CMSC 204

Assignment #2

Spring 2016

Office Depo an office supply retailing company donates its surpluses to colleges at the end of each year. Volunteers will help deliver packages of supplies to representative of colleges (Recipient of supplies). Write an application to simulate delivering packages from the container of packages by the volunteers to recipients.

**Concepts tested:**

Generic Queue, co

Generic Stack,

Exception handling,

**Data Elements**

**Volunteer-** Holds the relevant information for a Volunteer: *name*

**DonationPackage-** Holds the information of the package to be donated: *description, weight*

**Recipient-** Holds the information of the recipients: *name*

**Data Structures**

1. Create a generic queue class called *MyQueue* to implement the queue of *volunteers* and *recipients* line. *MyQueue* will implement the *QueueInterface* given you.
2. Create a generic stack class called *MyStack* to place the *DonationPackage* on a *Container*. *MyStack* will implement the *Stack Interface* given you.

There will be a *VolunteerLine* class, a *RecipientLine* Class and a *Container* class The *VolunteerLine* class will implement the *VolunteerLineInterface* , the *RecipientLine* class will implement the *RecipientLineInterface* class and the *Container* class will implement the *ContainerInterface* class

**Limit the size of each collection to 5.**

**Data Manager**  
The *DonationManager* class will manage adding a new package to the container, a new volunteer to the volunteer queue line , a new recipient to the recipient queue and donating package by the volunteer to the recipient. *DonationManager* will implement the interface *DonationManager* *Interface*.

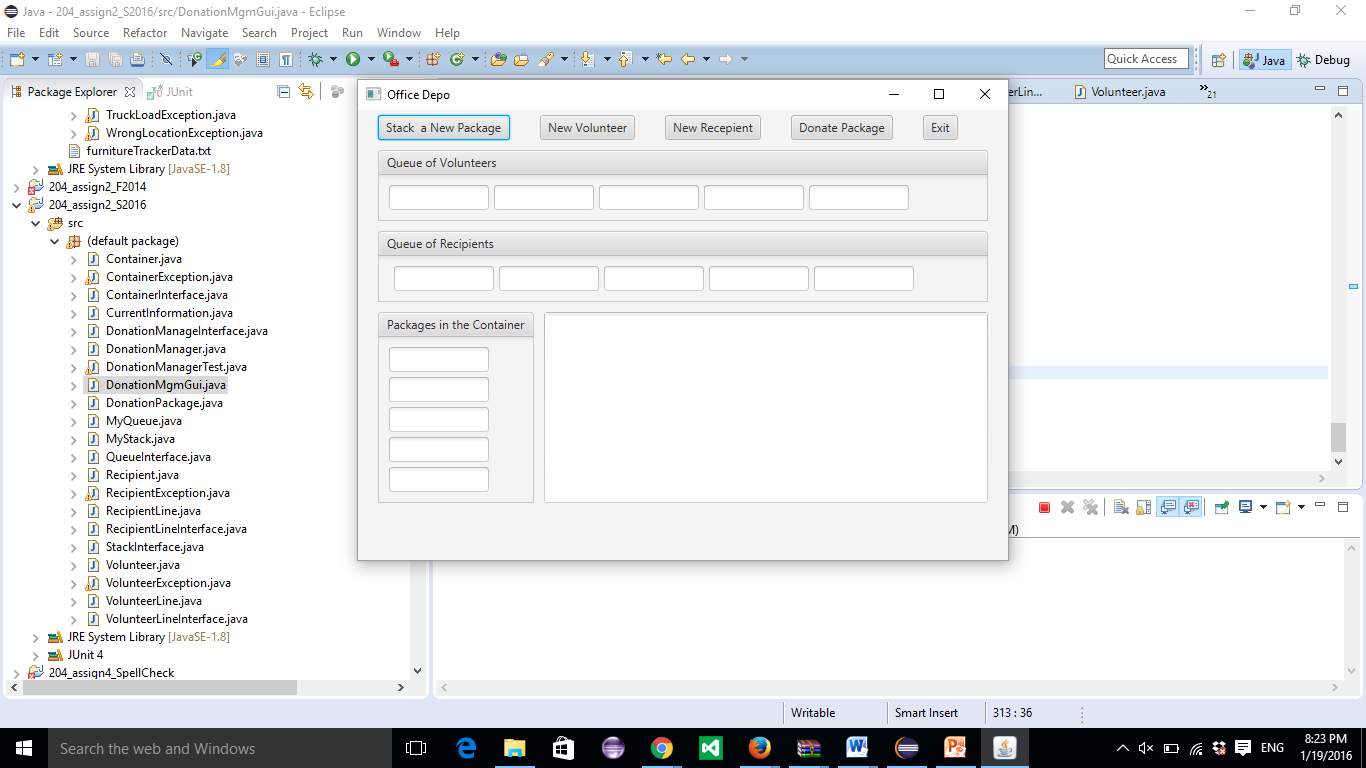
**You may add additional methods and attributes to your classes**

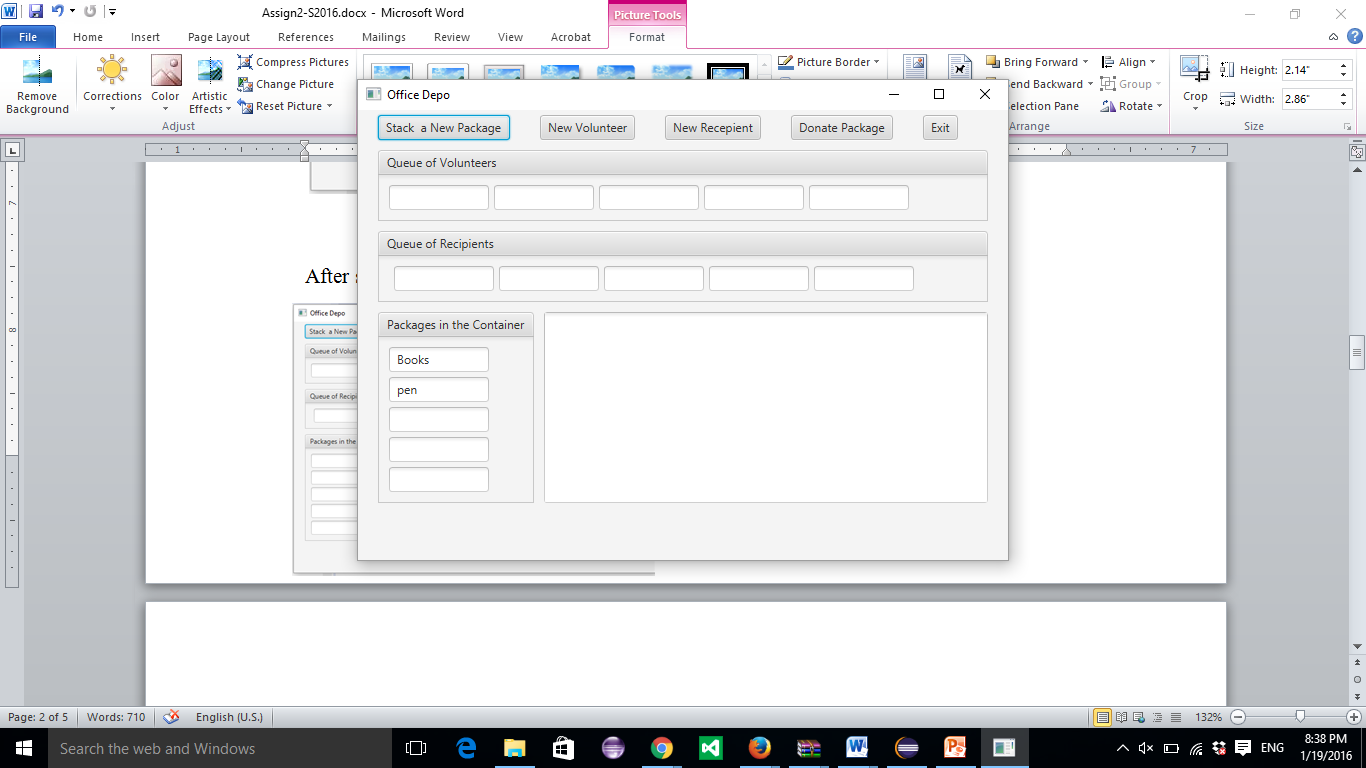
**GUI Driver**

1. Initially there is no package on the container, any volunteers and recipients on the queue.
2. Allow the user to load new package to the container and add new volunteers to volunteer’s line and new recipient to recipients’ line.
3. Do not allow packages heavier than 20lbs to be added to the container.
4. Allow the user to simulate process of donating the package from the container by volunteer on the queue to the recipient on the queue. When a volunteer helps donating the package to the recipient, he/she will be back to the queue of volunteers to continue the process of donation.
5. Provide three exception classes:
   1. RecipientException: If the user attempts to add new Recipient and the recipient line is full (5 recipients).
   2. VolunteerException – If the user attempts to add new Volunteer and the volunteer line is full (5 volunteers).
   3. ContainerException – If the user attempts to add new Donation Package and the Container line is full (5 packages).

**Examples:**

At startup



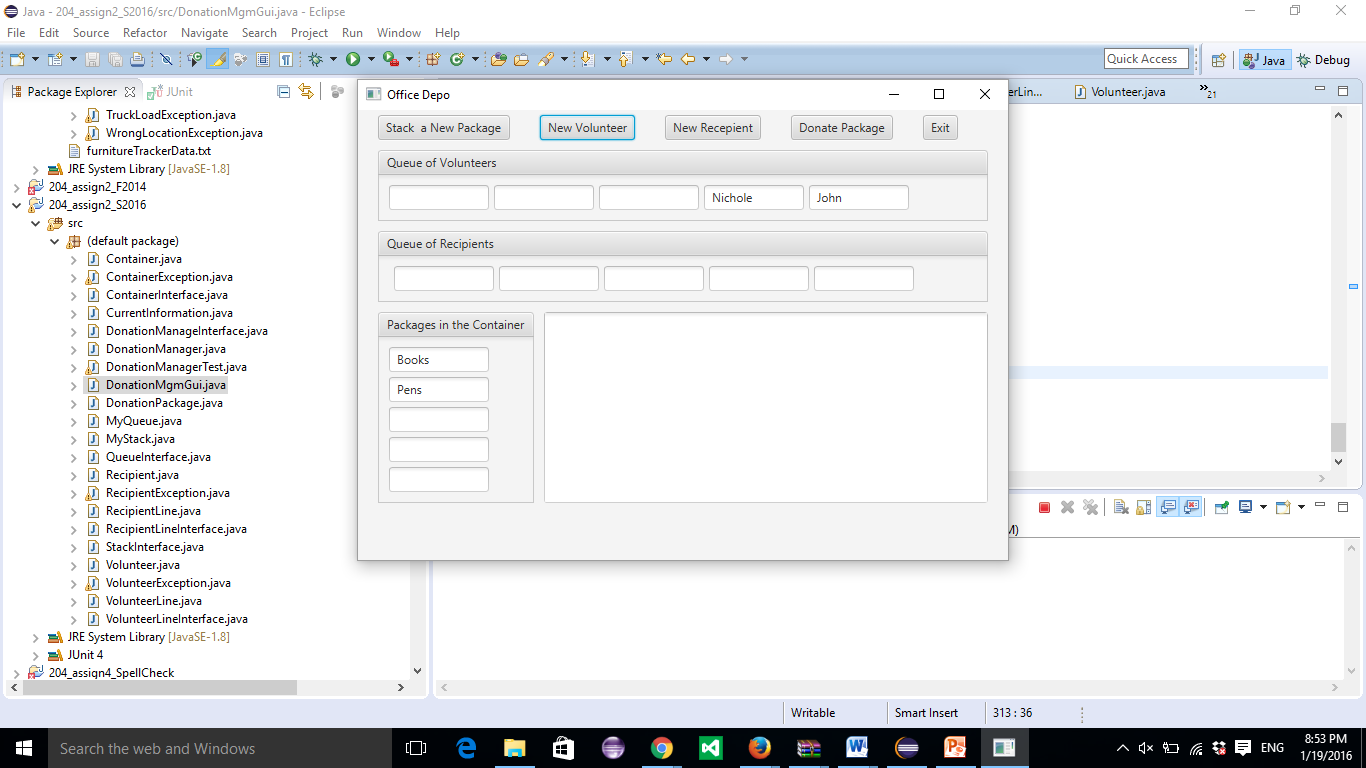
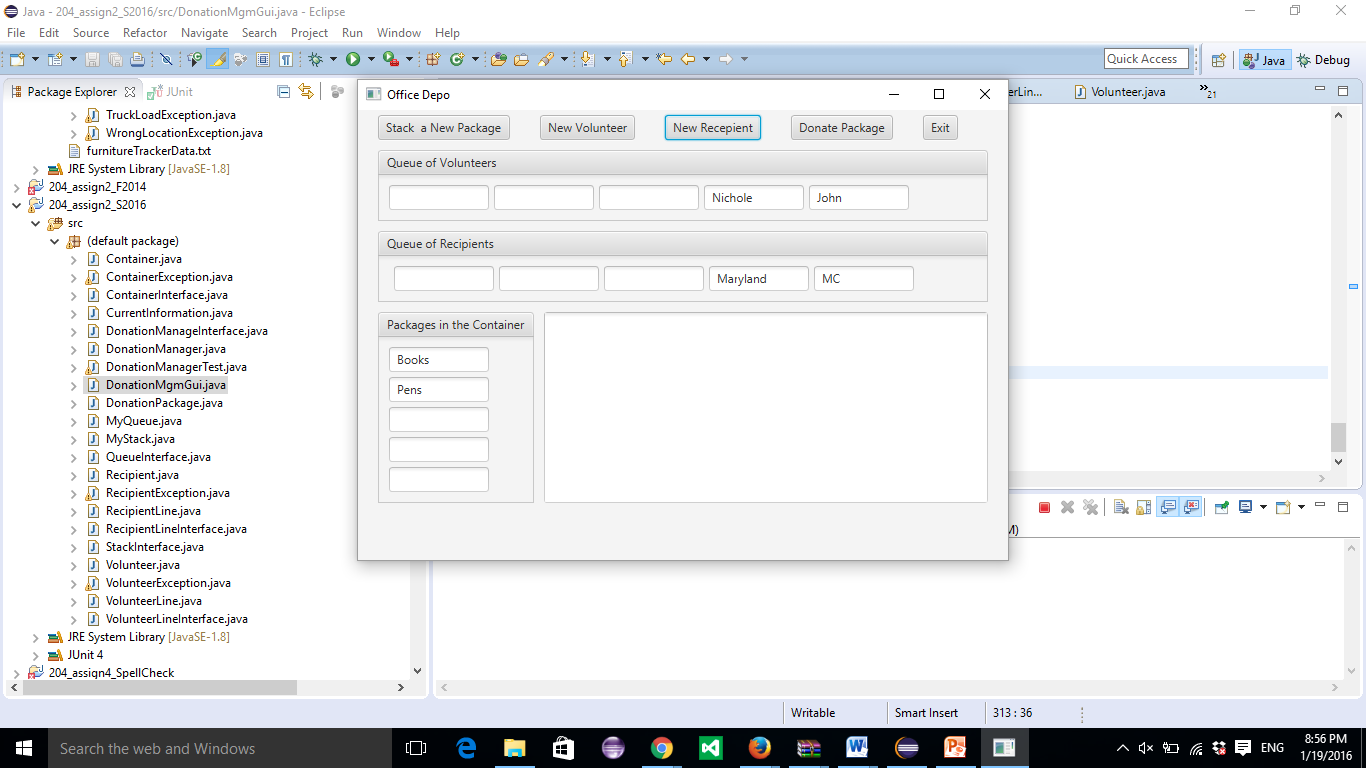


After selecting “Stack New Package”

Pen(first) and Books (second).

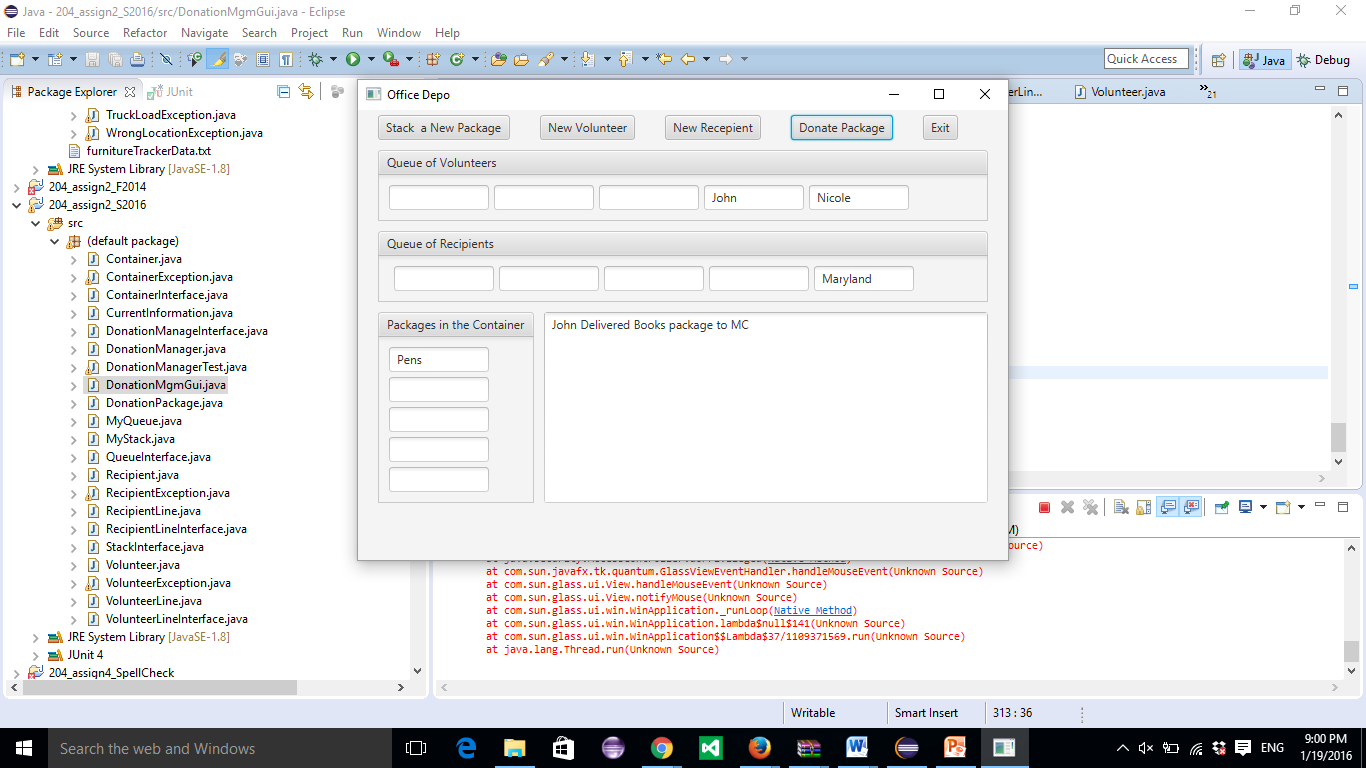
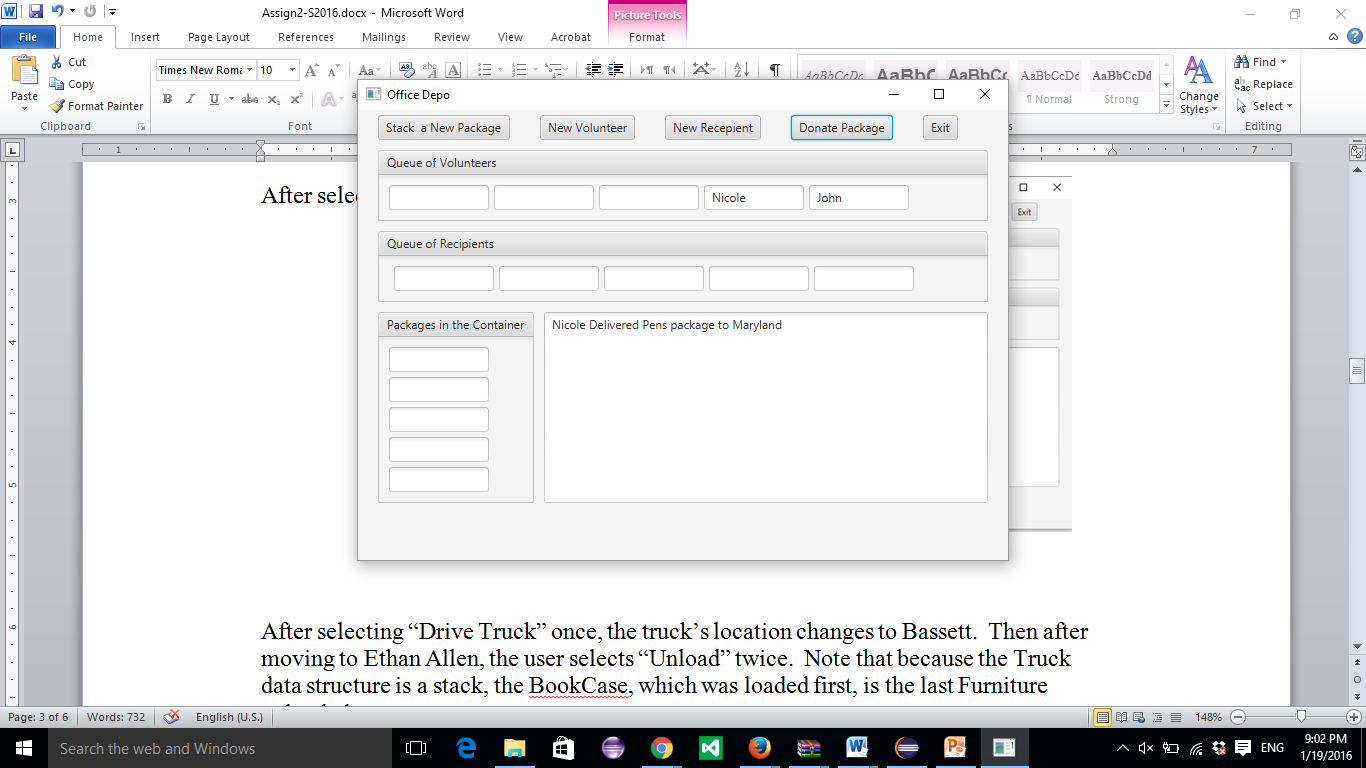
After adding new Volunteers John (first), after adding new Recipient MC,

Nicole (second) Maryland

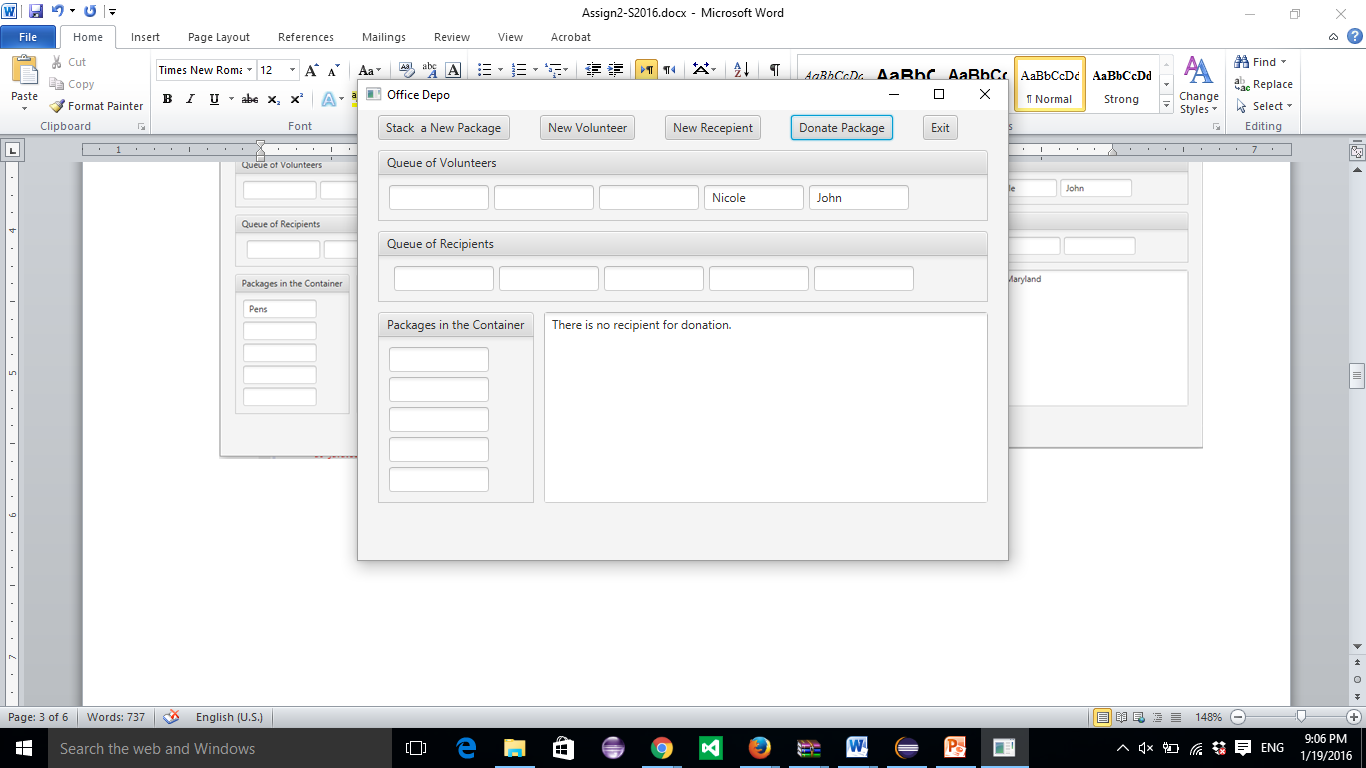


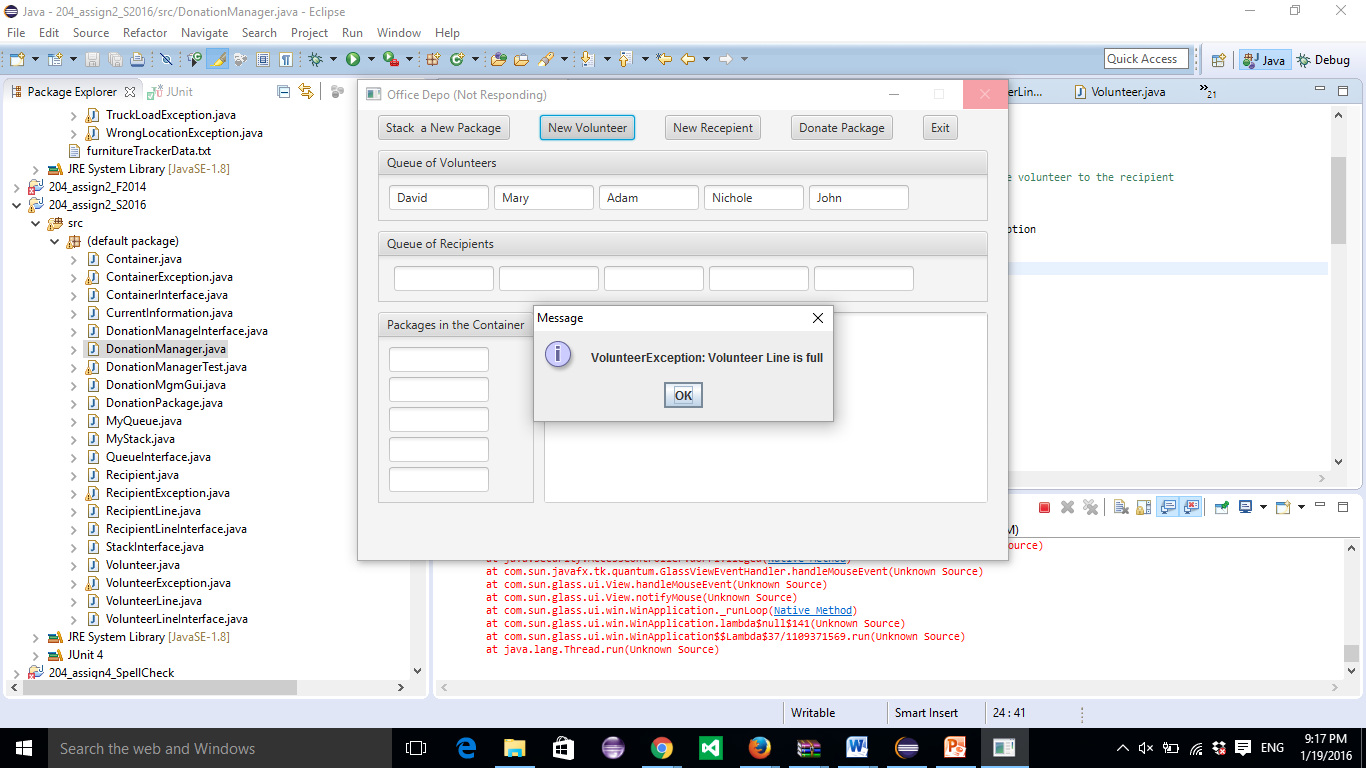
After selecting Donate Package After selecting Donate Package Second time

First Time



After selecting Donate Package Third time



 If the user tries to add a new volunteer:

**Program Grade Sheet**

Assignment #2

CMSC 204

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## DOCUMENTATION

# CheckList for Assignment 2 is included and completed 1 pt \_\_\_\_\_

# Class documentation - Javadoc for all user created classes 4 pt \_\_\_\_\_

Class Description, @author

Method description, @param, @return

Test Cases 6 pt \_\_\_\_\_

JUnit Test Class

Implement STUDENT test for DonationManagerTest

Create a MyQueueTest and MyStackTest

UML Diagram 4 pt \_\_\_\_\_

Lessons Learned 5 pt \_\_\_\_\_

In 3+ paragraphs, highlight your lessons learned and learning experience from working on this project. How did you do? What have you learned? What did you struggle with? How will you approach your next project differently?

**PROGRAMMING**

Internal class documentation (within source code) 6 pts \_\_\_\_\_

Class description using Javadoc

Author’s Name, Class, Class Time, @author

Methods commented using Javadoc, @param, @return

Compiles and Runs without errors 6 pts \_\_\_\_\_

Program user interface

Clear to user how data is to be entered 4 pts \_\_\_\_\_

Output is easy to understand 4 pts \_\_\_\_\_

Accuracy

Received correct output

1. Public tests (given to you and those you wrote) 10 pts \_\_\_\_\_
2. Private tests 10 pts \_\_\_\_\_

Program Details

Data Structures 10 pts \_\_\_\_\_

1. Stack class for Container
2. Queue for volunteer line
3. Queue for Recipient line

Data Elements – DonationPackage, Volunteer, Recipient

1. Contains required attributes 6 pts \_\_\_\_\_
2. Contains toString any other necessary methods

Data Manager - DonationManager 10 pts \_\_\_\_\_

1. Implements DonationManager Interface
2. Creates Volunteer, Recipient, and DonationPackage objects
3. Donate Package implemented
4. Correctly handles exceptions

GUI classes: 14pts \_\_\_\_\_

1. Displays information for the volunteers, recipients and packages

Correctly as queue and stack.

1. Provides buttons for user actions

Total 100 pts \_\_\_\_\_