Middle East Technical University Department of Computer Engineering

CENG 443

Introduction to Object-Oriented
Programming Languages and Systems

2024-2025 Fall

Lab 1 Preliminary - Economics 101
(Ver 1.0)

1 Introduction

Keywords: *Object-Oriented Programming, Error Handling*

In this assignment, you are asked to write a simulation for a small trading area to familiarize you with object-oriented concepts and design principles for the upcoming Lab Quiz.

2 Overview

The application simulates money exchanges between customers and stores on a confined area. Stores keep stock of a single type of product like food and customers on the area visits several of the stores to buy their needs according to their shopping lists.

To give you an intuition you can find a video of a sample run on ODTUCLASS page of the assignment.

3 Simulation Entities

Full or partial implementations of some classes are provided to you. Your assignment is to design rest of the system according to object-oriented concepts and design principles and complete partially implemented classes accordingly. You should only add your code below **TODO** comments and not change any other part of the given code. You are expected to design rest of the simulation and write new classes accordingly.

Below you can find the explanation of given classes and expected behaviors of the parts you need to design.

3.1 SimulationRunner Class

Contains the main method.

3.2 Display Class

Represents the display on which entities are repeatedly drawn.

3.3 Common Class

Brings together simulation parameters, instances of entities, and other utility fields/methods that are required for running the application.

3.4 Entity Class

Represents a simulation entity like a store and a customer.

3.5 Position Class

Represents a position (x and y coordinates). Already filled for you.

3.6 Products

There are three types of products:

- Food
- Electronics
- Luxury

Shopping list of a customer can contain arbitrary number of products of these types and a store can have only single type of product in its store.

3.7 Customers

Customers keep a shopping list that the products they will buy. To buy a product, a customer should be inside a shop (their drawn entities should collide) that sells and have that type of product. A customer can only buy one product at a single step and should wait for the next time step to buy again. When they buy everything in their lists, customers leave the area and are removed from the simulation afterwards.

When shopping, a customer can use one of several strategies:

- **Cheapest:** Find and go to the cheapest store that **have** the **first product** on the list. Upon arrival, if the store is out of stock wait until the product is available again.
- **Closest:** Find and go to the closest store that **have** the **first product** on the list. Upon arrival, if the store is out of stock, try to go to the nearest closest store that sells the product from there.
- **Travelling:** Similar to **Closest** strategy, find the closest store, **buy** anything available and on the list and go to closest store from there that is not visited yet. After visiting all stores, if there are still things to buy on the shopping list, start to visit all stores again with same strategy.

Customers should be drawn as gray. Their strategies and first three products on their lists in abbreviation should be printed (Ch: Cheapest, Cl: Closest, Tr: Travelling; F: Food, E: Electronics, L: Luxury).

3.8 Store

Stores are added to simulation from the start, and they are permanent. A store can have a single type of product and its maximum stock and selling price is defined when created.

When a customer wants to buy an item and the store is out of stock it should signal it to the customer by **raising an error**.

The simulation replenishes all stocks in all stores, in intervals. You should also handle this case.

Stores should be drawn as orange and their product types, their stocks and their prices should be printed.

4 Some Remarks

Although your code will not be graded, and you have freedom in your design, It is crucial for you to understand, design and implement your solution according to object-oriented concepts and design principles, since the upcoming lab will be graded accordingly.

- Try to separate responsibility of entities into logical units and avoid all-in-one classes or God objects
- Ensure that you handled erroneous situations
- Use Java 8 or a newer version