Assignment #1 – Due July 14

**Object Oriented Programming**

The assignment can be done **individually** or in teams of **two**. Submit one assignment per team of two via Omnivox and NOT MIO. Assignments sent via MIO will be deducted marks. Assignments must be done alone or in groups and collaboration between individuals or groups is strictly forbidden.

**General Guidelines When Writing Your Programs:**

Include the following in comments at the top of your program:

// -----------------------------------------------------

// Assignment (include number)

// Written by: (include your name(s) and student ID(s))

// Short Description of your project/code and how you designed it.

// -----------------------------------------------------

Throughout your program, include comments in your program describing the main steps.

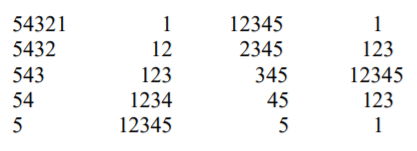
**QUESTION One: Printing Patterns (7 points)**

Write a Java program that prints one of the following patterns based on the user choice of a pattern number, which must be between 1 and 4, or 5 to quit (See examples below), and an input value, which must be larger than 0 and smaller than 10, and according to the following:

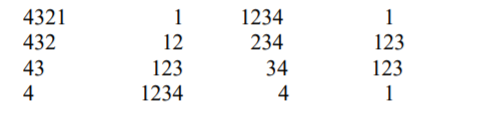
A) If the user enters any invalid pattern number, then the program should display a message indicating that input was invalid and request the user to either enters a correct pattern number or 5 to quit the entire program. That is, an entry of 5 would simply terminate the program.

B) Once the pattern number is correctly supplied, the program requests the user to enter the input value. If the user enters any invalid value that is not within the expected range, then the program should reject this entry and asks the user to re-enter another value; this would repeat indefinitely until a good value is entered.

C) Upon the entry of a good input value, the program must check whether this value is odd or even. If the user enters an odd number, then the program would draw a pattern that is similar to the following (i.e. the shown pattern is drawn if the user enters 5. You should notice that this is only an example; your program must allow for the general case for different appropriate values as indicated above.) You should notice that the patterns are actually quite similar whether the input is odd or even, with the exception of pattern # 4. See below:



If the user enters an even number, the pattern would look as one of the following (i.e. the shown pattern is drawn if the user enters 4. Again this is just an example; your program must work for the general case.)



**Restrictions:**

• You can use any combinations of the selection and repetition statements that we have seen in class. You must however write the needed code yourself (i.e. do not use any built-in methods that may perform what is needed.)

**General Algorithm:**

In general your program should:

1. Display a welcome message with your name(s) on it (i.e. “Welcome to Mike and Linda Triangle/Diamond Printer”).

2. Prompt the user for the pattern they want to display. Make sure the pattern requested is a legal one.

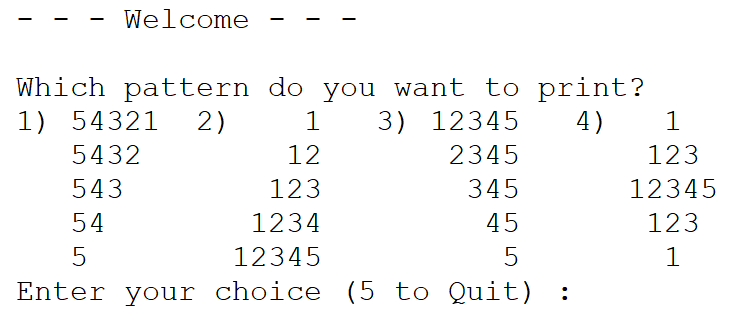
3. Prompt the user for the input value. The input value should be greater than 0 and less than 10. Keep prompting the user until the input value is valid.

4. Display the requested pattern.

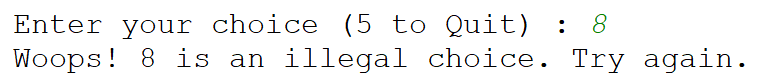
5. As long as the user wants to display another pattern your program repeats steps 2 to 4

6. Display a closing message

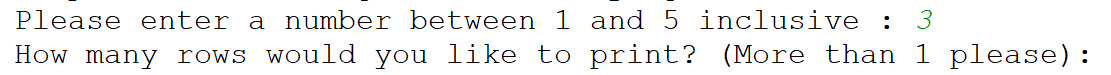
Display a welcome message and the options to choose



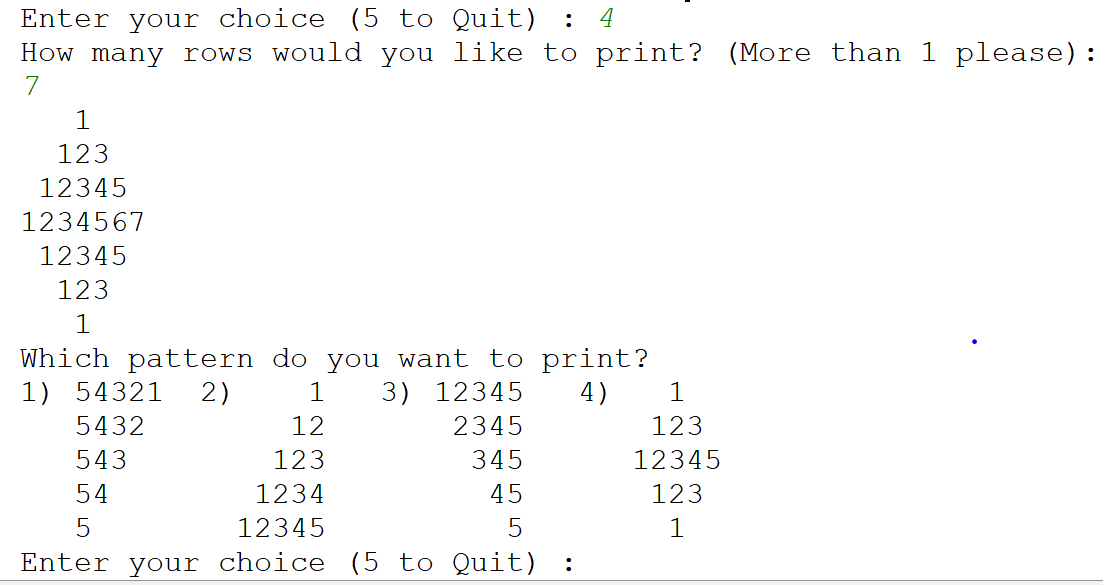
Incorrect choices will result in a message allowing the user to choose again:



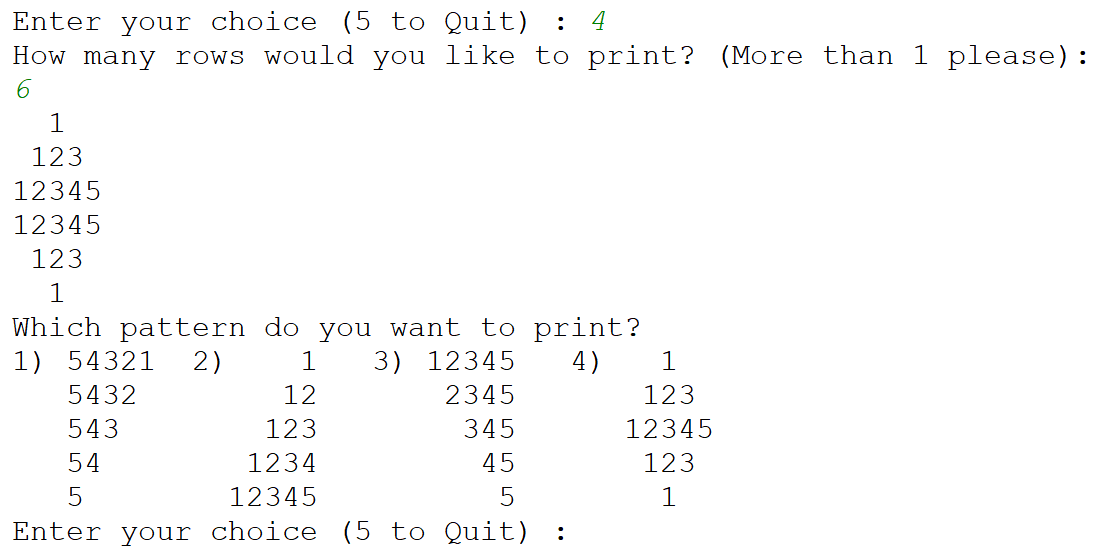
After correct number is chosen ask the user for the number of rows to print:



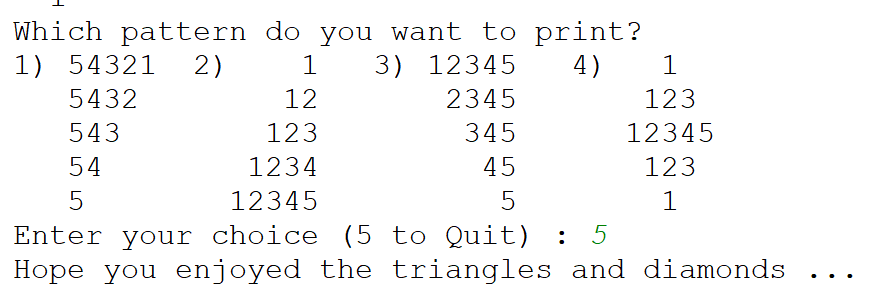
Users should be able to return to the original menu after the pattern is printed:



Pattern (4) should display the triangle differently depending if the number of rows entered is an even number or an odd one:

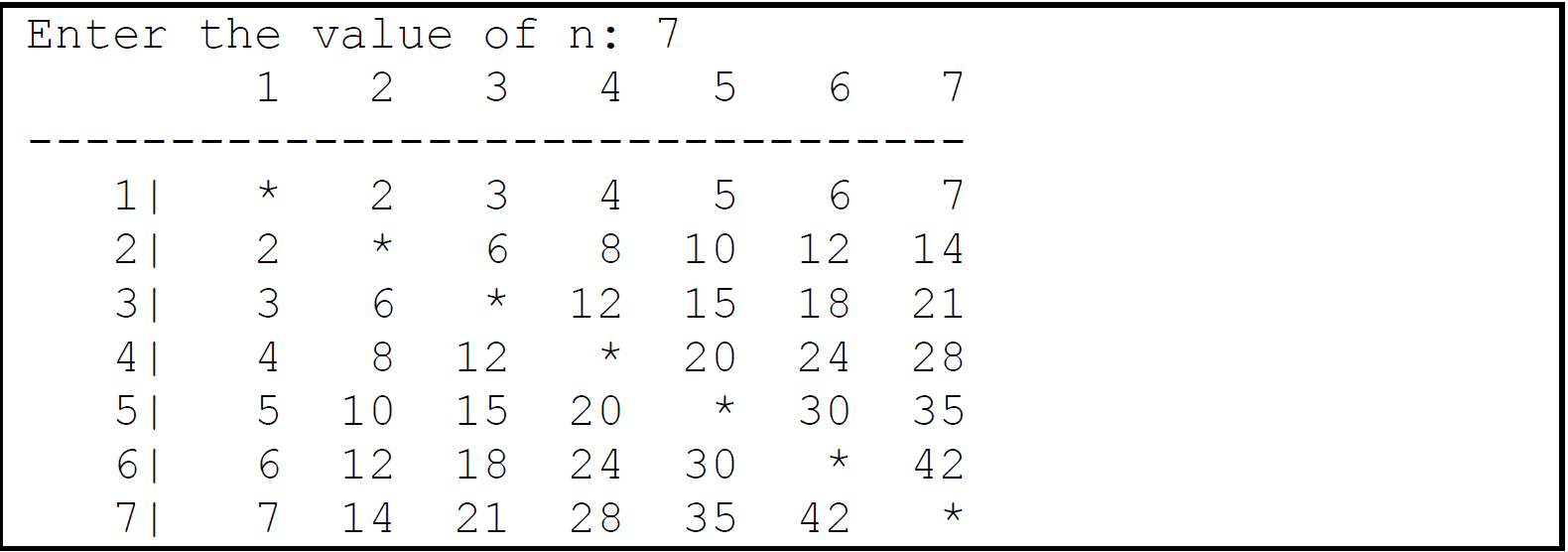


Include a leaving message for when the user wants to quit the application:



**QUESTION Two: Printing Multiplication Table (5 points)**

Write a program to display on the screen a multiplication table from 1 to n; where n is an integer entered by the user. The main diagonal of the table must be filled with the character ‘\*’. For example, if the user enters 7, your program should display**:**



**QUESTION Three: Classes (6 points):**

You are given the class Pizza (Pizza.java) which stores information about a single pizza. It contains the following:

-Private instance variables to store the size of the pizza (small, medium or large), the number of cheese toppings, the number of pepperoni toppings, and the number of mushroom toppings.

o Constructors:

-one that takes four arguments and sets all of the corresponding instance variables

-a default constructor, that takes no arguments and initializes all instance variables to the zero of their type.

- Public methods to get (accessor) and set (mutator) each instance variable individually.

- A public method called calcCost() that returns a double that is the cost of the pizza. Pizza cost is determined by: Small: $10 + $2 per topping Medium: $12 + $2 per topping Large: $14 + $2 per topping.

-A toString() method which takes no arguments and returns a string the pizza size, quantity of each topping, and the pizza cost as calculated by . For example, a large pizza with 1 cheese, 2 pepperoni and 1 mushroom toppings should cost $22.

- An equals() method which is redefined to test for the equality of the content of 2 objects of type Pizza.

**You are asked to write a driver which:**

- Creates 3 different pizzas: First one with the default constructor, and for the 2nd and third one prompt the user.

- Output the descriptions of the 3 pizzas.

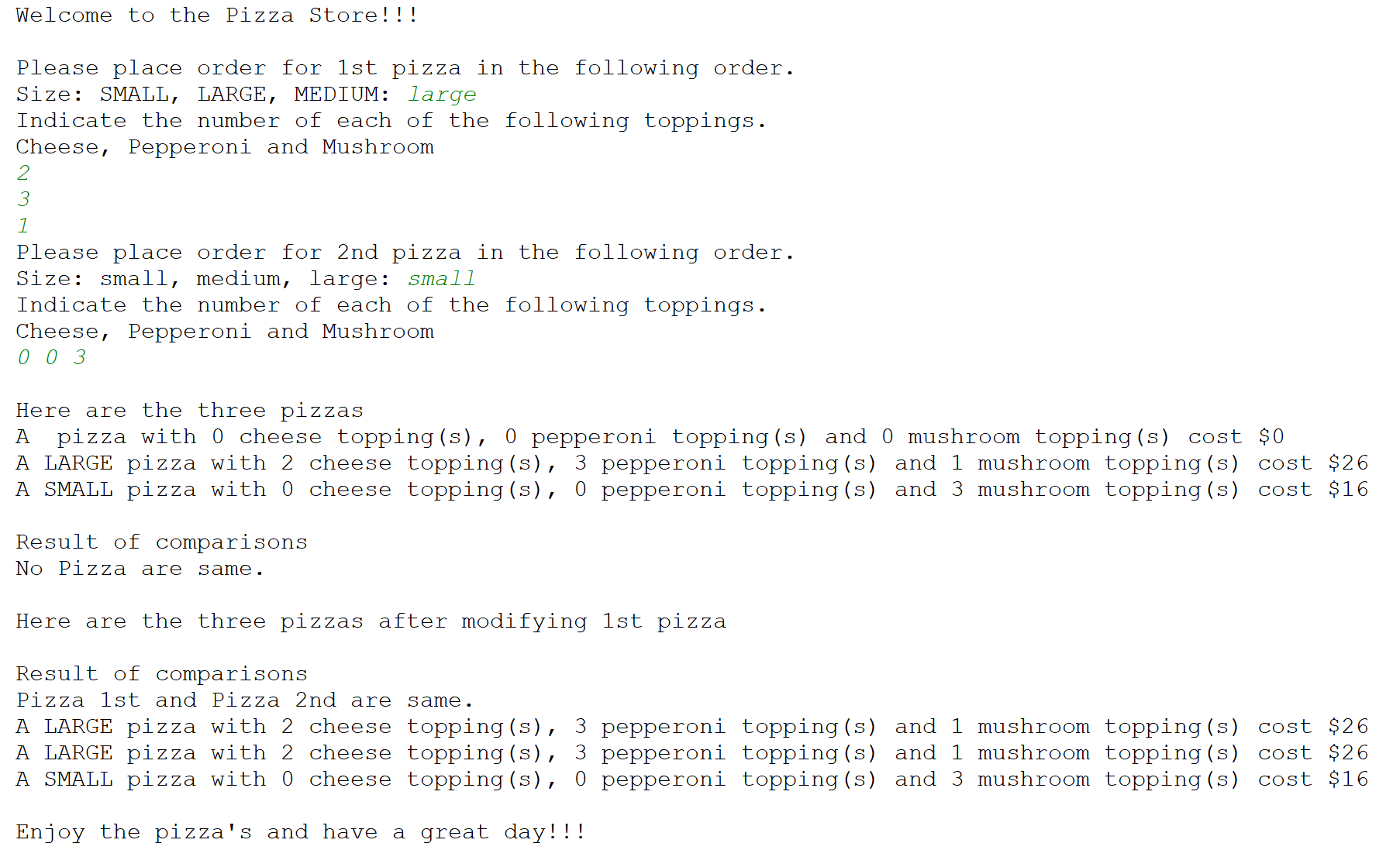
- Compare the 3 pizzas and display a message saying whether they are identical in content or not. Make sure you code test for the following scenarios when comparing the pizzas: that all three are the same, only 2 are the same, or none are the same.

- Change the first pizza’s content to be the same as one of the other ones.

- Compare the 3 pizzas again and display a message saying whether they are identical in content or not as well as their contents.

*Note*: Assume the user always enters a valid input.

Sample Output:



Grading Grid:

|  |  |  |
| --- | --- | --- |
| **Common Criteria for Questions 1,2,3 – (2 pts.)** | | |
| Comments & Programming Style which includes:  - Description of the program (authors, date, purpose)  - Significant names for variables  - Description of variables  - Description of the algorithm | 2 | pts. |
| **Question 1 – (7 pts).** | | |
| No bugs occur | 2 | pts. |
| Menu operation/error checking | 1 | pt. |
| Clarity of output | 2 | pts. |
| Correct Patterns | 2 | pts. |
| Format of output | 2 | pts. |
| **Question 2 – (5 pts).** |  |  |
| Correct Output | 3 | pts |
| No bugs | 2 | pts |
| **Question 3 – (6 pts.)** | | |
| Create the 3 pizza objects | 1 | pts. |
| Output description of pizzas | 1 | pt. |
| Compare pizzas | 2 | pts. |
| Change content of one of the pizzas | 2 | pts. |
| **TOTAL** | **20** | **%pts** |