SuperMarket

# Submission specifications:

* Name the program correctly: Market + *name1* + *name2* [ + *name3* ]
* The code of the program and any dependencies (the whole solution !)

YOU MUST DO AN OBJECT ORIENTED DESIGN AND CODE. YOU MUST ADD YOUR CLASS DIAGRAM MADE BY DRAW.IO AND SUBMIT WITH YOUR CODE + A README FILE.

# Description:

Create a prototype of a program that simulates a supermarket manager’s software. An interface that allows the user to track, receive, manage and order inventory, as well as pay employees and sell inventory.

* The “warehouse” can store at least up to 50 items.
  + “infinite” items for bonus mark.
* Buy inventory will show a catalog of 6 different items that can be purchased. The user can select one to buy.
  + User selects by index or name, both for a bonus mark.
  + It asks how much of the inventory to purchase. (The quantity)
  + The inventory should be added to the warehouse.
  + The cost should be recorded for “spending” tracking.
* Functionality to “Sell Inventory”
  + User selects which items to sell. (Any combination)
  + User then enters how much all that inventory is sold for.
  + The total amount of inventory items sold should be tracked
  + The total amount of earnings (sales) should be tracked.
* User can see a display that shows….
  + The current time (when display opens, doesn’t have to update).
  + The number of items bought.
  + The number of items sold.
  + Cost all of item purchased.
  + Cost of amount paid to employees.
  + The total expenses. (Item purchases and employee pay).
  + The total sales (The total amount sold of all items).
  + The total profit (net profit).
  + The user can enter an item type and see..
    - Number of that item sold
    - Number of that item bought
* The user can press a button to pay employees. Which costs 2600$ per minute since the last time employees were paid.
  + The total amount paid to employees should be tracked.

# Remarks:

* The system time used in the program is the actual time in real life.
* When the program ends, the memory should be properly managed.

## Presentation

This is a team assignment and must be worked together as a group, and **not copied from any external sources.** Each member is expected to write code, and will present the section in an individual presentation.

Each team member is expected to be familiar with all the code even in sections they did not write. Questions will be asked of team members who seem unfamiliar with the code, and a percent of the marks will be equally deducted.

# Evaluation:

The project will be graded according to the following criteria:

## Exactitude of the program:

Does the program do what it is supposed to do?

## Visual appearance:

All text displayed to the screen should be well arranged and written in proper English.

## Input validation:

The program should properly handle errors that can occur during data input.

## Structure of the code:

The program should be split up into functions according to the relevant needs.

## Arrangement and clarity of the code:

The code should be properly indented, **with relevant comments**, and it should respect programming conventions (variables in camelCase, constants in

UPPER\_CASE\_SEPARATED\_BY\_UNDERSCORES). Hard-coded values should be put into constants (magic numbers).

|  |  |
| --- | --- |
| **Tasks to accomplish** | **Point** |
| Respect submission specifications | 1 |
| Program compiles and executes | 5 |
| Structure of the code (comments,  etc) | 2 |
| Warehouse storage | 2 |
| Sell Inventory | 6 |
| Buy Inventory | 6 |
| Inventory removed properly | 3 |
| Inventory added properly | 3 |
| User Display functionality | 10 |
| User Display clarity (UI) | 3 |
| Pay Employees | 3 |
| Input validations | 1 |
| **Total** (with bonus in parentheses) | 45 |