, which is true if $\Delta t$ represents a short time period.

Using the parameters from class, we have:

- $\log u = \log(e^{\sqrt{\sigma\Delta t}}) = \sqrt{\sigma\Delta t}$
- $\log d = \log(e^{-\sqrt{\sigma\Delta t}}) = -\sqrt{\sigma\Delta t}$

Therefore, the only major difference between the two parameters in the value of $p\log u + (1-p)\log d$ and $(\log u - \log d)\sqrt{p-p^2}$ is the value of $p$.

However, for small $\Delta t$ we have:

$$\frac{\mu\sqrt{\Delta t}}{2\sigma} \approx \frac{\alpha\sqrt{\Delta t}}{2\sigma}$$

Therefore, for small $\Delta t$, we have:

$$p = \frac{1}{2} + \frac{\mu\sqrt{\Delta t}}{2\sigma} \approx \frac{1}{2} + \frac{\alpha\sqrt{\Delta t}}{2\sigma}$$

Therefore, for small $\Delta t$ and with Wilmott's parameters, we have:

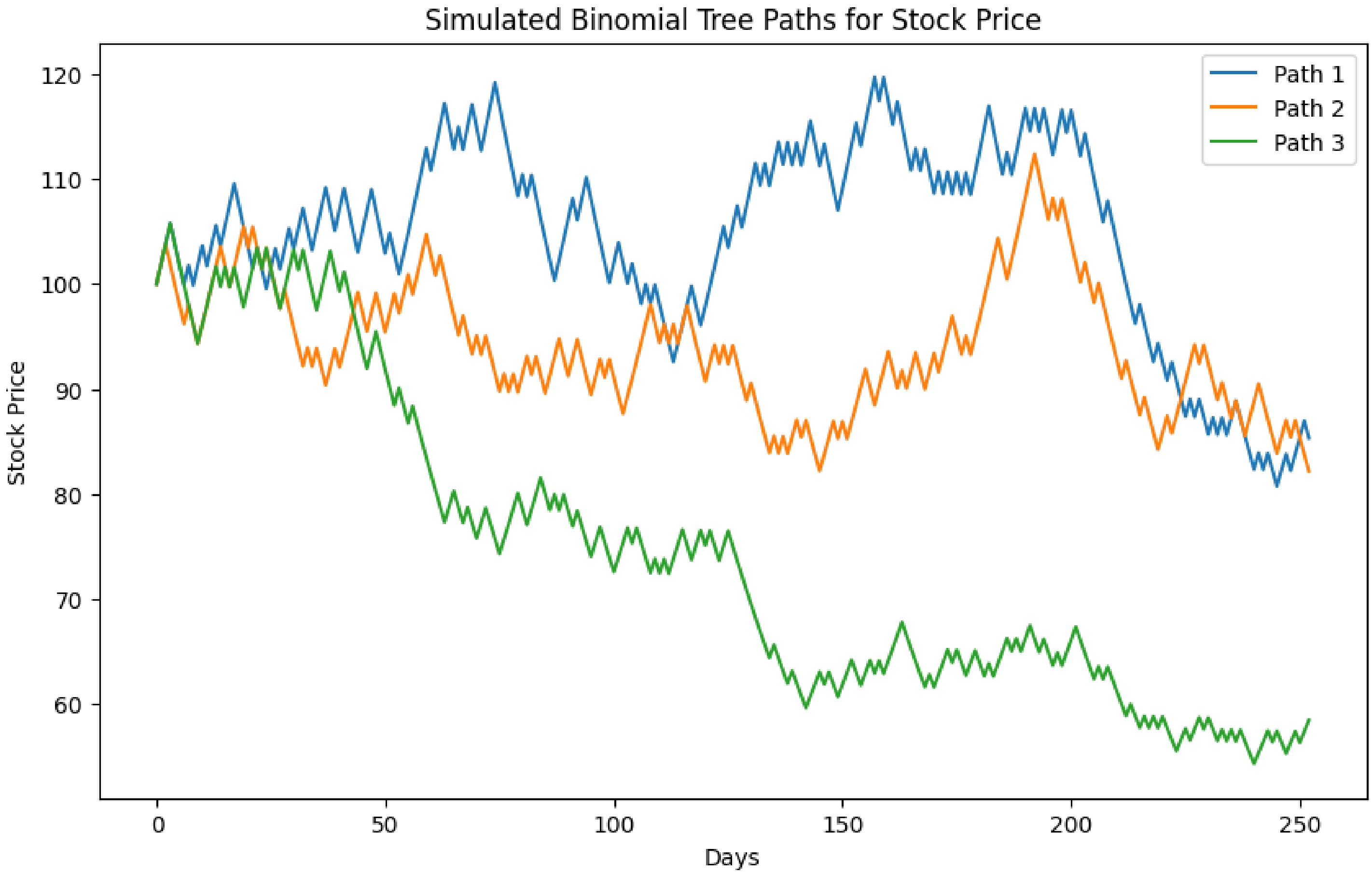
$$\mu\Delta t \approx p\log u + (1-p)\log d, \quad \sigma\sqrt{\Delta t} \approx (\log u - \log d)\sqrt{p-p^2}$$

## Exercise 4

Suppose a stock has annualized mean rate-of-return 10% and volatility 30%. In Python, implement the binomial tree model to simulate one year of stock price movements. Use $N = 252$ time-steps, i.e. one time-step is $\Delta t = 1$ day. Run a simulation, and create a plot of the daily prices $S_0, S_1, S_2, \ldots, S_N$ showing three random paths.

### Answer

```python
import numpy as np
import matplotlib.pyplot as plt

# Parameters
mu = 0.10  # annualized mean rate-of-return
sigma = 0.30  # volatility
S0 = 100  # initial stock price
N = 252  # number of time-steps (days)
dt = 1 / 252  # time-step in years

# Calculating the up and down factors and probability (using Wilmott's method because we want mu to be mean ra
# u = np.exp(sigma * np.sqrt(dt))
u = 1 + sigma * np.sqrt(dt)
# d = np.exp(-1 * sigma * np.sqrt(dt))
d = 1 - sigma * np.sqrt(dt)
p = 0.5 + mu * np.sqrt(dt) / (2 * sigma)

# Function to simulate a binomial path
def simulate_binomial_path():
    path = [S0]
    for _ in range(N):
        if np.random.rand() < p:
            path.append(path[-1] * u)
        else:
            path.append(path[-1] * d)
    return path

# Simulating three random paths
path1 = simulate_binomial_path()
path2 = simulate_binomial_path()
path3 = simulate_binomial_path()

# Plotting the paths
plt.figure(figsize=(10, 6))
plt.plot(path1, label='Path 1')
plt.plot(path2, label='Path 2')
plt.plot(path3, label='Path 3')
plt.title('Simulated Binomial Tree Paths for Stock Price')
plt.xlabel('Days')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```




### Exercise 5

#### (a)

Write a Python function that calculates the price of a call option using the binomial tree model. The function should take as input parameters:

- Spot price $S_0$
- Strike price $K$
- Time to maturity $T$ (in years)
- Number of time-steps $N$
- Volatility $\sigma$ (annualized)
- Risk free rate r (annualized)

**Answer**

```
import numpy as np
import matplotlib.pyplot as plt

def binomial_tree_call_option(S0, K, T, N, sigma, r):
    """
    Calculate the European call option price using a binomial tree model.

    Parameters:
    S0 (float): initial stock price
    K (float): strike price of the option
    T (float): time to expiration in years
    N (int): number of time steps
    sigma (float): annualized volatility of the stock
    r (float): annualized risk-free interest rate
```

```python
    Returns:
    float: price of the call option
    """
    # Time step
    dt = T / N
    # Up and down factors
    # u = np.exp(sigma * np.sqrt(dt))
    u = 1 + sigma * np.sqrt(dt)
    # d = np.exp(-1 * sigma * np.sqrt(dt))
    d = 1 - sigma * np.sqrt(dt)
    # Risk-neutral probability
    q = (np.exp(r * dt) - d) / (u - d)

    # Initialize asset prices at maturity (this holds an array of all possible stock prices at maturity)
    S = np.zeros(N + 1)
    S[0] = S0 * d**N
    for i in range(1, N + 1):
        S[i] = S[i - 1] * u / d

    # Initialize option values at maturity (this holds an array of all possible option values at maturity)
    V = np.maximum(S - K, 0)

    # Recursive calculation of option value at each node (working backwards from maturity):
    # The exp term discounts the expected value of the option at the next time step back to the current time s
    # The q and (1-q) terms are the risk-neutral probabilities of the stock price going up and down, respectiv
    # q * V[1:] is the expected value of the option if the stock price goes up
    # (1 - q) * V[:-1] is the expected value of the option if the stock price goes down
    # V is getting shorter as we work backwards in time
    for i in range(N - 1, -1, -1):
        V[:-1] = np.exp(-r * dt) * (q * V[1:] + (1 - q) * V[:-1])

    return V[0] # option value at t=0
```

```python
binomial_tree_call_option(S0=100, K=100, T=1, N=252, sigma=0.30, r=0.05)
```

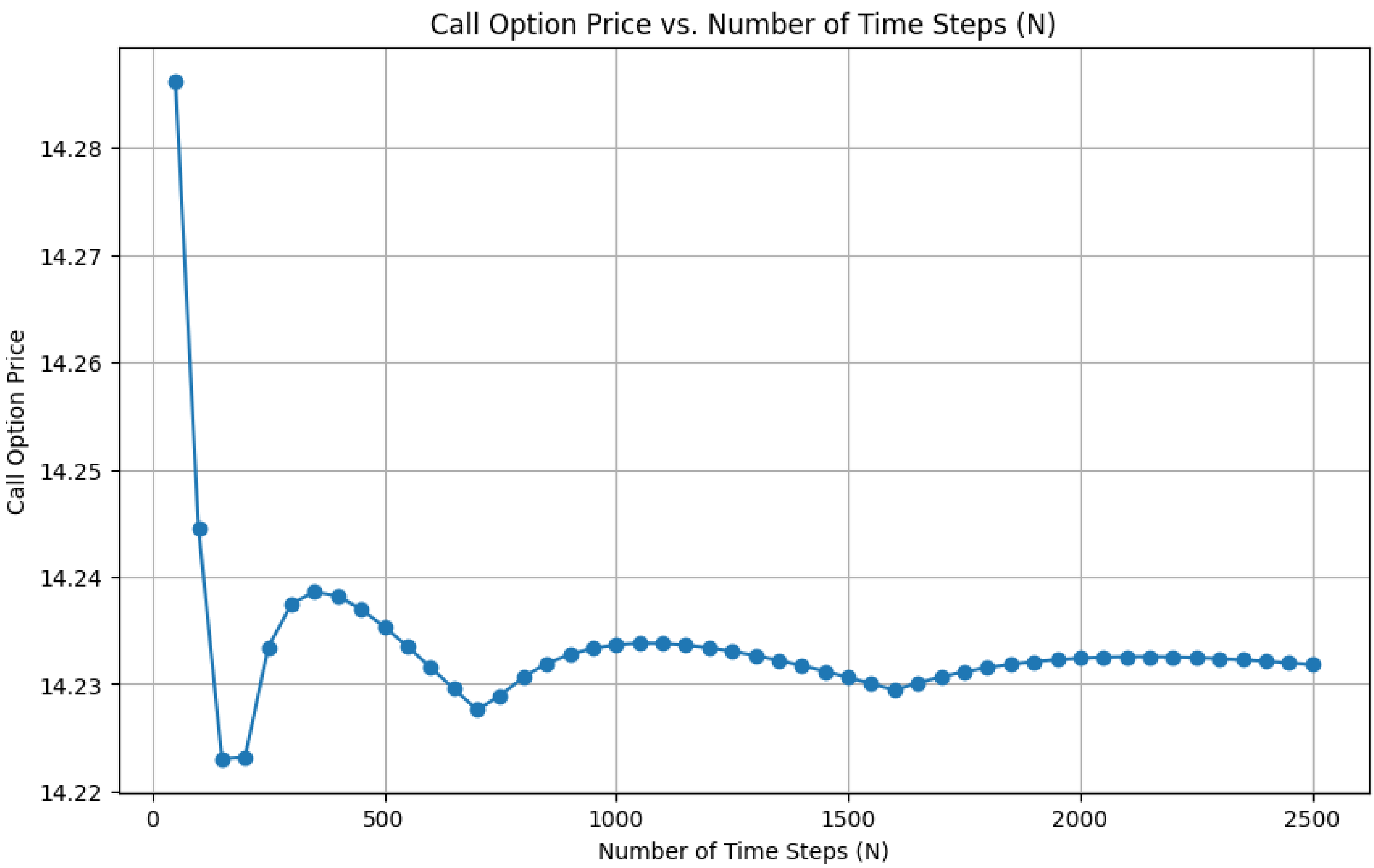
```
14.233658780211217
```

**(b)**

Use your function to calculate the price of an at-the-money call option with $S_0 = 100$, $K = 100$, $T = 1$ , $\sigma = 30\%$, $r = 5\%$. Do this for all values of $N$ in the range $50, 100, 150, \ldots, 2500$ (with increments of 50). Plot the resulting price as a function of the number of time steps $N$.

**Answer**

```python
# Range of N values
N_values = range(50, 2501, 50)  # From 50 to 2500 with increments of 50

# Calculate option prices for each N
option_prices = [binomial_tree_call_option(S0=100, K=100, T=1, N=N_i, sigma=0.30, r=0.05) for N_i in N_values]

# Plotting
plt.figure(figsize=(10, 6))
plt.plot(N_values, option_prices, marker='o')
plt.title('Call Option Price vs. Number of Time Steps (N)')
plt.xlabel('Number of Time Steps (N)')
plt.ylabel('Call Option Price')
plt.grid(True)
plt.show()
```




**(c)**

Find a Black-Scholes calculator online. What is the price of the call option of item (b) according to the Black-Scholes calculator? Does your code confirm that the price calculated by the binomial tree model converges to the Black- Scholes price as $N \to \infty$?

**Answer** The price of the call is \$14.23124, or around \$14.23, and the graph in (b) does converge to that value as $N \to \infty$.

**(d)**

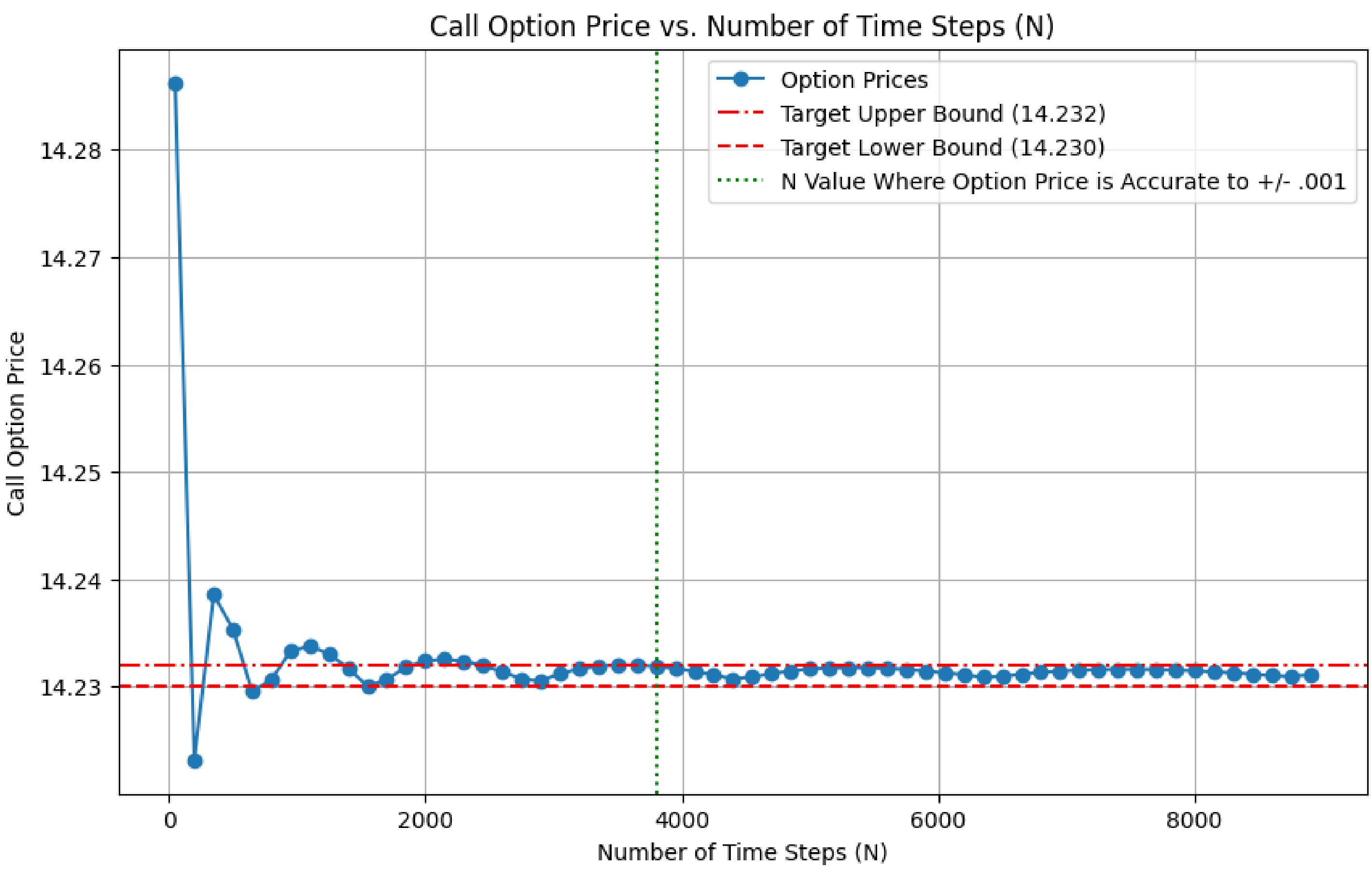
For which value of $N$ is the price calculated by your code within \$0.001 of the Black-Scholes price?

**Answer** We interpreted this question as which value of $N$ is the price calculated by our code within \$0.001 of the Black-Scholes price for all $n \geq N$ (not just the first value of N that is within \$0.001 of the Black-Scholes price).

```
import matplotlib.pyplot as plt

# Range of N values
N_values = range(50, 9001, 150)  # From 50 to 2500 with increments of 50

# Calculate option prices for each N
option_prices = [binomial_tree_call_option(S0=100, K=100, T=1, N=N_i, sigma=0.30, r=0.05) for N_i in N_values]

# Find the N value where the option price is between 14.230 and 14.232
accurate_N_index = None

for i in range(len(N_values)):
    if option_prices[i] > 14.23 and option_prices[i] < 14.232 and accurate_N_index is None:
        accurate_N_index = i
    elif option_prices[i] < 14.23 or option_prices[i] > 14.232:
        accurate_N_index = None
```

```python
print(f'Found +/- .001 price at N = {N_values[accurate_N_index]}')

# Plotting
plt.figure(figsize=(10, 6))
plt.plot(N_values, option_prices, marker='o', label='Option Prices')
plt.axhline(y=14.232, color='red', linestyle='-.', label='Target Upper Bound (14.232)')
plt.axhline(y=14.230, color='red', linestyle='--', label='Target Lower Bound (14.230)')
plt.axvline(x=N_values[accurate_N_index], color='green', linestyle=':', label='N Value Where Option Price is A

plt.title('Call Option Price vs. Number of Time Steps (N)')
plt.xlabel('Number of Time Steps (N)')
plt.ylabel('Call Option Price')
plt.legend()
plt.grid(True)
plt.show()
```

```
Found +/- .001 price at N = 3800
```




### (e)

Plot the price of an at-the-money call option with $S_0 = 100,\ T = 1,\ \sigma = 30\%,\ r = 5\%$ as a function of strike $K$. The x-axis shows $K$. The y-axis shows the price of the option. (Use your code to find all the option prices for $K = 20, 25, 30, \ldots, 200$.)

Combine in one graph a plot for the option price if volatility is 30%, and a plot if volatility is 60%.

```python
import matplotlib.pyplot as plt

# Range of strike prices K
K_values = range(20, 201, 5)

# Volatility levels
```

```python
volatility_levels = [0.30, 0.60]

# Plotting
plt.figure(figsize=(10, 6))

for sigma in volatility_levels:
    option_prices = [binomial_tree_call_option(S0=100, K=K_i, T=1, N=3000, sigma=sigma, r=0.05) for K_i in K_v
    plt.plot(K_values, option_prices, marker='o', label=f'Volatility {sigma * 100}%')

plt.title('Call Option Price vs. Strike Price')
plt.xlabel('Strike Price (K)')
plt.ylabel('Call Option Price')
plt.legend()
plt.grid(True)
plt.show()
```


**(f)**

Based on your graph, what is the effect of increased volatility on the option price?

**Answer** Looking at the graph in (e), **a higher volatility would lead to a higher option price**, since the option will be more valuable for speculating on or hedging against the volatility of an underlying stock.