# Experiment： Marketing Pricing Simulation

# Name: Li Guo

## Experiment Objectives:

This is an application that simulate the market pricing with implementing the particle swarm optimization Algorithm. It meant to help sellers, set up their price before entering a new market. It is inspired by the invisible hand theory suggested by Adam Smith. The invisible hand is a term to describe the unintended social benefits of individual actions. In the book The Wealth of Nations, Adam Smith supposed that if each consumer can choose freely what to buy and each producer can choose freely what to sell and how to produce it, the market will settle on a product distribution and prices that are beneficial to all the individual members of a community, and hence to the overall community.

## Experiment Design:

To implement this project, there are couple Objects need to be defined with certain properties per the Particle Swarm Optimization Algorithm:

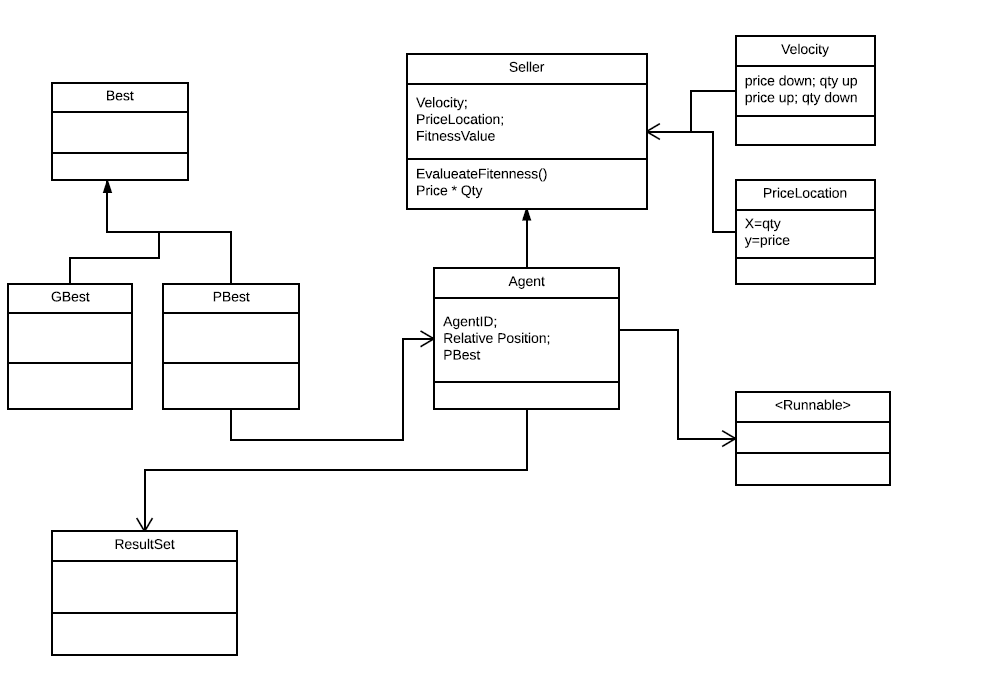
1. Particle: sellers/Agent
2. Velocity: the price difference to the other sellers’
   1. They can only make adjustment on their pricing strategy and the quantity of sell will be adjusted accordingly, based on the products’ elasticity
   2. Velocity for price: will be decided by sellers; based on the game theory
   3. Velocity for quantity: will change based on the demand-price elasticity formula
3. Location: current price of this seller
   1. Double X-qty: quantity, how much sell will achieve
   2. Double Y-price: how much the seller is pricing its product
4. Process: held the execute method
5. Resultset: holding all instantiated agents/threads, so that they could be viewed by other classes

And the function will be designed per common decision making process and game theory. When one seller sees his prices is too low comparing to the sellers around him, he would want to adjust his price accordingly.

There are some methods necessary to be defined as well:

1. calculateFitness();//the seller wants to achieve the highest profit as possible, so the fitness function is qty\*price = revenue
2. initializeSwarm();//initialize the swarm
3. execute();//the way how particles check himself in the market(global=overall market; local=niche market)

## Application pseudo code and UML:



Create a swarm of agents (initiate them with desired price range and set a wanted product elasticity)

Initiate the agents and let them execute();

In execute method:

For each agent; let them keep check the other agents’ price in a range of 20\*20 px and update local best;

Move toward the local best agent’s price accordingly (calculate price velocity);

Quantity if affected passively (calculate quantity based on the quantity-demand elasticity formula);

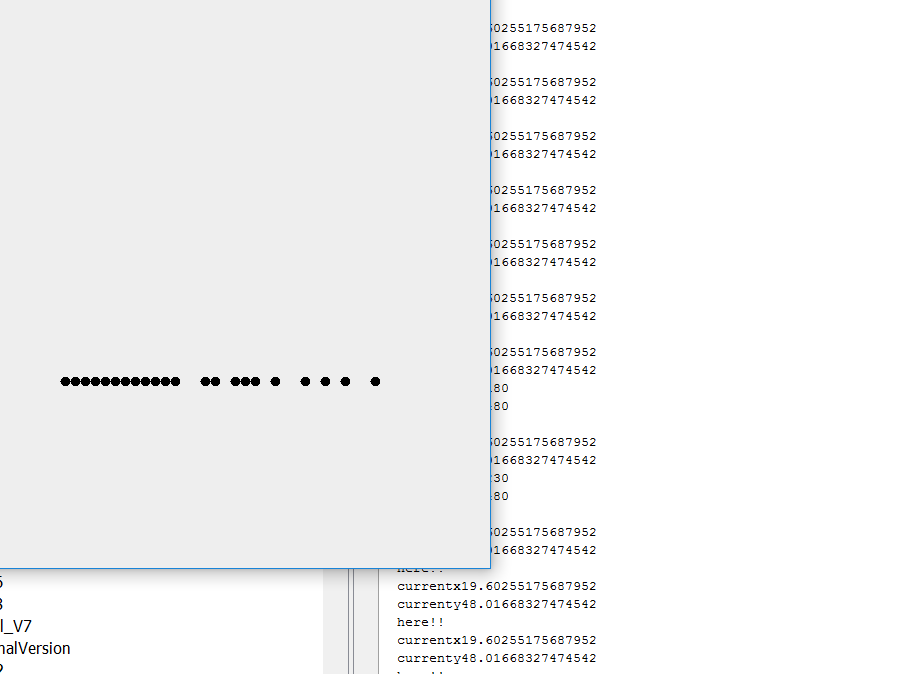
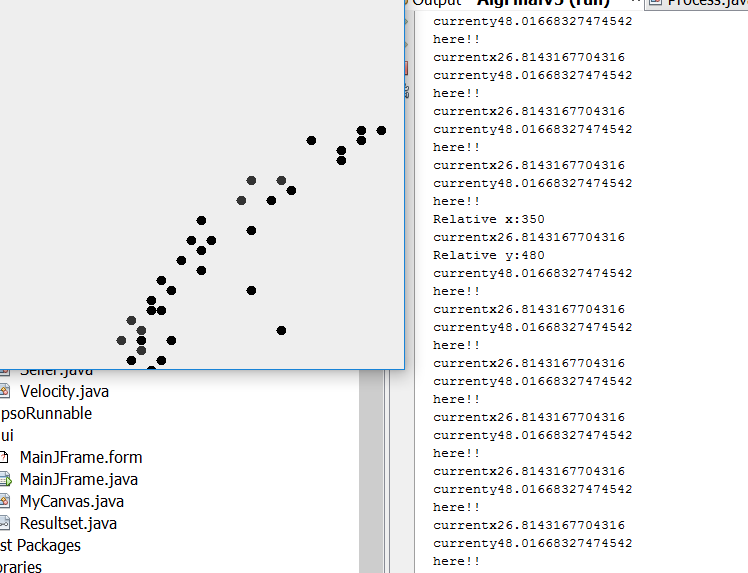
Meanwhile; update global best-optimum market price;

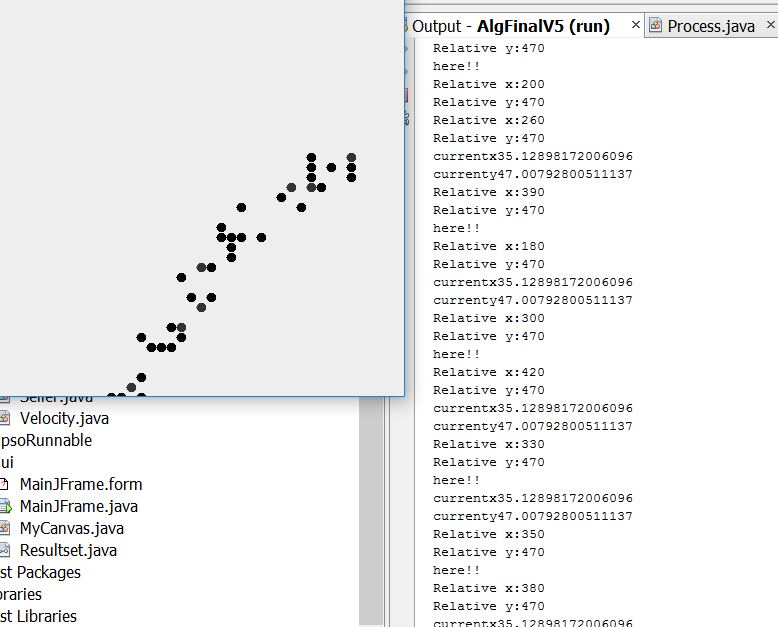
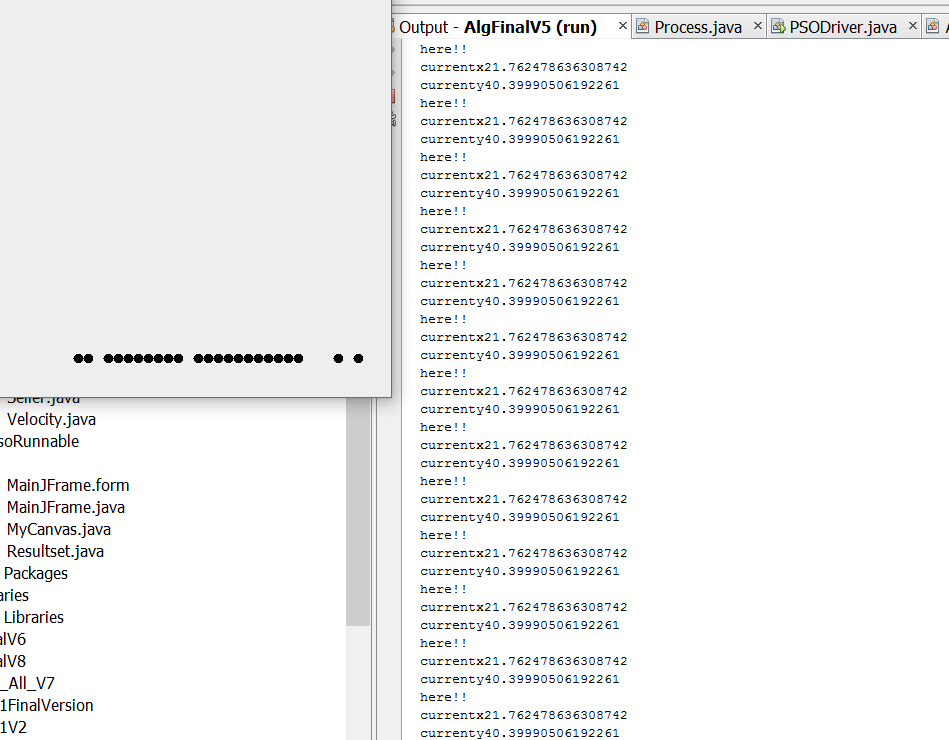
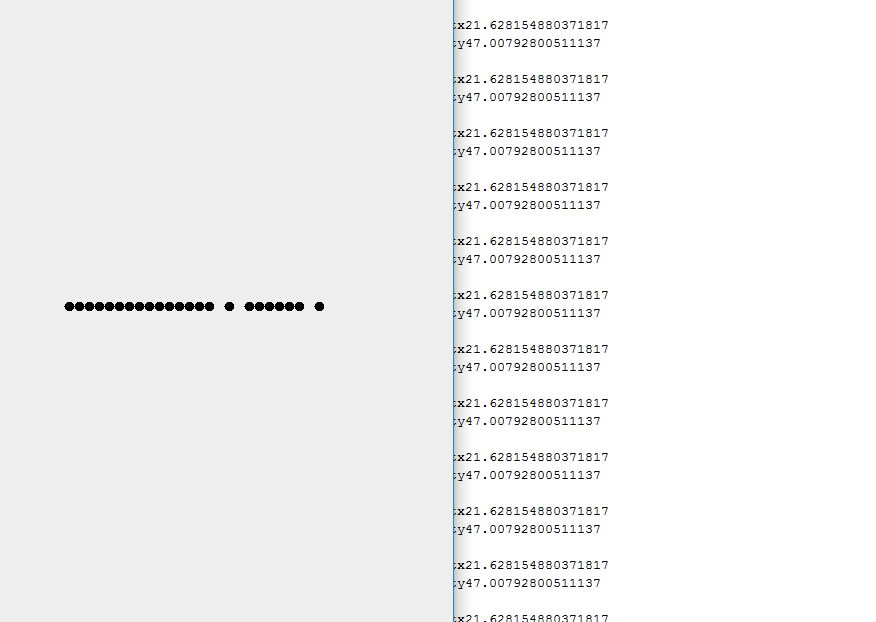
Paint result

## Experiment Result:

The result printed in console is shown below.

Started at randomized location:



The above are three test cases, and you can see all agents started at random location, but ends up to one price, which would be the optimal price for the chosen product in the market. In some other cases (which I didn’t get chance to take screen shot, there may be two price level, as the market may be break into high-end market or low-end market).

## Conclusion:

Per the report above, it simulated how the market adjust itself, and come up with the suitable price level for a product.

Both PPT presentation file and Java application are included in the zip file.