Class UsedCarLot

java.lang.Object
    UsedCarLot

```
public class UsedCarLot
extends Object
```

This class represents a UsedCarLot object

Author:

Zhan Xiang Zheng

Constructor Summary

Constructors

Constructor	Description
`UsedCarLot()`	Initialize a UsedCarLost object

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method	Description
`void`	`addCar(int indexToAdd, Car carToAdd)`	Adds a Car to the inventory list at the index specified by indexToAdd; this method increases the size of inventory by 1
`void`	`addCar(Car newCar)`	Adds car to inventory
`ArrayList <Car>`	`getInventory()`	Gets the inventory ArrayList
`void`	`moveCar(int indexOfCarToMove, int destinationIndex)`	Moves Car located at index indexOfCarToMove to index destinationIndex; if destinationIndex < indexOfCarToMove, moves the Car to the right in inventory; if destinationIndex < indexOfCarToMove, moves the Car to the left in the inventory.
`Car`	`sellCarNoShift (int indexOfCarToSell)`	"sells" the Car located at indexOfCarToSell, but instead of

		removing it and shifting the inventory list to the left, REPLACE the Car at indexOfCarToSell with NULL, thus creating an "empty parking spot" on the lot; this method does NOT reduce the size of inventory by 1
`Car`	`sellCarShift(int indexOfCarToSell)`	"Sells" the Car located at indexOfCarToSell which removes it from the inventory list and shifting the remaining Cars in the inventory list to the left to fill in the gap; this method reduces the size of inventory by 1
`boolean`	`swap(int index1, int index2)`	Swaps the place of two cars in the inventory

Methods inherited from class java.lang.Object

`clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait , wait , wait`

Constructor Details

UsedCarLot

```
public UsedCarLot()
```

Initialize a UsedCarLost object

Method Details

getInventory

```
public ArrayList <Car> getInventory()
```

Gets the inventory ArrayList

Returns:

inventory

addCar

```
public void addCar(Car newCar)
```

Adds car to inventory

Parameters:

`newCar` - new car to be added

swap

```
public boolean swap(int index1,
                    int index2)
```

Swaps the place of two cars in the inventory

Parameters:

`index1` - the index of the first car to be switched

`index2` - the index of the second car to be switched

Returns:

whether the operation was successful or not

addCar

```
public void addCar(int indexToAdd,
                   Car carToAdd)
```

Adds a Car to the inventory list at the index specified by indexToAdd; this method increases the size of inventory by 1

PRECONDITION: 0 <= indexToAdd < inventory.size()

Parameters:

`indexToAdd` - the index of where to add the car

`carToAdd` - the car object to be added

sellCarShift

```
public Car sellCarShift(int indexOfCarToSell)
```

"Sells" the Car located at indexOfCarToSell which removes it from the inventory list and shifting the remaining Cars in the inventory list to the left to fill in the gap; this method reduces the size of inventory by 1

PRECONDITION: indexOfCarToSell < inventory.size()

Parameters:

`indexOfCarToSell` - the index of the car to be sold

Returns:

the car being sold

sellCarNoShift

```
public Car sellCarNoShift(int indexOfCarToSell)
```

"sells" the Car located at indexOfCarToSell, but instead of removing it and shifting the inventory list to the left, REPLACE the Car at indexOfCarToSell with NULL, thus creating an "empty parking spot" on the lot; this method does NOT reduce the size of inventory by 1

PRECONDITION: indexOfCarToSell < inventory.size()

Parameters:

`indexOfCarToSell` - index of the car to be sold

Returns:

the Car that is being "sold" (replaced with null)

moveCar

```
public void moveCar(int indexOfCarToMove,
                    int destinationIndex)
```

Moves Car located at index indexOfCarToMove to index destinationIndex; if destinationIndex < indexOfCarToMove, moves the Car to the right in inventory; if destinationIndex < indexOfCarToMove, moves the Car to the left in the inventory. All other cars in the inventory should shift accordingly

PRECONDITIONS: indexOfCarToMove < inventory.size() destinationIndex < inventory.size()

Parameters:

`indexOfCarToMove` - the car to be moved

`destinationIndex` - the index the car is moving to