Programming Languages CS 181
Winter of 2005

Programming Assignment #1
Simple Warehouse Manager in Prolog
Due: Sunday, March 5, 2005, 11:59 pm

Skeleton Program:

```
manager :-    getCommand(Command),
              executeCommand(Command, Status),
              displayReport(Command, Status),
              fail.

manager :-    write_ln('Good Bye!').
```

Note:

- The predicate "**getCommand**" sequentially retrieves individual commands to be processed by the warehouse manager. I will provide this predicate during grading. **DO NOT** include it in your submitted programs.
- The predicate "**executeCommand**" attempts to faithfully execute the given command and returns a status report.
- The predicate "**displayReport**" must output a clear and complete message to the screen providing the detailed status of the execution of the last command. The message must include:
  - Listing of the actual command
  - English description of the meaning of the command
  - Any output produced by the command itself.
  - Actions that have been performed
  - Any problems encountered, including any command syntax problems.
- Initially there are no items in stock in the warehouse.
- You can use dynamic predicates **assert** and **retract** only to to store information about the warehouse stock, customers and their past orders..

Types of Commands:

There are several types of commands. Required arguments are listed in **red** – all the others are optional. **ITEM** is an abbreviation for **item(model(Model), name(Name), brand(Brand)),** where only Model is required.

- ### Delivery Commands:

```
delivery(ITEM, count(Count), price(Price)).
```

A new shipment of Count of Items arrived, each priced at Price dollars. For example:

```
delivery(item(model(1234),name(dvd), brand(sony)), count(20), price(99))
delivery(item(model(1234),name(_), brand(_)), count(20), price(99))
```

- ### Order Commands:

```
order(customer(Customer), ITEM, count(Count), max_price(MaxPrice), type(Type)).
```

where Type = new | _ .

Customer wants to purchase Count of Items of a given Type, assuming they are available and the price is **<=** MaxPrice. If Type=new then customer wants only new items, otherwise open-box (returned) items are also acceptable. For example:

```
order(customer(ann),item(model(1234),name(_), brand(sony)), count(2),
max_price(300), type(_))
```

Satisfy the customer's order as much as possible. If there are no Count items available, sell as many as are available.

- **Return Commands:**

```
return(customer(Customer), ITEM, count(Count) )
```

A Customer wants to return Count of unwanted Items. Verify that the customer indeed placed an order for these items and did not yet return them. If so, return all the items (or as many as are eligible) back into stock but mark them as "open_box" items and reduce their price by 20%. When selling "open_box" items to customers who are willing to accept them, inform customers about it.

For example:

```
return(customer(ann), item(model(1234), name(_), brand(_)), count(2) )
```

- **Special Sale Commands:**

```
sale( model(Model), name(Name), brand(Brand), percentage(Percentage))
```

All the merchandise matching Model, Name and/or Brand is now on sale and their prices must be lowered by the given Percentage. This applies to both new and open-box items. Percentage is an integer between 1 and 90. For example:

```
sale( model(_), name(dvd), brand(sony), percentage(15))
```

- **Inventory Commands:**

```
inventory(model(Model), name(Name), brand(Brand),  type(Type) )
```

Prints a detailed inventory list of all items currently in stock that match the given Model, Name, Brand and Type, where **`where Type = new | open_box | _`** .. For example:

```
inventory(model(_), name(dvd), brand(_),  type(new) )
```

Good Luck!