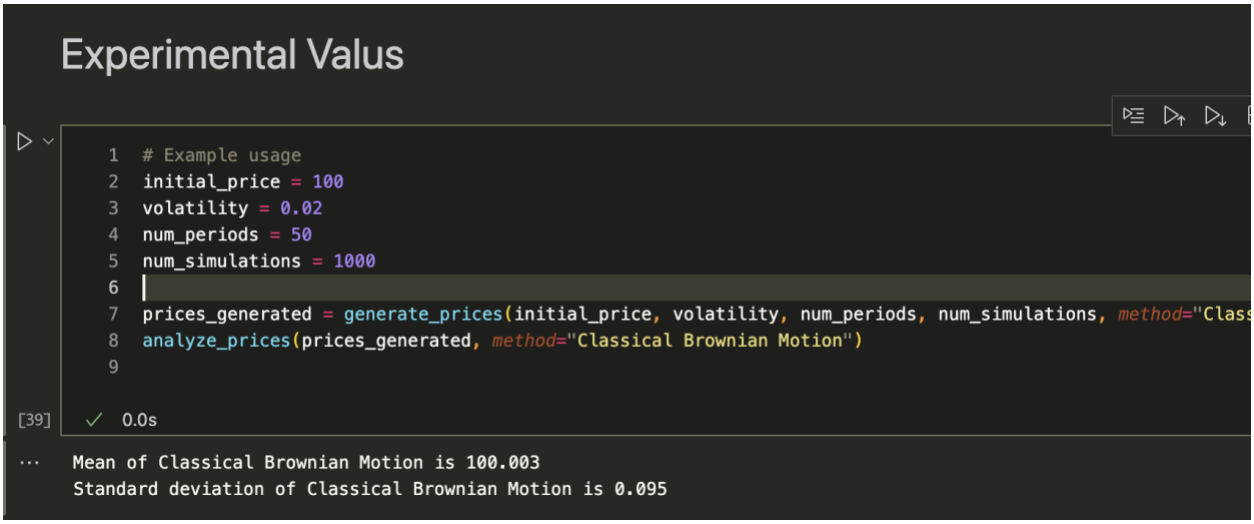
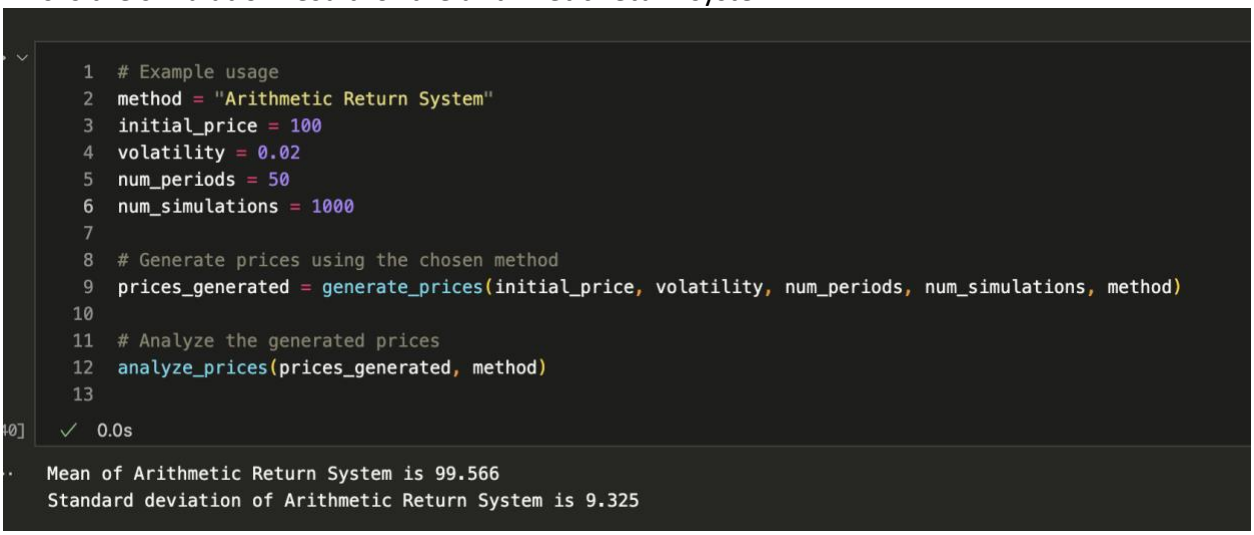
For Problem 1:
This is the result for the classical Brownian motion:



This is the simulation result for the arithmetic return system:



And here is the result for the geometric Brownian motion:

```python
# Example usage
method = "Log Return"
initial_price = 100
volatility = 0.02
num_periods = 50
num_simulations = 1000

# Generate prices using the chosen method
prices_generated = generate_prices(initial_price, volatility, num_periods, num_

# Analyze the generated prices
analyze_prices(prices_generated, method)

```

✓ 0.0s

```
Mean of Log Return is 100.476
Standard deviation of Log Return is 9.327
```

And then I calculated the theoretical value for the P_t for the three methods respectively:

Theoritical Value

```python
mean_value = math.exp(math.log(100) + 0.02 / 2)
sd_value = math.sqrt((math.exp(0.02) - 1) * (math.exp(2 * math.log(100) + 0.02)))
print(mean_value, sd_value)
```

[44] ✓ 0.0s

```
101.00501670841683 14.355986265538286
```

Theoritical Value

```python
mean_value_3 = np.mean(np.log(np.array(price3)))
print(mean_value_3)
mean_return = np.mean(return_value)
print(mean_return + np.mean(np.array(np.log(price3))))
```

[30] ✓ 0.0s

```
6.0472968628918995
6.053934532618685
```

```python
sd_3 = np.std(np.log(np.array(price3)))
print(sd_3)
sd_return = np.std(return_value)
print(sd_return)
```

[31] ✓ 0.0s

```
1.0499582395543046
0.09412544716060432
```

And find that the theoretical Value and the simulation value are quite close.

For Problem2:
Calculate the arithmetic returns for all prices, and the results are shown below:

	Date	SPY	AAPL	MSFT	AMZN	NVDA	GOOGL	TSLA	GOOG	BRK-B	...
0	2022-09-02	-0.010544	-0.013611	-0.016667	-0.002425	-0.020808	-0.017223	-0.025076	-0.016915	-0.016854	...
1	2022-09-06	-0.003773	-0.008215	-0.010974	-0.010980	-0.013336	-0.009643	0.015581	-0.011042	-0.003890	...
2	2022-09-07	0.017965	0.009254	0.019111	0.026723	0.018795	0.024717	0.033817	0.027912	0.016089	...
3	2022-09-08	0.006536	-0.009618	0.001666	0.002626	0.020126	-0.009776	0.019598	-0.009595	0.008184	...
4	2022-09-09	0.015535	0.018840	0.022977	0.026575	0.028377	0.020945	0.036023	0.021568	0.008576	...
...	...	...	...	...	...	...	...	...	...	...	...
260	2023-09-18	0.000586	0.016913	-0.003513	-0.002920	0.001503	0.005895	-0.033201	0.004772	0.006986	...
261	2023-09-19	-0.002074	0.006181	-0.001246	-0.016788	-0.010144	-0.001230	0.004599	-0.000936	0.000135	...
262	2023-09-20	-0.009193	-0.019992	-0.023977	-0.017002	-0.029435	-0.031150	-0.014672	-0.030541	-0.009879	...
263	2023-09-21	-0.016528	-0.008889	-0.003866	-0.044053	-0.028931	-0.024675	-0.026239	-0.023999	-0.009651	...
264	2023-09-22	-0.002249	0.004945	-0.007887	-0.001624	0.014457	-0.001457	-0.042315	-0.000837	-0.008588	...

265 rows × 102 columns

And then remove the mean from the series so that the mean(META)=0.

1. Calculate VaR using a normal distribution.

```
✓ 0.0s

VaR for Normal Distribution model is 0.07907071888888416
```

2. Using a normal distribution with an Exponentially Weighted variance (λ = 0. 94)

```
✓ 0.0s

VaR for normal distribution with exponentially weighted variance is: 0.029930783600041692
```

3. Using a MLE fitted T distribution.

```
✓ 0.9s

VaR for maximum likelihood estimated T distribution is: 0.04375001669592293
```

4. Using a fitted AR(1) model.

```
24

✓ 0.2s

VaR for AR 1 model is: 0.0014132935564990965
```

5. Using a Historic Simulation

```
    print("VAR for Historic Simulation is: {}".format(VaR))
[31] ✓ 0.0s
...  VAR for Historic Simulation is: 0.03939050784430346
```

Comparing the above result, we could find that the VaR value calculated with fitted AR(1) model is the smallest. And VaR values calculated using other methods are quite close. And all VaR values are close to 0.

For Problem 3:
Using exponentially weighted covariance matrix for VaR with lambda = 0.94, we could get the following results:

```
✓ 0.4s

The current value for A is: 1089316.16
VaR for A with Weighted Covirance is: 15426.97
VaR for A with Historic Simulation is: 17933.41

The current value for B is: 574542.41
VaR for B with Weighted Covirance is: 8082.57
VaR for B with Historic Simulation is: 10983.46

The current value for C is: 1387409.51
VaR for C with Weighted Covirance is: 18163.29
VaR for C with Historic Simulation is: 23083.95

The current value for All is: 3051268.07
VaR for All with Weighted Covirance is: 38941.38
VaR for All with Historic Simulation is: 48481.10
```

We could find that the asset B has the lowest VaR, while the asset C has the highest VaR. And we could also find that the VaR calculated with exponentially weighted covariance matrix is generally smaller than the VaR values calculated with the Historic Simulation method. And the reason for choosing the Historic Simulation method is that it's relatively simple and intuitive to understand. It directly uses historical data to estimate potential future losses, and it's based on actual historical data, it captures extreme events and tail risks effectively. In addition, it makes no assumption about the distribution of returns, which can be an advantage when dealing with non-normal or unknown distributions.
And the difference between the two method is that:
Historical Simulation implicitly assumes that past observations are equally likely to occur in the future, regardless of their age. In contrast, Exponentially Weighted Covariance gives more weight to recent observations and less to older ones. And exponentially Weighted Covariance tends to handle changes in volatility and stationarity better by assigning more weight to recent observations, adapting to changing market conditions. Historical Simulation treats all historical data equally, which might not reflect changing market conditions accurately.