Psuedo Code for Analysis - Staff Answer Key

ISYS2001 Introduction to Business Programming

Table of contents

Introduction

These notes are staff-in-confidence please DO NOT share with the students. They provide one way to achieve the tasks, even though I risk influencing your guidance to the students. They may be wrong, but are provided for your understand. It is expected to be read in conjunction with the assignment specification. The aim is to provide a structured approach to the analysis tasks in the simulation assignments, guiding students through the process of exploring different scenarios and analysing the results. The pseudo code outlines the steps to follow and the functions to implement for each task, helping students understand the impact of various factors on the simulated systems.

Stock Market Simulation (and Resource Price FLuctuations) Analysis - Staff Answer Key

Here is some pseudo code for each task in the analysis, followed by the adjusted guidance on how to conduct visual analysis using the `StockMarketSimulation` class. This approach provides structured yet open-ended guidance to encourage student experimentation and learning.

> This is very similar to resource price fluctuations, but with a different context. The pseudo code is structured to guide students through the analysis process, helping them understand the impact of different factors on stock prices and investment strategies. Apply the same strategies to resource price fluctuations.

1. Investigate the Impact of Volatility

- Objective: Explore how different levels of volatility affect stock price fluctuations and the potential for investment gains or losses.

```
# Pseudo code for analysing the impact of different volatility levels on stock price stabili

volatility_levels = [0.01, 0.02, 0.05]  # Different levels of volatility to test
for volatility in volatility_levels:
    sim = StockMarketSimulation(start_price=100, days=365, volatility=volatility, drift=0.00
    prices = sim.run_simulation()
    plot_prices(prices)  # Visualise stock price fluctuations for each volatility level
    print_statistics(prices)  # Function to print or calculate statistical metrics (mean, sto
```

2. Simulate a Major Market Event

- Objective: Analyse the immediate and long-term effects of a major market event on stock prices.

```
# Pseudo code for simulating and analysing a major market event

event_days = [50, 150, 250]  # Days to simulate a market event
event_impacts = [0.1, -0.1]  # Positive for beneficial, negative for detrimental impacts
for day in event_days:
    for impact in event_impacts:
        sim = StockMarketSimulation(start_price=100, days=365, volatility=0.02,
                                    drift=0.001, event_day=day, event_impact=impact)
        prices = sim.run_simulation()
        plot_prices(prices, event_day=day)  # Visualise with the event day marked
        analyse_event_impact(prices, day)  # Analyse and compare before and after event price
```

3. Develop and Test Trading Strategies

- Objective: Develop simple trading strategies and test their effectiveness over the simulation period.

```
# Pseudo code for developing and testing trading strategies based on stock price trends

strategies = ['moving_average_crossover', 'buy_and_hold', 'contrarian']
for strategy in strategies:
    sim = StockMarketSimulation(start_price=100, days=365, volatility=0.02, drift=0.001)
```

```
    prices = sim.run_simulation()
    strategy_results = apply_strategy(prices, strategy)  # Apply and analyse strategy
    plot_strategy_results(strategy_results)  # Visualise effectiveness of strategies
```

Auxiliary Functions to Implement

- **print_statistics(prices)**: function to print or calculate statistical metrics (mean, std, max drawdown)
- **analyse_event_impact(prices, day)**: Analyse and compare before and after event prices
- **apply_strategy(strategy)**: apply and analyse strategy
- **plot_strategy_results(strategy_results)**: Visualise effectiveness of strategies

Product Popularity Simulation Analysis - Staff Answer Key

Here's the pseudo code for each of the tasks in the analysis section of your product popularity simulation assignment. This will help students structure their experiments and analyses effectively.

1. Examine How Changes in Growth Rate and Marketing Impact Affect Demand

- Objective: Investigate how different settings for natural growth rate and ongoing marketing efforts influence the overall demand for a product.

```
# Pseudo code for analysing changes in growth rate and marketing impact

growth_rates = [0.01, 0.02, 0.03]  # Different growth rates to test
marketing_impacts = [0.05, 0.1, 0.15]  # Different marketing impacts to test

for growth_rate in growth_rates:
    for marketing_impact in marketing_impacts:
        sim = ProductPopularitySimulation(start_demand=500, days=180, growth_rate=growth_rate
                                          marketing_impact=marketing_impact)
        demand = sim.run_simulation()
        plot_demand(demand)  # Visualise the demand curve for each combination
        print_demand_statistics(demand)  # Function to print or calculate demand statistics
```

2. Simulate a Major Marketing Campaign and Analyse Its Effect on Demand Growth

- Objective: Analyse how a significant promotional campaign affects the demand for a product both immediately and over time.

```
# Pseudo code for simulating and analysing a major marketing campaign

promotion_days = [30, 60, 90]  # Days to start the promotion campaign
promotion_effects = [0.2, 0.5, 0.7]  # Different levels of campaign effectiveness

for day in promotion_days:
    for effect in promotion_effects:
        sim = ProductPopularitySimulation(start_demand=500, days=180, growth_rate=0.02,
                                          marketing_impact=0.1, promotion_day=day, promotion_
        demand = sim.run_simulation()
        plot_demand(demand, promotion_day=day)  # Visualise with the campaign day marked
        analyse_campaign_effect(demand, day)  # Analyse and compare before and after campaig
```

3. Explore Different Marketing Strategies and Their Cost-Effectiveness

- Objective: Explore various marketing strategies and evaluate their return on investment (ROI) based on the increase in demand relative to the costs.

```
# Pseudo code for exploring and comparing different marketing strategies

strategies = ['social_media_boost', 'discount_offers', 'influencer_partnerships']
costs = {'social_media_boost': 1000, 'discount_offers': 500, 'influencer_partnerships': 1500

for strategy in strategies:
    # Assume different effectiveness and costs for each strategy
    sim = ProductPopularitySimulation(start_demand=500, days=180, growth_rate=0.02,
                                      marketing_impact=0.1, promotion_day=30, promotion_effe
    demand = sim.run_simulation()
    plot_demand(demand)  # Visualise effectiveness of strategies
    total_cost = costs[strategy]
    evaluate_roi(demand, total_cost)  # Calculate and display the ROI of each strategy
```

Auxiliary Functions to Implement

- **print_demand_statistics(demand)**: Calculates and prints statistics like mean demand, peak demand, etc.

- **analyse_campaign_effect(demand, promotion_day)**: Compares demand levels before and after the promotional campaign to assess its impact.
- **strategy_effectiveness(strategy)**: Returns a hypothetical effectiveness value based on the strategy type.
- **evaluate_roi(demand, cost)**: Calculates return on investment by comparing the increase in demand against the strategy cost.