**COMP15 Project 1**

**Part 1 Due February 20, 2013 11:59pm**
**Part 2 Due March 1, 2013 11:59pm**
**Part 3 Due March 6, 2013 11:59pm**

# Nozama Online

In this project you will simulate the operations of an order processing facility for a top online retailer. The warehouse facility has two main functions: **retrieving an order** and **packaging an order**. The warehouse has only one retrieval unit. It has three packaging units. The reason for the simulation is to determine the potential effects of their new supreme membership program on their regular customers.

## Orders

The orders for Nozama have an **arrival time**, a **fetch time**, and a **pack time**. For this simulation we will only have one item per order, future version will manage multiple items in an order. The fetch time is how long, in minutes, after the order arrival time, it takes to find the item in the order and load it into one of the packaging units. The pack time is how long the item will take to be packaged by one of the packaging units. Some orders will be for supreme members, these orders will get priority over other orders. For this simulation, orders will never arrive at the same time. An order will have a **total time** in the system, this time is the total number of minutes that have occurred from the time the order arrived until it has been packaged. For tracking purposes an order has an **order number** - a unique id. The arrival time is the number of minutes since the simulation started. You will get a input file detailing the orders placed over some time period. Example entries would look like:

```
Order 123 for regular customer arrived at time 0
fetch time 15 minutes, pack time 22 minutes

Order 127 for supreme customer arrived at time 12
fetch time 3 minutes, pack time 12 minutes
```

## Retrieval/Fetching

There is only one retrieval unit in this simulation. (Your design should consider how additional units may be added in the future.) The retrieval unit checks the list of orders that have arrived. If an order has arrived the system then fetches that item. The retrieval unit can only retrieve one item from the warehouse shelves at a time. Supreme orders are never handled differently by the retrieval unit. When an order has been retrieved, the retrieval unit sends the order to the packaging unit with the shortest queue. The size of the queue is measured in the total time needed to package all items currently in the queue.

## Packaging

There are three packaging units in our warehouse. (Your design should be able to handle any fixed number of units.) At any given time, a packaging unit can only be working on one item. Although not required, a packaging unit may keep track of the number of orders currently in the queue and the total time needed to package all of the

orders currently in the queue. When an order is packaged it leaves the warehouse. At that time the elapsed time is set. It is the difference between the current time and the time the order arrived.

## Packaging Protocols

We will be working with two different packaging protocols to help analyze efficiency.

**First in First out without interruption**

One simulation will test the scheduling protocol first in, first out. This means that every package will be processed in the order that they arrive. They will be processed until they are completely packaged. This simulation ignores supreme members.

**First in First out with interruption (supreme members)**

The second simulation will schedule first in, first out, except when a supreme member's order arrives. If the packaging unit is working on a customer which is not a supreme member, the unit will stop working on that order and process the supreme member first. Once the supreme member's order has been processed, it will return to the other package, unless a second supreme member's order has arrived, in which case it will process it before returning to the original order. Therefore, a supreme order prefers a queue with no current supreme orders. If all queues have them, it prefers the shortest wait.

To analyze these protocols you will create a file as follows

```
Order 123 for regular customer arrived at time 0
fetch time 15 minutes, pack time 22 minutes
elapsed time sim1 39 minutes, total time sim2 41 minutes

Order 127 for supreme customer arrived at time 12
fetch time 3 minutes, pack time 12 minutes
elapsed time sim1 18 minutes, total time sim2 16 minutes

...

sim1 min elapsed time 2 minutes
sim1 max elapsed time 10 minutes
sim1 mean elapsed time 7 minutes
126 orders processed

sim2 min total time 2 minutes
sim2 max total time 17 minutes
sim2 mean total time for all customers 17 minutes
sim2 mean total time for supreme customers 3 minutes
sim2 mean total time for regular customers 25 minutes
126 orders processed
```

Above each order is detail is printed, unlike the file read in, this file now has the correct elapsed time for an order. Notice the elapsed time may be different for the two different protocols. At the end of the file, you will have the min, max, mean data for each

simulation. The second simulation which considers regular and supreme members will have additional data.

## Test Files

We will provide a series of test files as it get closer to the March deadline. Students should hard code small test cases. You should also test individual functions while you are coding.

## Part One Deliverables

Part one is due February 20th, 2013. You will turn in all of your header files for your classes. **These headers files are allowed to change.** Namely, you may change your design after the first deadline. In the header files you should have detailed comments explaining your design. These header files and comments should describe what the class represents, the data in the class, and the functions (public and private) in the class. In addition to listing class definitions, describe how the classes interact. For example, does one class contain other classes as members? Does a class function take another object as an argument? What kinds of connections can components have? Instead of a data structure and a call tree, design a set of classes and their connections. You should also include a readme file. It is a text file listing the names of the header files included in your submission.

The provide command is

`provide comp15 proj1part1` ...all of your .h files...

## Part Two Deliverables

Part two is due March 1, 2013. You will implement the retrieval unit queue. It will process the input file and manage incoming orders. It will simulate the fetch/retrieval function described above. It will output a file of orders similiar to:

```
Order 123 for regular customer arrived at time 0
fetch time 15 minutes, pack time 22 minutes
the order was added to packaging queue at time 15

Order 127 for supreme customer arrived at time 12
fetch time 3 minutes, pack time 12 minutes
the order was added to packaging queue at time 18
```

The provide command is

`provide comp15 proj1part2` ...all of your .h files, .cpp files, and a main.cpp to test your code.

## Part Three Deliverables

Part three is due March 6, 2013. It will include all .h and .cpp files as well as a main.cpp file. It will also include a readme file, not only will it detail the include files, but also how to compile the files.

The provide command is

`provide comp15 proj1part3` ...all of your files...