**Sayuj Shrestha  
CS 2050 (M/W)**  
Car Vending Machine Design

# UML Design for class

|  |  |
| --- | --- |
| Classes | Needed |
| Car class | -Private int floor  -Private int space  -Private int year  -Private double year  -Private String make  -Private String model  Getters  +getFloor()  +getSpace()  +getYear()  +getPrice()  +getMake()  +getModel()  +toString() |
| Vending machine Class | -Private int floors  -Private int spaces  -Private Car[][] inventory  Methods  +addCar()  +displayVendingMachine()  +retrieveCar()  +printSortedInventoryByPrice()  +printSortedInventoryByYear()  -printCars()  -flatten2DArrayTo1DArray()  -insertionSort() |
| Driver Class | +main() -readCarFromFile() |

# Unit testing for methods

## addCar()

|  |  |  |
| --- | --- | --- |
| Preconditions | Expected Output | Notes |
| Empty VendingMachine  Add Car at (1,2) | Car is stored at (1,2) | If no car exists at (1,2), the car is added successfully because location is null |
| Slot (1,2) already has a car | Error: Slot at Floor: 1 Space: 2 is already occupied. | If a car exists at (1,2), the method should not overwrite it |
| Adding a car outside valid floor/space range  Add Car at (5,3)  (If the floor limit is 4) | Error: Invalid position at Floor: 5 Space: 3 |  |
| Adding a car with negative floor/space | Error: Invalid position at Floor: -1 Space: -1 | Negative should not be allowed |
| Adding multiple cars at different valid locations | Cars will be stored in their valid locations | Multiple cars can be added if slots are available |

## displayVendingMachine()

|  |  |  |
| --- | --- | --- |
| Preconditions | Expected Output | Notes |
| VendingMachine is empty | Displays all spaces as “EMPTY” |  |
| Some cars are in the machine | Displays cars in correct positions | Each car’s details should be printed in its assigned slot |
| All sorts are filled | Displays all cars correctly |  |
| Irregular distribution | Shows a mix of “EMPTY” and car details | Make sure the placements are correct and the readability is good |

## retrieveCar()

|  |  |  |
| --- | --- | --- |
| Preconditions | Expected Output | Notes |
| Car exists at (1,2) | Car retrieved from Floor 1 Location 2: (Car details) | Should display the car details |
| No car at (1,2) | No car located at Floor 1 Location 2 |  |

## printSortedInventoryByPrice() and printSortedInventoryByYear()

|  |  |  |
| --- | --- | --- |
| Preconditions | Expected Output | Notes |
| VendingMachine is empty | No cars are printed | No sorting happens |
| Multiple cars in inventory | Cars are displayed in ascending order | Sorting is based on selected attribute, price or year |
| One car in inventory | Displays the single car |  |

## readCarFromFile()

|  |  |  |
| --- | --- | --- |
| Preconditions | Expected Output | Notes |
| File exists with car data | Cars are added to the vending machine based on file data |  |
| File contains invalid data | Error message | Not handling this yet, it will cause an InputMismatchError |
| File does not exist  Load cars10000.txt | Error message  Error: File cars10000.txt not found. | Handle FileNotFoundException |
| File has data but they’re out of bound car positions  Loaded cars from file with 5 floors but dealership only has (4,4) spaces | Error: Invalid position at Floor: 5 Space: 5  Can not place Car (Car Details) |  |

## Pseudocode