Dollar Cost Averaging Strategy Explained

Introduction

Dollar Cost Averaging (DCA) is an investment strategy where an investor divides the total amount to be invested into periodic purchases of a target asset, aiming to reduce the impact of volatility. This approach can be particularly effective in volatile markets, as it spreads out the cost basis over time.

In this notebook, we will use **backtrader**, a Python library for backtesting trading strategies, to assess how the DCA strategy would have performed historically on the SPY ETF, an exchange-traded fund tracking the S&P 500 index.

Objectives:

1. Fetch historical data for SPY ETF using `yfinance`.
2. Implement the DCA strategy in **backtrader**.
3. Visualize and interpret the results.

Libraries Used:

- `yfinance`: Fetch financial data from Yahoo Finance.
- `backtrader`: Backtesting library for financial strategies.
- `matplotlib`: Plotting and visualization.
- `numpy` and `pandas`: Data manipulation.

In [1]:
```
import datetime
import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import yfinance as yf
import backtrader as bt
```

Data Retrieval

We fetch historical price data for the SPY ETF using `yfinance`. The data spans approximately 10 years (2520 trading days). This data will be used as input for the backtesting process.

Steps:

1. Define the target stock (SPY) and the date range for analysis.
2. Use `yfinance.download` to fetch adjusted closing prices.
3. Prepare the data for compatibility with `backtrader`.

Below is the code to fetch and preprocess the data.

In [2]:
```python
# Function to fetch stock data using yfinance
def get_data(stocks, start, end):
    """
    Fetch historical price data for specified stocks.

    Args:
        stocks (list): List of stock tickers.
        start (datetime): Start date for data fetching.
        end (datetime): End date for data fetching.

    Returns:
        pd.DataFrame: DataFrame containing stock prices.
    """
    stockData = yf.download(stocks, start, end)
    return stockData

# Define stock ticker and date range
stockList = ['SPY']  # SPY ETF tracks S&P 500
endDate = datetime.datetime.now()
startDate = endDate - datetime.timedelta(days=2520)  # Approx. 10 years o

# Fetch data
stockData = get_data(stockList, startDate, endDate)

# Preprocess data for backtrader compatibility
stockData.columns = stockData.columns.droplevel(1)  # Flatten MultiIndex
stockData.columns.name = None

# Verify the actual data start date
actualStart = stockData.index[0]

data = bt.feeds.PandasData(dataname=stockData)
```

```
[**********************100%**********************]  1 of 1 completed
```

Strategy Implementation

We implement two strategies:

1. **Buy and Hold:** Invest the entire available cash at the start and hold until the end of the investment period.
2. **Buy and Hold with Additional Investments:** Start with a fixed amount and add a specified amount of cash monthly, simulating a DCA approach.

Each strategy calculates key metrics such as return on investment (ROI), gross returns, and annualized performance.

In [3]:
```python
class BuyAndHold(bt.Strategy):
    def start(self):
        # Initial cash at the start of the strategy
        self.val_start = self.broker.get_cash()

    def nextstart(self):
        # Invest all available cash (minus a safety buffer) in the stock
```

```
        size = math.floor((self.broker.get_cash() - 10) / self.data[0])
        self.buy(size=size)

    def stop(self):
        # Calculate and display the performance metrics
        self.roi = (self.broker.get_value() / self.val_start) - 1
        print('-' * 50)
        print('BUY & HOLD')
        print(f'Starting Value:  ${self.val_start:,.2f}')
        print(f'ROI:              {self.roi * 100.0:.2f}%')
        print(f'Annualised:       {100 * ((1 + self.roi) ** (365 / (endDa
        print(f'Gross Return:    ${self.broker.get_value() - self.val_sta
```

Buy and Hold with Monthly Investments (DCA)

This strategy simulates a Dollar Cost Averaging (DCA) approach. It:

1. Starts with a small amount of initial capital.
2. Adds a fixed monthly amount at random intervals within a specified range.
3. Invests all available cash each month.

The strategy evaluates metrics such as:

- Total cost of investments.
- Gross and percentage returns.
- ROI and fund value.

In [4]:
```
class BuyAndHold_More_Fund(bt.Strategy):
    params = dict(
        monthly_cash=1000,  # Amount to add monthly
        monthly_range=[5, 20]  # Random days of the month for investment
    )

    def __init__(self):
        # Initialize strategy variables
        self.order = None
        self.totalcost = 0  # Track total invested amount including commi
        self.cost_wo_bro = 0  # Total cost excluding commissions
        self.units = 0  # Number of units bought
        self.times = 0  # Number of times investments were made

    def log(self, txt, dt=None):
        # Logging function for tracking activity
        dt = dt or self.datas[0].datetime.date(0)
        print(f'{dt.isoformat()}, {txt}')

    def start(self):
        # Initialize broker settings
        self.broker.set_fundmode(fundmode=True, fundstartval=100.0)
        self.cash_start = self.broker.get_cash()
        self.val_start = 100.0

        # Add a timer for monthly investments
        self.add_timer(
            when=bt.timer.SESSION_START,
            monthdays=[i for i in self.p.monthly_range],
            monthcarry=True
```

```
        )

    def notify_timer(self, timer, when, *args):
        # Add monthly cash and invest
        self.broker.add_cash(self.p.monthly_cash)
        target_value = self.broker.get_value() + self.p.monthly_cash - 10
        self.order_target_value(target=target_value)

    def notify_order(self, order):
        # Track order completion and log execution details
        if order.status in [order.Completed]:
            if order.isbuy():
                self.log(
                    f'BUY EXECUTED, Price {order.executed.price:.2f}, Cos
                    f'Comm {order.executed.comm:.2f}, Size {order.execute
                )
                self.units += order.executed.size
                self.totalcost += order.executed.value + order.executed.c
                self.cost_wo_bro += order.executed.value
                self.times += 1

        elif order.status in [order.Canceled, order.Margin, order.Rejecte
            self.log('Order Canceled/Margin/Rejected')

        self.order = None

    def stop(self):
        # Calculate and display the performance metrics
        self.roi = (self.broker.get_value() / self.cash_start) - 1
        self.froi = (self.broker.get_fundvalue() - self.val_start)
        value = self.datas[0].close * self.units + self.broker.get_cash()
        print('-' * 50)
        print('BUY & BUY MORE')
        print(f'Time in Market: {(endDate - actualStart).days / 365:.1f}
        print(f'#Times:         {self.times:.0f}')
        print(f'Value:         ${value:,.2f}')
        print(f'Cost:          ${self.totalcost:,.2f}')
        print(f'Gross Return:  ${value - self.totalcost:,.2f}')
        print(f'Gross %:        {(value / self.totalcost - 1) * 100:.2f}%
        print(f'ROI:            {self.roi * 100.0:.2f}%')
        print(f'Fund Value:     {self.froi:.2f}%')
        print(f'Annualised:     {100 * ((1 + self.froi / 100) ** (365 / (
        print('-' * 50)
```

Custom Commission Scheme

A fixed commission scheme is implemented to simulate realistic trading costs. This ensures that the backtesting results account for brokerage fees.

Key Parameters:

- `commission`: The fixed amount charged per transaction.
- `stocklike`: Indicates that the asset behaves like a stock.
- `commtype`: Specifies the type of commission (fixed in this case).

In [5]:
```
class FixedCommisionScheme(bt.CommInfoBase):
    params = (
```

```
        ('commission', 10),  # Fixed commission per trade
        ('stocklike', True),  # Treat the asset like a stock
        ('commtype', bt.CommInfoBase.COMM_FIXED)  # Fixed commission type
    )

    def _getcommission(self, size, price, pseudoexec):
        # Calculate commission based on fixed amount
        return self.p.commission
```

Running the Strategies

The `run` function executes the two strategies (`BuyAndHold` and `BuyAndHold_More_Fund`) with the following steps:

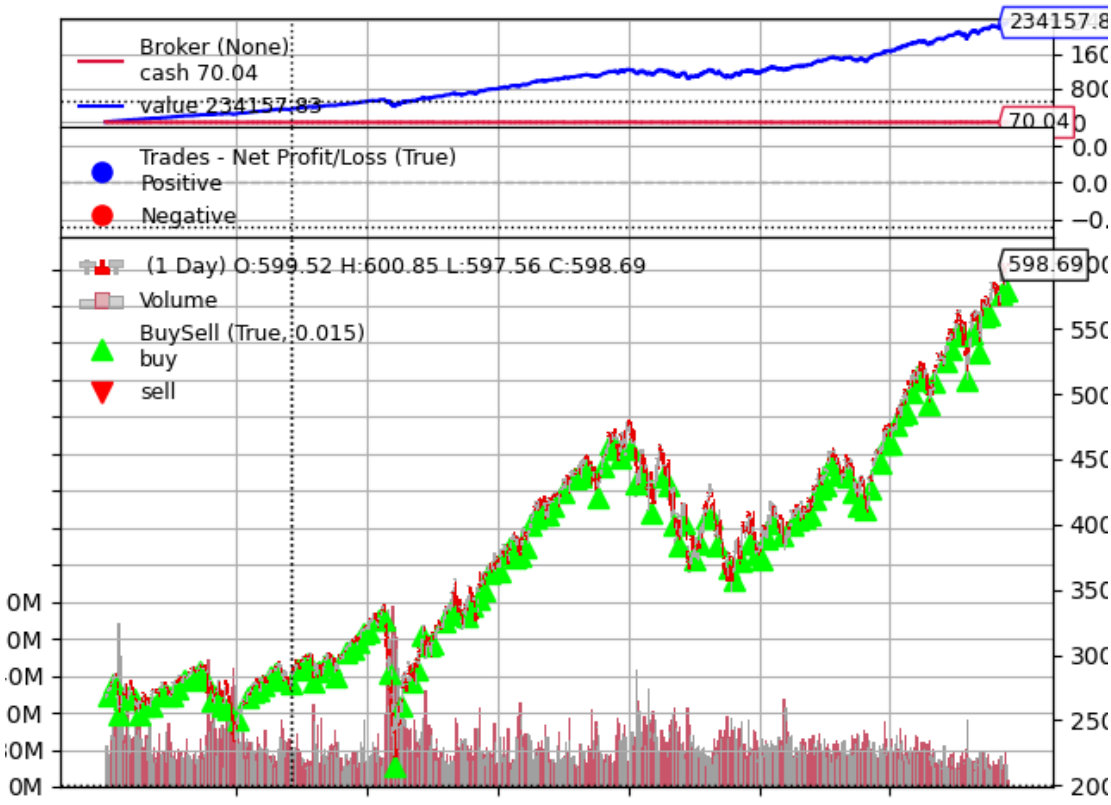
1. **Initialize `Cerebro` engine:** The main framework for running backtests in `backtrader`.
2. **Add data feed:** Input the stock data fetched earlier.
3. **Add strategies:** Include the two strategies for comparison.
4. **Set broker settings:** Define initial cash, commission schemes, and broker arguments.
5. **Execute and visualize results:** Run the strategies and generate performance plots.

In [6]:
```
def run(data):
    # BUY and HOLD
    cerebro = bt.Cerebro()
    cerebro.adddata(data)  # Add data feed to Cerebro
    cerebro.addstrategy(BuyAndHold)  # Add Buy and Hold strategy

    # Configure broker for the Buy and Hold strategy
    broker_args = dict(coc=True)  # Enable cash-on-cash calculation
    cerebro.broker = bt.brokers.BackBroker(**broker_args)
    comminfo = FixedCommisionScheme()  # Use the fixed commission scheme
    cerebro.broker.addcommissioninfo(comminfo)
    cerebro.broker.set_cash(100000)  # Set initial cash

    # BUY and BUY MORE
    cerebro1 = bt.Cerebro()
    cerebro1.adddata(data)  # Add data feed to Cerebro
    cerebro1.addstrategy(BuyAndHold_More_Fund)  # Add DCA strategy

    # Configure broker for the DCA strategy
    cerebro1.broker = bt.brokers.BackBroker(**broker_args)
    cerebro1.broker.addcommissioninfo(comminfo)
    cerebro1.broker.set_cash(1000)  # Start with a smaller cash amount

    # Run strategies
    cerebro1.run()
    cerebro.run()

    # Plot results
    cerebro.plot(iplot=False, style='candlestick')  # Visualize Buy and H
    cerebro1.plot(iplot=False, style='candlestick')  # Visualize DCA stra

# Run the backtesting function with the prepared data
run(data)
```

2018-01-08, BUY EXECUTED, Price 273.42, Cost 1913.94, Comm 10.00, Size 7
2018-01-23, BUY EXECUTED, Price 282.69, Cost 848.07, Comm 10.00, Size 3
2018-02-06, BUY EXECUTED, Price 263.93, Cost 1055.72, Comm 10.00, Size 4
2018-02-21, BUY EXECUTED, Price 271.40, Cost 1085.60, Comm 10.00, Size 4
2018-03-06, BUY EXECUTED, Price 272.19, Cost 816.57, Comm 10.00, Size 3
2018-03-21, BUY EXECUTED, Price 270.95, Cost 1083.80, Comm 10.00, Size 4
2018-04-06, BUY EXECUTED, Price 265.64, Cost 1062.56, Comm 10.00, Size 4
2018-04-23, BUY EXECUTED, Price 266.61, Cost 799.83, Comm 10.00, Size 3
2018-05-08, BUY EXECUTED, Price 266.92, Cost 1067.68, Comm 10.00, Size 4
2018-06-06, BUY EXECUTED, Price 275.10, Cost 1100.40, Comm 10.00, Size 4
2018-06-21, BUY EXECUTED, Price 275.97, Cost 827.91, Comm 10.00, Size 3
2018-07-06, BUY EXECUTED, Price 273.11, Cost 1092.44, Comm 10.00, Size 4
2018-07-23, BUY EXECUTED, Price 279.68, Cost 839.04, Comm 10.00, Size 3
2018-08-07, BUY EXECUTED, Price 284.64, Cost 1138.56, Comm 10.00, Size 4
2018-09-06, BUY EXECUTED, Price 289.03, Cost 867.09, Comm 10.00, Size 3
2018-09-21, BUY EXECUTED, Price 293.58, Cost 1174.32, Comm 10.00, Size 4
2018-10-08, BUY EXECUTED, Price 287.82, Cost 863.46, Comm 10.00, Size 3
2018-10-23, BUY EXECUTED, Price 275.01, Cost 1100.04, Comm 10.00, Size 4
2018-11-06, BUY EXECUTED, Price 273.39, Cost 820.17, Comm 10.00, Size 3
2018-11-21, BUY EXECUTED, Price 264.12, Cost 1056.48, Comm 10.00, Size 4
2018-12-07, BUY EXECUTED, Price 269.84, Cost 1079.36, Comm 10.00, Size 4
2019-01-08, BUY EXECUTED, Price 254.38, Cost 1017.52, Comm 10.00, Size 4
2019-02-06, BUY EXECUTED, Price 273.10, Cost 819.30, Comm 10.00, Size 3
2019-02-21, BUY EXECUTED, Price 278.41, Cost 1113.64, Comm 10.00, Size 4
2019-03-06, BUY EXECUTED, Price 279.02, Cost 837.06, Comm 10.00, Size 3
2019-03-21, BUY EXECUTED, Price 281.55, Cost 1126.20, Comm 10.00, Size 4
2019-04-08, BUY EXECUTED, Price 288.57, Cost 865.71, Comm 10.00, Size 3
2019-04-23, BUY EXECUTED, Price 290.27, Cost 1161.08, Comm 10.00, Size 4
2019-05-07, BUY EXECUTED, Price 292.82, Cost 878.46, Comm 10.00, Size 3
2019-06-06, BUY EXECUTED, Price 282.96, Cost 1131.84, Comm 10.00, Size 4
2019-06-21, BUY EXECUTED, Price 295.86, Cost 887.58, Comm 10.00, Size 3
2019-07-08, BUY EXECUTED, Price 298.46, Cost 895.38, Comm 10.00, Size 3
2019-07-23, BUY EXECUTED, Price 297.90, Cost 1191.60, Comm 10.00, Size 4
2019-08-06, BUY EXECUTED, Price 283.82, Cost 851.46, Comm 10.00, Size 3
2019-08-21, BUY EXECUTED, Price 290.09, Cost 1160.36, Comm 10.00, Size 4
2019-09-06, BUY EXECUTED, Price 297.82, Cost 893.46, Comm 10.00, Size 3
2019-09-23, BUY EXECUTED, Price 298.28, Cost 894.84, Comm 10.00, Size 3
2019-10-08, BUY EXECUTED, Price 293.08, Cost 1172.32, Comm 10.00, Size 4
2019-11-06, BUY EXECUTED, Price 307.03, Cost 921.09, Comm 10.00, Size 3
2019-11-21, BUY EXECUTED, Price 310.77, Cost 932.31, Comm 10.00, Size 3
2019-12-06, BUY EXECUTED, Price 312.02, Cost 936.06, Comm 10.00, Size 3
2019-12-23, BUY EXECUTED, Price 320.73, Cost 962.19, Comm 10.00, Size 3
2020-01-07, BUY EXECUTED, Price 323.64, Cost 970.92, Comm 10.00, Size 3
2020-02-06, BUY EXECUTED, Price 332.86, Cost 998.58, Comm 10.00, Size 3
2020-02-21, BUY EXECUTED, Price 336.95, Cost 1010.85, Comm 10.00, Size 3
2020-03-06, BUY EXECUTED, Price 302.46, Cost 1209.84, Comm 10.00, Size 4
2020-03-23, BUY EXECUTED, Price 228.80, Cost 915.20, Comm 10.00, Size 4
2020-04-07, BUY EXECUTED, Price 264.86, Cost 1059.44, Comm 10.00, Size 4
2020-05-06, BUY EXECUTED, Price 286.19, Cost 858.57, Comm 10.00, Size 3
2020-05-21, BUY EXECUTED, Price 296.93, Cost 890.79, Comm 10.00, Size 3
2020-06-08, BUY EXECUTED, Price 319.34, Cost 958.02, Comm 10.00, Size 3
2020-06-23, BUY EXECUTED, Price 310.62, Cost 1242.48, Comm 10.00, Size 4
2020-07-07, BUY EXECUTED, Price 317.05, Cost 951.15, Comm 10.00, Size 3
2020-08-06, BUY EXECUTED, Price 332.11, Cost 996.33, Comm 10.00, Size 3
2020-08-21, BUY EXECUTED, Price 338.28, Cost 1014.84, Comm 10.00, Size 3
2020-09-09, BUY EXECUTED, Price 333.21, Cost 999.63, Comm 10.00, Size 3
2020-10-06, BUY EXECUTED, Price 339.76, Cost 679.52, Comm 10.00, Size 2
2020-10-21, BUY EXECUTED, Price 343.38, Cost 1030.14, Comm 10.00, Size 3
2020-11-06, BUY EXECUTED, Price 350.24, Cost 1050.72, Comm 10.00, Size 3
2020-11-23, BUY EXECUTED, Price 355.33, Cost 1065.99, Comm 10.00, Size 3

2020-12-08, BUY EXECUTED, Price 369.09, Cost 1107.27, Comm 10.00, Size 3
2021-01-06, BUY EXECUTED, Price 371.33, Cost 742.66, Comm 10.00, Size 2
2021-01-21, BUY EXECUTED, Price 383.89, Cost 1151.67, Comm 10.00, Size 3
2021-02-08, BUY EXECUTED, Price 387.71, Cost 775.42, Comm 10.00, Size 2
2021-02-23, BUY EXECUTED, Price 387.03, Cost 1161.09, Comm 10.00, Size 3
2021-03-08, BUY EXECUTED, Price 383.63, Cost 1150.89, Comm 10.00, Size 3
2021-03-23, BUY EXECUTED, Price 392.59, Cost 785.18, Comm 10.00, Size 2
2021-04-06, BUY EXECUTED, Price 406.36, Cost 812.72, Comm 10.00, Size 2
2021-04-21, BUY EXECUTED, Price 412.17, Cost 1236.51, Comm 10.00, Size 3
2021-05-06, BUY EXECUTED, Price 415.75, Cost 831.50, Comm 10.00, Size 2
2021-05-21, BUY EXECUTED, Price 415.28, Cost 1245.84, Comm 10.00, Size 3
2021-06-08, BUY EXECUTED, Price 422.19, Cost 844.38, Comm 10.00, Size 2
2021-07-07, BUY EXECUTED, Price 432.93, Cost 865.86, Comm 10.00, Size 2
2021-08-06, BUY EXECUTED, Price 441.76, Cost 883.52, Comm 10.00, Size 2
2021-08-23, BUY EXECUTED, Price 443.36, Cost 1330.08, Comm 10.00, Size 3
2021-09-08, BUY EXECUTED, Price 451.46, Cost 902.92, Comm 10.00, Size 2
2021-10-06, BUY EXECUTED, Price 433.10, Cost 866.20, Comm 10.00, Size 2
2021-10-21, BUY EXECUTED, Price 452.41, Cost 904.82, Comm 10.00, Size 2
2021-11-08, BUY EXECUTED, Price 468.53, Cost 937.06, Comm 10.00, Size 2
2021-11-23, BUY EXECUTED, Price 467.57, Cost 1402.71, Comm 10.00, Size 3
2021-12-07, BUY EXECUTED, Price 458.79, Cost 917.58, Comm 10.00, Size 2
2022-01-06, BUY EXECUTED, Price 468.38, Cost 936.76, Comm 10.00, Size 2
2022-01-21, BUY EXECUTED, Price 446.75, Cost 893.50, Comm 10.00, Size 2
2022-02-08, BUY EXECUTED, Price 447.26, Cost 894.52, Comm 10.00, Size 2
2022-03-08, BUY EXECUTED, Price 419.43, Cost 1258.29, Comm 10.00, Size 3
2022-04-06, BUY EXECUTED, Price 451.03, Cost 902.06, Comm 10.00, Size 2
2022-04-21, BUY EXECUTED, Price 444.71, Cost 889.42, Comm 10.00, Size 2
2022-05-06, BUY EXECUTED, Price 413.81, Cost 1241.43, Comm 10.00, Size 3
2022-05-23, BUY EXECUTED, Price 389.63, Cost 779.26, Comm 10.00, Size 2
2022-06-07, BUY EXECUTED, Price 411.79, Cost 823.58, Comm 10.00, Size 2
2022-07-06, BUY EXECUTED, Price 381.96, Cost 1145.88, Comm 10.00, Size 3
2022-07-21, BUY EXECUTED, Price 394.77, Cost 1184.31, Comm 10.00, Size 3
2022-08-08, BUY EXECUTED, Price 413.47, Cost 826.94, Comm 10.00, Size 2
2022-08-23, BUY EXECUTED, Price 413.35, Cost 826.70, Comm 10.00, Size 2
2022-09-07, BUY EXECUTED, Price 390.76, Cost 1172.28, Comm 10.00, Size 3
2022-10-06, BUY EXECUTED, Price 377.09, Cost 1131.27, Comm 10.00, Size 3
2022-10-21, BUY EXECUTED, Price 365.41, Cost 730.82, Comm 10.00, Size 2
2022-11-08, BUY EXECUTED, Price 379.95, Cost 1139.85, Comm 10.00, Size 3
2022-12-06, BUY EXECUTED, Price 399.59, Cost 799.18, Comm 10.00, Size 2
2022-12-21, BUY EXECUTED, Price 380.54, Cost 1141.62, Comm 10.00, Size 3
2023-01-06, BUY EXECUTED, Price 379.38, Cost 1138.14, Comm 10.00, Size 3
2023-01-23, BUY EXECUTED, Price 395.88, Cost 791.76, Comm 10.00, Size 2
2023-02-07, BUY EXECUTED, Price 409.83, Cost 819.66, Comm 10.00, Size 2
2023-03-07, BUY EXECUTED, Price 404.47, Cost 1213.41, Comm 10.00, Size 3
2023-04-06, BUY EXECUTED, Price 407.60, Cost 815.20, Comm 10.00, Size 2
2023-04-21, BUY EXECUTED, Price 411.88, Cost 1235.64, Comm 10.00, Size 3
2023-05-08, BUY EXECUTED, Price 412.63, Cost 825.26, Comm 10.00, Size 2
2023-05-23, BUY EXECUTED, Price 418.79, Cost 1256.37, Comm 10.00, Size 3
2023-06-06, BUY EXECUTED, Price 427.10, Cost 854.20, Comm 10.00, Size 2
2023-06-21, BUY EXECUTED, Price 437.18, Cost 874.36, Comm 10.00, Size 2
2023-07-06, BUY EXECUTED, Price 443.13, Cost 886.26, Comm 10.00, Size 2
2023-07-21, BUY EXECUTED, Price 452.18, Cost 904.36, Comm 10.00, Size 2
2023-08-08, BUY EXECUTED, Price 450.71, Cost 1352.13, Comm 10.00, Size 3
2023-09-06, BUY EXECUTED, Price 449.24, Cost 898.48, Comm 10.00, Size 2
2023-09-21, BUY EXECUTED, Price 438.64, Cost 877.28, Comm 10.00, Size 2
2023-10-06, BUY EXECUTED, Price 424.50, Cost 1273.50, Comm 10.00, Size 3
2023-10-23, BUY EXECUTED, Price 421.19, Cost 842.38, Comm 10.00, Size 2
2023-11-07, BUY EXECUTED, Price 435.69, Cost 871.38, Comm 10.00, Size 2
2023-12-06, BUY EXECUTED, Price 456.60, Cost 913.20, Comm 10.00, Size 2
2023-12-21, BUY EXECUTED, Price 468.26, Cost 936.52, Comm 10.00, Size 2

```
2024-01-08, BUY EXECUTED, Price 467.92, Cost 935.84, Comm 10.00, Size 2
2024-01-23, BUY EXECUTED, Price 483.45, Cost 966.90, Comm 10.00, Size 2
2024-02-06, BUY EXECUTED, Price 492.55, Cost 985.10, Comm 10.00, Size 2
2024-02-21, BUY EXECUTED, Price 496.76, Cost 993.52, Comm 10.00, Size 2
2024-03-06, BUY EXECUTED, Price 507.18, Cost 1014.36, Comm 10.00, Size 2
2024-03-21, BUY EXECUTED, Price 520.48, Cost 1040.96, Comm 10.00, Size 2
2024-04-08, BUY EXECUTED, Price 518.43, Cost 1036.86, Comm 10.00, Size 2
2024-04-23, BUY EXECUTED, Price 499.72, Cost 999.44, Comm 10.00, Size 2
2024-05-07, BUY EXECUTED, Price 516.57, Cost 1033.14, Comm 10.00, Size 2
2024-06-06, BUY EXECUTED, Price 534.67, Cost 1069.34, Comm 10.00, Size 2
2024-06-21, BUY EXECUTED, Price 547.00, Cost 1094.00, Comm 10.00, Size 2
2024-07-08, BUY EXECUTED, Price 554.64, Cost 1109.28, Comm 10.00, Size 2
2024-07-23, BUY EXECUTED, Price 554.65, Cost 554.65, Comm 10.00, Size 1
2024-08-06, BUY EXECUTED, Price 517.38, Cost 1034.76, Comm 10.00, Size 2
2024-08-21, BUY EXECUTED, Price 558.70, Cost 1117.40, Comm 10.00, Size 2
2024-09-06, BUY EXECUTED, Price 549.61, Cost 1099.22, Comm 10.00, Size 2
2024-09-23, BUY EXECUTED, Price 568.25, Cost 1136.50, Comm 10.00, Size 2
2024-10-08, BUY EXECUTED, Price 567.80, Cost 567.80, Comm 10.00, Size 1
2024-11-06, BUY EXECUTED, Price 576.70, Cost 1153.40, Comm 10.00, Size 2
2024-11-21, BUY EXECUTED, Price 590.50, Cost 1181.00, Comm 10.00, Size 2
--------------------------------------------------
BUY & BUY MORE
Time in Market: 6.9 years
#Times:         140
Value:         $231,322.10
Cost:          $140,922.02
Gross Return:  $90,400.08
Gross %:        64.15%
ROI:            23032.21%
Fund Value:     109.58%
Annualised:     11.31%
--------------------------------------------------
--------------------------------------------------
BUY & HOLD
Starting Value:  $100,000.00
ROI:              122.29%
Annualised:       12.27%
Gross Return:    $122,288.73
```



Final Thoughts

The Dollar Cost Averaging (DCA) strategy offers a disciplined approach to investing, particularly in volatile markets. By comparing the Buy and Hold strategy with the DCA strategy, we can evaluate the benefits of systematic investing.

Next Steps:

1. Experiment with different monthly cash amounts and investment intervals.
2. Test the strategies on other stocks or ETFs.
3. Integrate risk management techniques to enhance the strategies.