```
"""import yfinance as yf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt


ticker = "AAPL"
start_date = "2010-01-01"
end_date = "2020-01-01"

data = yf.download(ticker, start=start_date, end=end_date)

print(data.head())


short_window = 50
long_window = 200

data['SMA50'] = data['Close'].rolling(window=50).mean()
data['SMA200'] = data['Close'].rolling(window=200).mean()

data['Signal'] = 0

data.loc[data['SMA50']>data['SMA200'],'Signal']=1
data.loc[data['SMA50']>data['SMA200'],'Signal']=-1


data['Position'] = data['Signal'].shift(1)

data['Daily Return'] = data['Close'].pct_change()

data['Strategy Return'] = data['Position'] * data['Daily Return']

data['Cumulative Market Return'] = (1 + data['Daily Return']).cumprod()
data['Cumulative Strategy Return'] = (1 + data['Strategy Return']).cumprod()"""

"""plt.figure(figsize=(14, 7))
plt.plot(data['Close'], label='Close Price', alpha=0.5)
plt.plot(data['SMA50'], label='SMA50', alpha=0.75)
plt.plot(data['SMA200'], label='SMA200', alpha=0.75)
plt.title(f"{ticker} Price and Moving Averages")
plt.legend()
plt.show()"""

"""plt.figure(figsize=(14, 7))
plt.plot(data['Cumulative Market Return'], label='Market Return', alpha=0.75)
plt.plot(data['Cumulative Strategy Return'], label='Strategy Return', alpha=0.75)
plt.title("Cumulative Returns")
plt.legend()
plt.show()"""
```

```
"""total_strategy_return = data['Cumulative Strategy Return'].iloc[-1] - 1
total_market_return = data['Cumulative Market Return'].iloc[-1] - 1

print(f"Total Strategy Return: {total_strategy_return:.2%}")
print(f"Total Market Return: {total_market_return:.2%}")"""

"""import yfinance as yf
import ta
import pandas as pd


data = yf.download("AAPL", period="5d", interval="15m")
data.dropna(inplace=True)

close_series = pd.Series(data['Close'].values.flatten(), index=data.index)

data['rsi'] = ta.momentum.RSIIndicator(close=close_series).rsi()
data['ema20'] = ta.trend.EMAIndicator(close=close_series,
window=20).ema_indicator()

def generate_signal(row):
    rsi = row['rsi'].item() if hasattr(row['rsi'], "item") else row['rsi']
    close = row['Close'].item() if hasattr(row['Close'], "item") else row['Close']
    ema20 = row['ema20'].item() if hasattr(row['ema20'], "item") else row['ema20']

    if rsi < 30 and close > ema20:
        return "BUY"
    elif rsi > 70 and close < ema20:
        return "SELL"
    else:
        return "HOLD"

data['signal'] = data.apply(generate_signal, axis=1)

print(data[['Close', 'rsi', 'ema20', 'signal']].tail())"""

"""import random
import pandas as pd

prices = [100]
for _ in range(99):

    prices.append(prices[-1] + random.uniform(-2, 2))

data = pd.DataFrame(prices, columns=['Close'])

data['rsi'] = [random.uniform(0, 100) for _ in range(len(data))]

data['ema20'] = data['Close'].rolling(window=20).mean().bfill()
```

```
def generate_signal(row):
    if row['rsi'] < 30 and row['Close'] > row['ema20']:
        return "BUY"
    elif row['rsi'] > 70 and row['Close'] < row['ema20']:
        return "SELL"
    else:
        return "HOLD"

data['signal'] = data.apply(generate_signal, axis=1)

print(data.tail(10))"""


"""portfolio = {"cash": 10000, "shares": 0}

def buy_stock(portfolio, price, qty):
    portfolio["cash"] -= price * qty
    portfolio["shares"] += qty

def sell_stock(portfolio, price, qty):
    portfolio["cash"] += price * qty
    portfolio["shares"] -= qty

buy_stock(portfolio, price=100, qty=10)
print(portfolio)

sell_stock(portfolio, price=120, qty=5)
print(portfolio)

portfolio2 = portfolio
portfolio2["cash"] = 5000
print(portfolio)

portfolio2 = portfolio.copy()"""

"""a = [1, 2, 3]
b = a
b.append(4)
print(a)

x = 10
y = x
y += 5
print(x)"""

"""import yfinance as yf
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
```

```
data = yf.download("AAPL", period="60d", interval="1d")
data = data[['Close']]

data = data.asfreq('D')
data = data.reset_index()

model = ARIMA(data['Close'], order=(5,1,0))
model_fit = model.fit()

forecast = model_fit.forecast(steps=5)
print("Forecasted Prices:\n", forecast)"""
```