Programming Project 6 – Ice Cream Shop

Note: When you turn in an assignment to be graded in this class, you are making the claim that you neither gave nor received assistance on the work you turned in (except, of course, assistance from the instructor or teaching assistants).

Mal's Ice Cream Shop sells ice cream and sundaes. They need a system to keep track of customer's purchases. The shop needs some abstraction to keep track of their customers and the ice cream that those customers purchase. You will design a data type using abstraction, that models ice cream options and another one that models a customer.

The name of the ice cream Java class is **`Ice Cream.java`**. The characteristics that define an ice cream selection are the following:

Size: SINGLE, DOUBLE, TRIPLE, SUNDAE

Flavor: VANILLA, CHOCOLATE, ROCKY_ROAD, STRAWBERRY, MINT_CHOCOLATE_CHIP, OREO, BUTTER_PECAN

Sauce: NONE, HOT_FUDGE, CARAMEL, STRAWBERRY

Topping: NONE, SPRINKLES, NUTS, CHOCOLATE_FLAKES, GUMMIES, CANDY

ToppingSun: NONE, WHIPPED_CREAM_AND_CHERRY

All the fields above will be expressed as Java Enumeration Types to restrict the values that can be assigned to these instance variables.

The parameterized constructor should have the following definition:

```
IceCream(Size size)
```

This constructor will only pass in the size of the ice cream. It will set default values for all the other characteristics, which are the first value in the enumerated lists. The default constructor will assign the default values for all the characteristics of the ice cream.

Once the IceCream object is created, the program will then ask the user to enter the flavor of the ice cream. Once the flavor has been selected, the program will ask the user for the sauce they would like, and the program will ask which topping they would like. If the user ordered a sundae, then ToppingSun will be assigned to WHIPPED_CREAM_AND_CHERRY, else it will be NONE.

The IceCream class will have accessors and mutators for each attribute/characteristic as well as a toString method that prints each characteristic of the IceCream object indented by a tab on its on line. You will start the printing with a new line and then a tab.

The other data type that needs to be defined represents a customer for the shop. The name of the Java class is **`Customer.java`**. The important characteristics of a customer for the application are the following:

firstName: String

lastName: String

phone: String

email: String

iceCream: ArrayList<IceCream>

When a new customer orders ice cream, they must provide their full name, phone number and email. You will use this data to create a customer object. Each time a user orders an ice cream option, that ice cream will be added to the list of ice creams for the customer's order. This list will contain objects of type IceCream.

The default constructor will place null into each of the String attribute fields when called and create an empty array list in the iceCream attribute.

Here is the definition for the parameterized constructor:

```
public Customer (String lastName, String firstName, String phone,
String email)
```

This constructor will initialize all attributes making sure to initialize the values passed in for the first four attributes.

You will have an accessor and mutator for the firstName, lastName, phone, and email attributes. You will have an orderIceCream() method that will add an IceCream object to the Customer object. It will also increment the numIceCream attribute. You will have a getNumIceCream that will let you know how many IceCream objects are associated with a Customer object. You will have a getIceCream method that will return the ArrayList of IceCreams associated with the Customer object.

Here are two examples of what a typical output would be from calling the toString() method for a customer with two ice cream selections and with one ice cream selection. Note that the Customer's toString() method must also call the toString() method for each IceCream object the customer has.

```
Caroline Budwell                                      Zach Whitten
804-937-8111                                          555-222-1111
ccbudwell@vcu.edu                                     zwhitten@vcu.edu
Ice Cream Order:                                      123 Maple Avenue
                                                      Ice Cream Order:
       DOUBLE
       ROCKY_ROAD                                            SINGLE
       NONE                                                  OREO
       NONE                                                  NONE
                                                             CANDY
       SUNDAE                                                NONE
       VANILLA
       HOT_FUDGE
       SPRINKLES
       WHIPPED_CREAM_AND_CHERRY
```

Make sure that you use the UML Class Diagrams below to build your classes. You will test your classes using the **IceCreamTest.java**, **EnumTest.java**, and **CustomerTest.java** JUnit tests. You can also test your code by building a IceCreamShop.java file, which will have a main method, where you will build Customer and IceCream objects and perform the methods from the classes on them.

This and all program files in this course must include a comment block at the beginning (top) of the source code file that contains:

- the Java program name
- project description
- your name
- the version date
- the course number and section

The comment block for the header should look like this:

```
/***************************************************************************
 * Java program name
 ***************************************************************************
 * Project description
*________________________________________________________
 * Your name
 * The version date
 * The course number and section
 ***************************************************************************/
```

In addition, each item created in each class file should be commented using **<u>Javadocs</u>** commenting format as show below:

```
/**
 The variable, constructor, or method is as specified.
 */
```

You will submit your code for your testing program as your algorithm. It should create two Customer objects and populate them with the IceCream object orders above. This code is due when your algorithm is due.

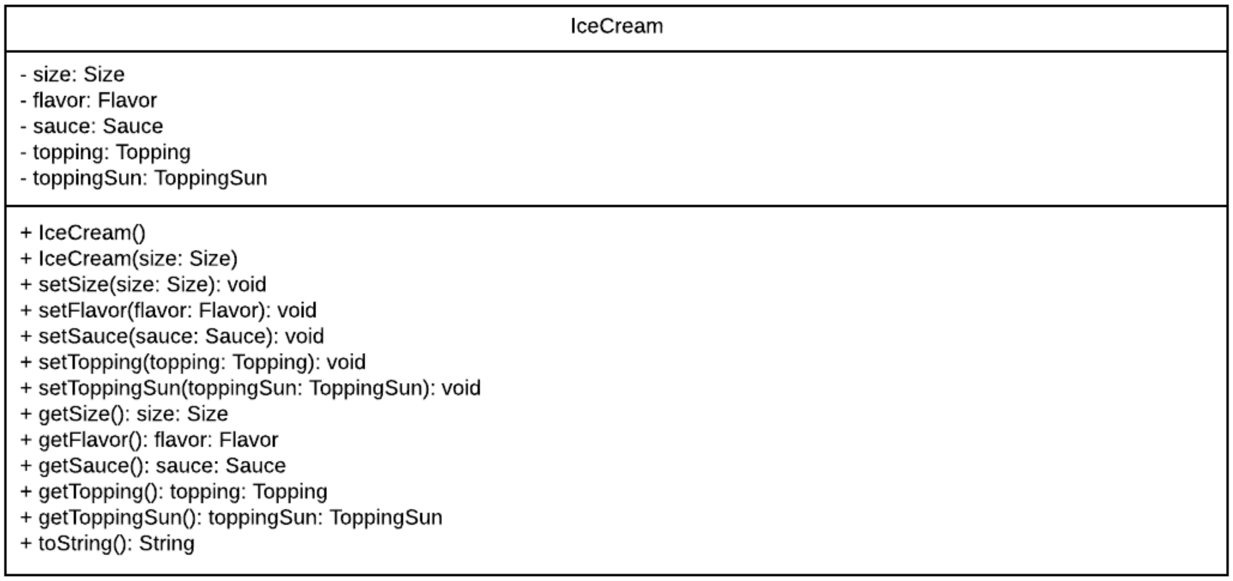
You will submit your Java source code files (IceCream.java, Customer.java, and the six Enum classes) by uploading the files to the Assignment link in Blackboard. Please **<u>do not</u>** submit your files in a zipped folder.

Ask questions about any part of the programming project that is not clear!

Grading Rubric

Criteria	Points
Program files named and submitted as specified	5
All variables, methods and constructors are properly commented using **Javadocs** commenting	10
Enumerations are syntactically correct and used properly	10
IceCream instance data members are correctly declared	5
IceCream default and parameterized constructor correctly written	5
IceCream specified accessor and mutator methods correctly constructed	20
IceCream toString() method written correctly	5
Customer instance data members are correctly declared	5
Customer default and parameterized constructor correctly written	5
Customer specified accessor and mutator methods correctly constructed	20
Customer toString() method written correctly	5
Program layout and appearance (Coding style is clear and easily understood)	5
TOTAL	100

UML Diagrams



```
IceCream
- size: Size
- flavor: Flavor
- sauce: Sauce
- topping: Topping
- toppingSun: ToppingSun
+ IceCream()
+ IceCream(size: Size)
+ setSize(size: Size): void
+ setFlavor(flavor: Flavor): void
+ setSauce(sauce: Sauce): void
+ setTopping(topping: Topping): void
+ setToppingSun(toppingSun: ToppingSun): void
+ getSize(): size: Size
+ getFlavor(): flavor: Flavor
+ getSauce(): sauce: Sauce
+ getTopping(): topping: Topping
+ getToppingSun(): toppingSun: ToppingSun
+ toString(): String
```

Size

SINGLE
DOUBLE
TRIPLE
SUNDAE

Flavor

VANILLA
CHOCOLATE
ROCKY_ROAD
STRAWBERRY
MINT_CHOCOLATE_CHIP
OREO
BUTTER_PECAN

Sauce

NONE
HOT_FUDGE
CARAMEL
STRAWBERRY

Topping

NONE
SPRINKLES
NUTS
CHOCOLATE_FLAKES
GUMMIES
CANDY

ToppingSun

NONE
WHIPPED_CREAM_AND_CHERRY

Customer

```
- lastName: String
- firstName: String
- phone: String
- email: String
- iceCream: ArrayList<IceCream>
```

```
+ Customer()
+ Customer(lastName: String, firstName: String, phone: String, email: String)
+ setLastName(lastName: String): void
+ setFirstName(firstName: String): void
+ setPhone(phone: String): void
+ setEmail(email: String): void
+ getLastName(): lastName: String
+ getFirstName(): firstName: String
+ getPhone(): phone: String
+ getEmail(): email: String
+ orderIceCream(iceCream: IceCream): void
+ getNumIceCream(): numIceCream: int
+ getIceCream(): iceCream: ArrayList<IceCream>
+ toString(): String
```