Computing and Algorithms I

Project 1

CS-101 **Fall, 2017**

Overview

In this project you are practicing sequential control of flow and using the java API. You will not be writing any code for selection, loops, arrays, nor will you be creating your own classes to define objects. Note that you will not use any of these reserved words: if, for, while, or switch. Furthermore, the design of the program has already been done. You will actually be converting the given design into java code with good style and comments.

Introduction

In project 1 you are simulating putting together an order for purchasing coffee. The JavaLumen corporation sells coffee exclusively in 2 pound bags. These bags are placed in boxes for shipment. Large boxes store 12 bags of coffee, medium boxes store 6 bags of coffee, and small boxes store 3 bags of coffee. JavaLumen charges $2.33 for each bag of coffee. You must also pay for all the boxes used to store the coffee for shipping. JavaLumen will always fully pack large and medium boxes. It will probably be the case that one small box is not full. The charges for boxes are: $1.51 for a large box, $0.96 for a medium box, and $0.57 for a small box. Note that you will use the division operator to determine how many large boxes are necessary. The remainder, or modulus, operator will determine the remaining bags. This same process will work for the medium boxes. For the number of small boxes, add 2 to the remaining bags and divide by 3. Note that the only possible value for the remaining bags is one of 0, 1, 2, 3, 4, or 5. The number of small boxes you will use is 0, 1, or 2. The following table shows how this process will work.

Number of bags	bags + 2	(bags + 2)/3	Number of Boxes Needed
0	2	0	0
1	3	1	1
2	4	1	1
3	5	1	1
4	6	2	2
5	7	2	2

For this project you will use JOptionPane methods (showInputDialog and showMessageDialog) to do input and output (I/0). You will also use other classes to complete this assignment. For the date you will use the SimpleDate class from chapter 3 of the text. You will need to use DecimalFormat and wrapper classes for this assignment. Your program will have the user place a coffee order, giving a date, customer name, and amount of coffee. Your code will calculate the cost of the order and the date that the order will arrive (7 days after the order). The formatted order form will then be printed.

Included in the assignment is a Java file which is a template for the code you will write. Note in the template comments, one section of comments identifies the programmer and specifies the purpose of the class you are writing. There is also a required design which has two parts: a data table and an algorithm. The algorithm and data table for the main method have been written for you in the design document. These will also appear in comments immediately before the main method in your code. Note that you will code this problem using only one method, and that is main.

Input

You will use JOptionPane methods to input the following values: customer name, date of order (in 3 values: month, day, and year - all integers), number of pounds of coffee as a double. Remember that all values returned by showInputDialog are Strings; you will need to use methods from wrapper classes to convert numeric values from String.

Processing

The input from using JOptionPane is of course a String. Most values must be converted to numeric types. Note that having the pounds of coffee input as a double is silly, you would not design a program to accept input in that fashion. We are doing this to gain practice in changing between types. Use a method from Math to get the smallest integer value greater than or equal to the number of pounds. It is possible that this is an odd number. You will assume that the customer wants this much coffee, so an additional pound will be added to the order (since coffee is sold only in 2 pound bags). In practice, simply add 1 to the number of pounds, then divide by 2 to get the number of bags. To see that this works, if the user asked for one pound of coffee, $(1 + 1)/2$ equals 1, the number of bags to ship. If the user asked for 2 pounds of coffee, $(2 + 1)/2$ equals 1, the number of bags to ship.

Output

Show your result using JOptionPane. Your output should be in a tabular display similar to

```
Customer:          John Doe
Date of Order:     01/27/2015
Date of Arrival:   02/03/2015

Number of Bags Ordered: 53 . . . . . $123.49

Boxes Used:
                    4 Large . . . .  $6.04
                    0 Medium  . . .  $0.00
                    2 Small   . . .  $1.14

Total cost of order: . . . . . $130.67
Thank you for your JavaLumen Corporation coffee purchase!
```

Format the dollar amounts and dates as shown above. Use a DecimalFormat object for the formatting of money.

Style

Use white space (indentation, blank lines) to show the program structure. A meaningful class name is an important part of the style. It should describe the purpose of the class. A meaningful name will be supplied as part of the design. (In particular, Project1 is not a good class name, it says nothing about creating a coffee order, and it will not be the name in the design). Likewise, all variable and constant names will be meaningful and will follow naming conventions.

Deliverables

On Tuesday October 17 at the beginning of class you will turn in a document printed from a printer containing the Java source code (of course with all comments as described above). Your Java code will be printed in portrait, not landscape, mode. None of your lines in the file are allowed to extend past the right edge of the paper. In particular, this means do not have statements wrapping around to the next line (which often happens automatically). If you have multiple pages of code, they will be stapled into one package before the beginning of class.

You will submit your code (just the .java file) as a zip file using Blackboard before class begins. The submission will be in the project page for this project. Be sure to zip the code, do not use other forms of compression such as rar. The reason for zipping the code is that blackboard, for some reason and at various times, changes variable names in code. Projects not turned in via Blackboard and/or not turned in as a document will receive no credit.

Grading

If your program does not compile, the grade for this project will be 0. If your program compiles but does not produce the correct output, the total number of points awarded (as described below) will be cut in half.

The program will be worth 10 points.

Clear messages written using the JOptionPane methods to communicate with the user of the program will be worth 5 points.

The data table and algorithm copied correctly and well formatted are worth 5 points.

The program style is worth 5 points. Use white space (indentation and blank lines) to clearly show the structure of the program.

The total maximum score for the project is thus 25 points.

No program will be accepted after the due date.