CS213 **Dr. Lily Chang**

Project #1 (45 points)

Due Date

• Thursday, September 24, by 11:59pm.

Submission

1. Group Submission (one copy per team)

(a) You must designate a submitter (one of the team members) and submit the zipped project folder to

Canvas.

(b) You must include both team members' names in the comment block on top of EVERY Java file.

(c) Your project folder must include the following subfolders/files for grading.

• Source folder, including all Java files [25 points]

o A testbed main, in the ShoppingBag class. [5 points]

• Javadoc folder, including all the files generated. [5 points]

• Test design document, typed with a document editor. [5 points]

2. Individual Submission (everyone must submit a copy)

• Personal time log, using the template posted on Canvas. [5 points]

Project Description

In this project, you will implement an array-based container class ShoppingBag and use it to hold a list of grocery

items. An instance of the ShoppingBag class is a growable bag with an initial capacity of holding 5 grocery items,

and automatically grows (increases) the capacity by 5 whenever it is full. You CANNOT use the ArrayList class,

or 0 points!

Your program shall allow the user to add, remove and display the items in the bag. In addition, your program shall

provide the functionality of checking out the grocery items in the bag. Checking out will empty the bag and display

the details of the grocery items, including the list of grocery items and the total amount paid. Each grocery item in

the list shall include the item name, unit price and taxable or nontaxable. The total amount paid is the sum of the unit

price in the list plus the sales tax, which is 6.625%. The sales tax only applies to the taxable items.

Your program shall read the commands from the standard input (keyboard), process the commands and display the

results to the standard output (console.) When your project runs, it shall display "Let's start shopping!". Next,

it will take the commands from the input and process the commands continuously until the user enters the “Q”

command to quit. All commands are case-sensitive, which means the commands with lowercase letters are invalid!

In addition, you are required to deal with bad commands not supported. You may need to discard the rest of the

command line if a bad command is read.

The data tokens of a command line from the standard input are separately by white spaces, i.e., space(s), tab or

newline. All commands are single letter commands. For the Add and Remove commands, you can always assume

the next 3 data tokens are the item name, unit price followed by a true or false indicating taxable or not taxable. For

simplicity, an item name is a single string. If a multi-word item name is used, underscores “_” or hyphens “-“ will be

used to connect the words. The details of the valid commands are listed below.

• A – add a grocery item to the bag. For example, a command line A milk 2.99 false will add the grocery item

“milk” with the unit price “2.99” to the bag. The item is not taxable. If the bag is full, the bag automatically

grows the capacity by 5. Display “milk added to the bag.” after the item was added.

• R – remove an item from the bag. For example, R milk 2.99 false will remove the nontaxable grocery item

“milk” with the price “2.99” from the bag. If there are more than one matching item, remove the first item found

in the bag. If the remove was not successful, display “Unable to remove, this item is not in the

bag.”; otherwise, display “milk 2.99 removed.”.

• P – display all items in the bag on the console. If the bag is empty, display “The bag is empty!”; otherwise,

display "**You have X item(s) in the bag: " where X is the number of items. Your program then displays

the list of the grocery items and displays "**End of list" after the last item.

• C – checking out the grocery items in the bag. If the bag is empty, display “Unable to check out, the bag

is empty!”; otherwise, display “**Checking out X item(s): ”, where X is the number of items in the bag.

Your program then lists all items with the sum of the prices “*Sales total: $xxx.xx”, and then displays:

“*Sales tax: $xx.xx”, “*Total amount paid: $xxx.xx.” Note that a floating-point number is displayed

with 2 decimal places.

• Q – if the bag is not empty, automatically check out all items; in all cases, display “Thanks for shopping

with us!” and stop the program execution.

Program Requirement

1. You MUST follow the software development ground rules. You will lose points if you are not following the rules.

2. There are some test cases (Sample Input) at the end of this document. If you enter the test cases in the same order,

your program should generate the same output in the Sample Outputsection. The graders will be using the Sample

Input to test your program. You will lose 2 points for each error output produced.

3. You are required to log your times working on this project with the template provided on Canvas. The time log

is an individual assignment. You will lose 5 points if the log is not submitted. If the times and comments are

not properly logged, you will only get partial credits. You must type, hand-writing is not acceptable.

4. Each Java class must go in a separate file. -2 points if you put more than one Java class into a file.

5. Your program MUST handle bad commands! The commands are case-sensitive. -2 points for each bad command

not handled.

6. Your project MUST include at least 4 classes described below. -5 points for each class missing.

(a) GroceryItem class.

```
public class GroceryItem {
 private String name;
 private double price;
 private boolean taxable;
...
 public boolean equals(Object obj){ }
 public String toString() { }
...
}
```

• This class defines the abstract data type GroceryItem, which encapsulates the data fields and methods of

a grocery item. You CANNOT add other data members, and you CANNOT change the signatures of the

toString() and equals() methods. -2 points for each violation. You should define a parameterized

constructor. You can add other methods if necessary.

• You CANNOT do I/O in this class, i.e., read from or write to the console. -2 points for each violation.

• toString() method returns a string representing the data fields of an object with the format

itemName: $xx.xx : tax free.

The unit price shall be displayed with 2 decimal places. -2 points if the format is not set properly. You

can use Sytem.out.printf() method or the format() method in the DecimalFormat class to format a

floating point number.

• equals() method returns true only if all the data field values of the two objects are the same.

(b) ShoppingBag class.

```
public class ShoppingBag {
 private GroceryItem[] bag; // array-based implementation of the bag
 private int size; // number of items currently in the bag
public ShoppingBag() { }
private int find(GroceryItem item) { } // helper method to find an item
private void grow() { } // helper method to grow the capacity
public void add(GroceryItem item) { }
public boolean remove(GroceryItem item) { }
public double salesPrice() { } // sales total; the sum of the prices in the bag
public double salesTax() { } // sales tax total of the taxable items in the bag
public void print() { }
}
```

• This is the container class that defines the abstract data type for a shopping bag and its operations. A

shopping bag holds an array of grocery items. You CANNOT add other data members, and you CANNOT

change the signatures of the methods. -2 points for each violation. You must implement all the methods

listed, -2 points for each method not implemented or not used. You can add other methods if necessary.

• You CANNOT do I/O in this class, except the print() method, -2 points for each violation.

• The remove() method calls the helper method find() and finds the index of the grocery item to be removed.

Move the last item in the array to replace the removing item with the index, and set the reference of the

last item to null.

• You MUST design the test cases for testing add(), remove(), grow() and salesTax() methods. The test

cases must be documented in a test document. You must follow the instructions in the "Test Specification"

section of the Software Development Ground Rules. You will lose 5 points if you do not submit the test

document, or not follow the format.

• You must write a test bed main for this class to implement the test cases in the test document, -1 points

for each method not tested. In the testbed main, you MUST print out the results to the console showing

the test cases are passed.

(c) Shopping class for handling the I/O.

```
public class Shopping {
...
 public void run() { }
...
}
```

• This class is the user interface class that handles the input commands, output data and messages. You

must define a run() method or you will lose 5 points. The main method in RunProject1.java will call

the run() method here.

• You can define the data members you need and define some private methods to handle the commands.

Make sure you follow the ground rules and have a good programming style.

(d) RunProject1 class (this is a driver class to run your Project 1)

```
public class RunProject1 {
 public static void main(String[] args) {
 new Shopping().run();
 }
}
```

7. You must follow the instructions in the Software Development Ground Rules and comment your code. You are

required to generate the Javadoc after you properly commented your code. Your Javadoc must include the

documentations for the constructors, private methods and public methods of all Java classes. Generate the

Javadoc in a single folder and include it in your project folder to be submitted to Canvas. You will lose 5 points

for not including the Javadoc.

Sample Input

```
r meat 2.99 false
a meat 2.99 fals
e
P
R toast 2.99 false
A toast 4.95 false
```

A jam 3.95 false

A jam 3.95 false P

R jam 3.95 false P

Blah blah blah

a meat 2.99 false

A jam 2.59 false

A jam 2.59 false

A jam 2.59 false

A toast 4.95 false

A jam 3.99 false P

R jam 2.59 false P

R toast 4.95 false PcC

R toast 4.95 false

A toast 4.95 false C

A milk 2.5 false

A salmon 5.99 false

A toilet_papers 7.95 true

A kitchen_towels 10 true

A bagels 3.99 false

A garbage_bags 4.59 true

A laundry_detergent 12.99 true pP

A chicken 6.75 false

A pork 4.59 false

A shrimp 13 false

A masks 15.59 true

A sanitizer 2.99 true

A bleach 3.95 true PCqQ

You shouldn't process this line!

Sample Output

Let's start shopping!

Invalid command!

Invalid command!

The bag is empty!

Unable to remove, this item is not in the bag.

toast added to the bag.

jam added to the bag.

jam added to the bag.

**You have 3 items in the bag.

·toast: $4.95 : tax free

·jam: $3.95 : tax free

·jam: $3.95 : tax free

**End of list

jam 3.95 removed.

**You have 2 items in the bag.

·toast: $4.95 : tax free

·jam: $3.95 : tax free

**End of list

Invalid command!

Invalid command!

jam added to the bag.

jam added to the bag.

jam added to the bag.

toast added to the bag.

jam added to the bag.

**You have 7 items in the bag.

·toast: $4.95 : tax free

·jam: $3.95 : tax free

·jam: $2.59 : tax free

·jam: $2.59 : tax free

·jam: $2.59 : tax free

·toast: $4.95 : tax free

·jam: $3.99 : tax free

**End of list

jam 2.59 removed.

**You have 6 items in the bag.

·toast: $4.95 : tax free

·jam: $3.95 : tax free

·jam: $3.99 : tax free

·jam: $2.59 : tax free

·jam: $2.59 : tax free

·toast: $4.95 : tax free

**End of list

toast 4.95 removed.

**You have 5 items in the bag.

·toast: $4.95 : tax free

·jam: $3.95 : tax free

·jam: $3.99 : tax free

·jam: $2.59 : tax free

·jam: $2.59 : tax free

**End of list

Invalid command!

**Checking out 5 items.

·toast: $4.95 : tax free

·jam: $3.95 : tax free

·jam: $3.99 : tax free

·jam: $2.59 : tax free

·jam: $2.59 : tax free

*Sales total: $18.07

*Sales tax: $0.00

*Total amount paid: $18.07

Unable to remove, this item is not in the bag.

toast added to the bag.

**Checking out 1 item.

·toast: $4.95 : tax free

*Sales total: $4.95

*Sales tax: $0.00

*Total amount paid: $4.95

milk added to the bag.

salmon added to the bag.

toilet_papers added to the bag.

kitchen_towels added to the bag.

bagels added to the bag.

garbage_bags added to the bag.

laundry_detergent added to the bag.

Invalid command!

**You have 7 items in the bag.

·milk: $2.50 : tax free

·salmon: $5.99 : tax free

·toilet_papers: $7.95 : is taxable

·kitchen_towels: $10.00 : is taxable

·bagels: $3.99 : tax free

·garbage_bags: $4.59 : is taxable

·laundry_detergent: $12.99 : is taxable

**End of list

chicken added to the bag.

pork added to the bag.

shrimp added to the bag.

masks added to the bag.

sanitizer added to the bag.

bleach added to the bag.

**You have 13 items in the bag.

·milk: $2.50 : tax free

·salmon: $5.99 : tax free

·toilet_papers: $7.95 : is taxable

·kitchen_towels: $10.00 : is taxable

·bagels: $3.99 : tax free

·garbage_bags: $4.59 : is taxable

·laundry_detergent: $12.99 : is taxable

·chicken: $6.75 : tax free

·pork: $4.59 : tax free

·shrimp: $13.00 : tax free

·masks: $15.59 : is taxable

·sanitizer: $2.99 : is taxable

·bleach: $3.95 : is taxable

**End of list

**Checking out 13 items.

·milk: $2.50 : tax free

·salmon: $5.99 : tax free

·toilet_papers: $7.95 : is taxable

·kitchen_towels: $10.00 : is taxable

·bagels: $3.99 : tax free

·garbage_bags: $4.59 : is taxable

·laundry_detergent: $12.99 : is taxable

·chicken: $6.75 : tax free

·pork: $4.59 : tax free

·shrimp: $13.00 : tax free

·masks: $15.59 : is taxable

·sanitizer: $2.99 : is taxable

·bleach: $3.95 : is taxable

*Sales total: $94.88

*Sales tax: $3.85

*Total amount paid: $98.73

Invalid command!

Thanks for shopping with us!