# Step 4: Market Functionality

# 1. Overview

The simulation game receives its market features in Step 4 allowing players to transact with apples and grain through buying and selling operations. Players can handle their financial resources in the market system by buying seeds to plant crops which they can turn into profits from the sale of matured crops. This independent system functions separately from all field management features including planting crops or performing harvests and it maintains their core game functions without disturbance. Players' balance remains the sole element connected to the market system while the rest of the game persists unchanged by this modern addition.

# 2. Key Features of the Market System

The market system lets players take economic decisions, improving gameplay with operational elements. The player's balance decreases after buying apples and grain at particular prices. Apples cost $2 and grains $1. After purchasing an item, the player's account receives a balanced update. The "Insufficient funds" error appears when the player's account balance is inadequate for item purchase.

The game lets players sell ripe crops. Apples sell for $3 and grain for $2. Sales are recorded by boosting the player's account balance by the payment amount. Successful crop sales trigger game interface notifications. The market mechanism lets users monitor their balance throughout games. Engaging gameplay allows players manage resources efficiently.

# 3. Structure of the Market Class

In the Market class, the system handles sales, purchases, and balance management. The market starts when its builder puts $10 in it. Players may buy apple or grain goods using the buy() function, deducting the amount from their cash. When funds are low, a "Insufficient funds" message prevents purchases. The sale() function lets players sell ripe crops for money. Full-grown crops may be sold in the game, but for $3 each apple and $2 every grain. After selling crops to the market, the player gets feedback and money.

Requesting getBalance() gives players account value information to assist them determine whether to purchase or sell. The Market class accurately handles financial transactions by updating the player's balance after each buy or sell and providing accurate feedback.

# 4. Step 4 Execution in Marker.java

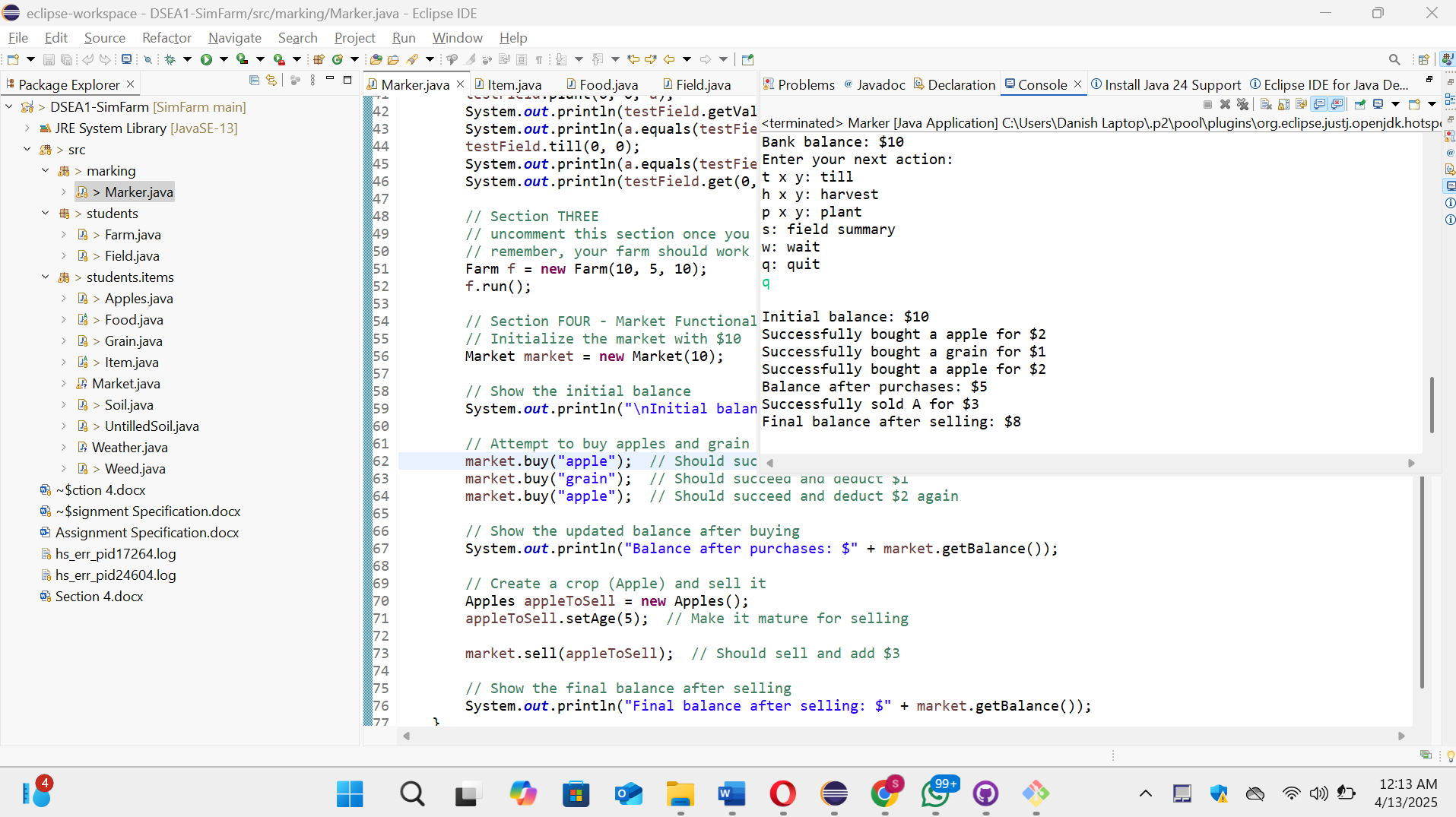
Developers added market testing to the conclusion of major farm activities in Marker.java Section FOUR. After crop planting, agricultural, and resource management tests, market system testing begins. First, the Market object accepts a $10 balance. Market purchases of apples and grain reduce account money. If the player attempts to buy with insufficient cash, a message will show. Sale of matured apples at $3 following crop acquisition increases game balance. After all purchases and sales that change a player's finances, the programme shows the adjusted final balance.   
The game uses the market mechanism to manage player finances while maintaining other gameplay aspects. Each transaction shows balance change, helping players comprehend market system implications on finances.

# 5. Testing

Marker.java does key farm activities to evaluate market functioning. Initial programme funds of $10 show on screen during execution. A $2 apple, $1 grain, and another apple buy drain the player's account to $5. Adding a matured apple sale will increase the amount by $3 since the user sets the apple age to 5. The system sends feedback messages after each buy or sell to check transaction status or notify the player of inadequate cash. When players try to spend more than their cash, the programme will warn them.

After each transaction, each product and commodity must be added to the account total to show market system operating. The programme should accurately adjust the player's balance during buying and selling transactions using real-time financial updates.

# 6. Expected Output for Testing



# 7. Conclusion

Step 4 enhances agricultural trade using a market structure. The market is autonomous from agricultural activities like planting and harvesting. Its market structure gives players more economic control. The game lets users oversee their farm's development by buying and selling crops. The integrated system of this feature adds dynamic gameplay components to the game while retaining basic agricultural functions.

Run Marker.java to see the market operating process and system updates that change account balance. Each market transaction gives the system clear feedback on buy and selling success. A strategic new UI component improves gameplay without changing farming mechanics.