ngee ann polytechnic

SCHOOL OF INFOCOMM TECHNOLOGY

Diploma in Information Technology
Diploma in Cybersecurity & Digital Forensic
Diploma in Data Science

PROGRAMMING II

Year 2023/24 - Semester 2

ASSIGNMENT

Duration	:	2½ weeks (15 to 31 Jan 2024)	
Weightage	:	30% of total coursework	
Individual/Team	:	Team of 2	
Format	:	Programming - Class Implementation	(10%)
		Basic Features	(50%)
		Advanced Features	(20%)
		Presentation	(20%)

Cut-Off Date/Time: Wednesday, 31 January 2024, 23:59 hours

Penalty for late submission:

- 10% of the marks will be deducted for each day (inclusive of Saturdays, Sundays and public holidays) after the deadline.
- No submission will be accepted after 07 Feb 2024, 23:59 hours.

There is a total of 10 pages (including this page) in this handout.

WARNING

If a student is found to have submitted work not done by him/her, he/she will not be awarded any marks for this assignment. Disciplinary action will also be taken.
Similar action will be taken for the student who allows other student(s) to copy his/her work.

Ice Cream Shop Management System

In this assignment, you are to apply Object Oriented Programming to develop a simple ***Ice Cream Management System***. The assignment requirements described below are broken down into 2 stages of development, described in this document as '***Basic Features***' and '***Advanced Features***'. You are advised to do your programming progressively in these stages. Refer to the '***Grading Criteria***' to have an idea of how the different components are graded.

1. Background

I.C.Treats is Singapore's first robotic ice cream store. This fully automated ice cream parlour offers delicious scoops of ice cream served up three ways: cup, cone, or waffle (all "handmade" by Otto the Friendly Robot Scooper). The standard cup or cone cost the same price, but selecting the waffle option will incur an additional cost of $3. For all three options, you can choose to have 1-3 scoops of any of the flavours seen in Table 2, with the premium flavours costing $2 extra per scoop. In addition, toppings can be added at a cost of $1 per topping added. Finally, customers may choose to upgrade their cone to a chocolate-dipped cone or choose what flavour waffle they want, for an additional cost, of course. Otherwise they can just stick with the originals.

Option	Scoops	Price ($)	Add-ons
Cup	Single	4.00	• Toppings (+$1 each)
	Double	5.50	
	Triple	6.50	
Cone	Single	4.00	• Toppings (+$1 each) • Chocolate-dipped cone (+$2)
	Double	5.50	
	Triple	6.50	
Waffle	Single	7.00	• Toppings (+$1 each) • Red velvet, charcoal, or pandan waffle (+$3)
	Double	8.50	
	Triple	9.50	

Table 1 – Information of the Ice Cream Options

Regular Flavours	Premium Flavours (+$2 per scoop)
Vanilla	Durian
Chocolate	Ube
Strawberry	Sea salt

Table 2 – Ice Cream Flavours

Toppings (+$1 each)
Sprinkles
Mochi
Sago
Oreos

Table 3 – Ice Cream Toppings

Otto the Friendly Robot Scooper relies on this system to take orders and manage the entire store all by itself or else it will lose its job to teenagers. When Otto receives an order from a customer, it gets placed in the queue with all the specifications of the order (option, scoops, flavours, etc.). Note: one order can have multiple ice creams in it (for example, a father orders 3 different ice creams for his whole family all in the same order). Otto prepares the orders in the sequence that they are received, only releasing them from the queue when they have been completed and displayed to the customer. In addition, Otto makes sure to keep track of the birthdays of all its customers, as the first order made by customers on their birthday will have the most expensive ice cream in that order be provided free of charge (i.e. only one ice cream ordered by a customer on his/her birthday will be free).

If they do not already have one, paying customers are automatically given a special PointCard and put into the rewards programme at I.C.Treats the moment they are registered into the system. A customer begins as an ordinary member and is only converted to a silver or gold membership once 50 and 100 points are earned, respectively. Once a customer enters a new tier of membership they cannot drop back down, even if they use up all their points and fall below 50 or 100 points.

The information of some registered customers is as shown in Table 4 below. More data is available in the data file ***customers.csv***.

Name	Membership status	Membership points
Amelia	Gold	150
Bob	Ordinary	5
Cody	Silver	65

Table 4 – Registered Customers

Points are earned from the final amount paid by customers for their orders at the shop. The conversion rate for points to be earned is 72% of the total amount paid rounded down to the nearest integer. For example, if Amelia buys a single scoop of Vanilla ice cream in a cup it will cost her $4.00 and net her Floor(4.00 * 0.72) = 2 points.

Only silver and gold members can redeem their points. Silver members and above can redeem their points to offset the cost of their final bill, where 1 point = $0.02. In addition, gold members have the added benefit that they will be placed into the special gold members order queue whenever they place their order. All the orders in the gold members order queue will always be fulfilled before fulfilling the orders in the regular queue.

In addition, all PointCards also have a punch card system set in place, where for every 10 ice creams a customer orders at I.C.Treats, their 11^{th} ice cream comes free of charge, after which the punch card is reset back to 0.

The class diagram for the Ice Cream Shop Management System is shown in Figure 1 on the next page.

The I.C.Treats Management System©

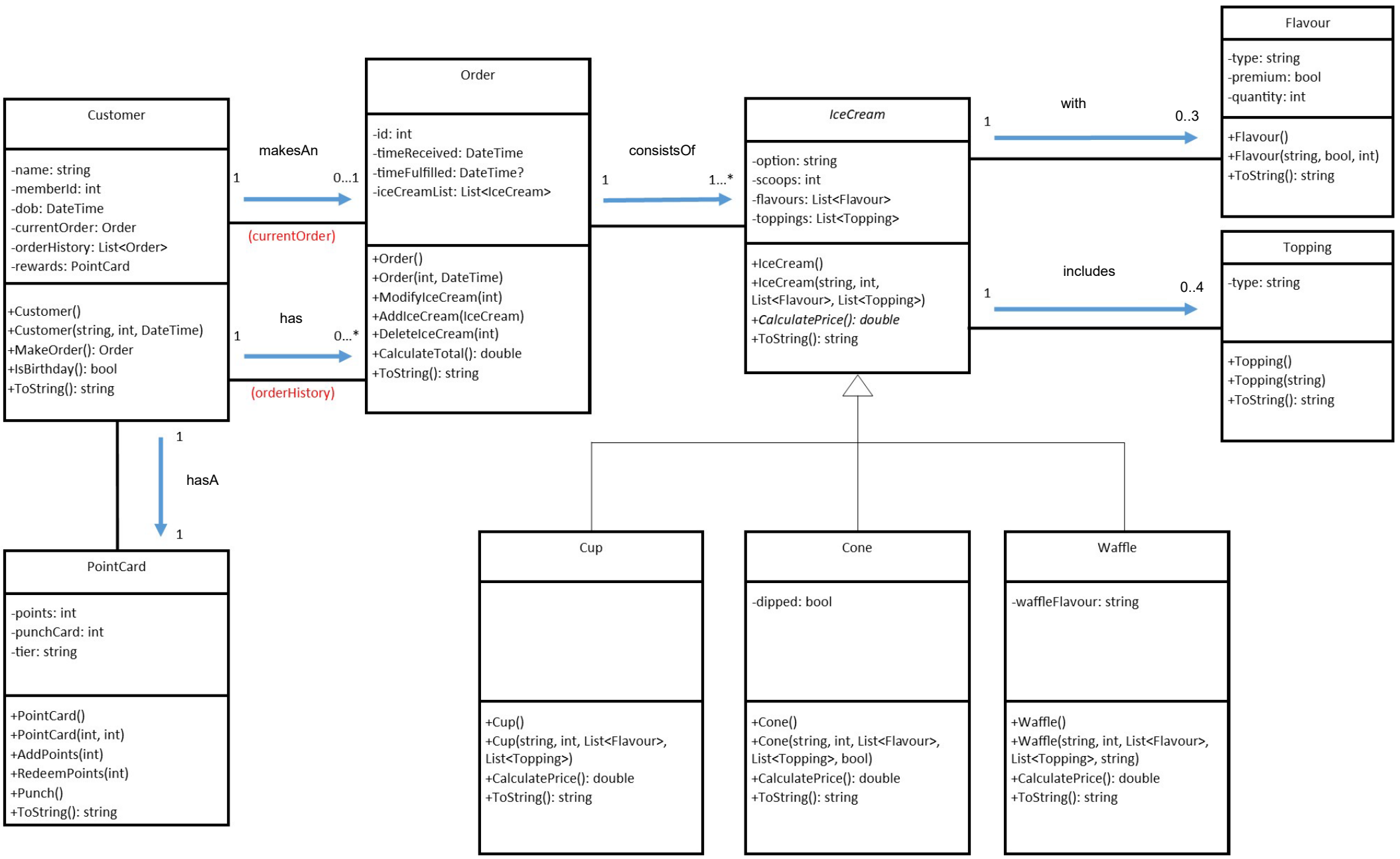

Figure 1: Class Diagram for I.C.Treats Management System

2. CLASS IMPLEMENTATION – 10% GROUP

The team has to divide the work when implementing the classes.

**Note it is highly recommended that teams use Git and GitHub to work together on the project and track their changes.*

3. BASIC FEATURES – 50% INDIVIDUAL

Your program is required to create customers and orders from the given data files at the onset. After which it displays a menu for user to choose to perform each of the feature describe below repeatedly until user chooses to exit from the menu.

1) List all customers

- *display the information of all the customers*

2) List all current orders

- *display the information of all current orders in both the gold members and regular queue*

3) Register a new customer

- *prompt user for the following information for the customer: name, id number, date of birth*
- *create a customer object with the information given*
- *create a Pointcard object*
- *assign Pointcard object to the customer*
- *append the customer information to the **customers.csv** file*
- *display a message to indicate registration status*

4) Create a customer's order

- *list the customers from the **customers.csv***
- *prompt user to select a customer and retrieve the selected customer*
- *create an order object*
- *prompt user to enter their ice cream order (option, scoops, flavours, toppings)*
- *create the proper ice cream object with the information given*
- *add the ice cream object to the order*
- *prompt the user asking if they would like to add another ice cream to the order, repeating the previous three steps if [Y] or continuing to the next step if [N]*
- *link the new order to the customer's current order*
- *if the customer has a gold-tier Pointcard, append their order to the back of the gold members order queue. Otherwise append the order to the back of the regular order queue*
- *display a message to indicate order has been made successfully*

5) Display order details of a customer

- *list the customers*
- *prompt user to select a customer and retrieve the selected customer*
- *retrieve all the order objects of the customer, past and current*
- *for each order, display all the details of the order including datetime received, datetime fulfilled (if applicable) and all ice cream details associated with the order*

6) Modify order details

- *list the customers*
- *prompt user to select a customer and retrieve the selected customer's current order*
- *list all the ice cream objects contained in the order*
- *prompt the user to either [1] choose an existing ice cream object to modify, [2] add an entirely new ice cream object to the order, or [3] choose an existing ice cream object to delete from the order*
  - *if [1] is selected, have the user select which ice cream to modify then prompt the user for the new information for the modifications they wish to make to the ice cream selected: option, scoops, flavours, toppings, dipped cone (if applicable), waffle flavour (if applicable) and update the ice cream object's info accordingly*
  - *if [2] is selected prompt the user for all the required info to create a new ice cream object and add it to the order*
  - *if [3] is selected, have the user select which ice cream to delete then remove that ice cream object from the order. But if this is the only ice cream in the order, then simply display a message saying they cannot have zero ice creams in an order*
- *display the new updated order*

- **Validations** (and feedback)
  - *The program should handle all invalid entries by the user*
    e.g. invalid option, invalid number of scoops, invalid flavour, etc.
  - *If user made a mistake in the entry, the program should inform the user via appropriate feedback*

IMPORTANT INSTRUCTIONS:

- *One student is to implement features 1, 3 & 4, and another for features 2, 5 & 6.*
- *Individual student without a team is required to implement features 1, 2, 4 & 6.*

4. ADVANCED FEATURES – 20% INDIVIDUAL

(a) Process an order and checkout

- *dequeue the first order in the queue*
- *display all the ice creams in the order*
- *display the total bill amount*
- *display the membership status & points of the customer*
- *check if it is the customer's birthday, and if it is, calculate the final bill while having the most expensive ice cream in the order cost $0.00*
- *check if the customer has completed their punch card. If so, then calculate the final bill while having the first ice cream in their order cost $0.00 and reset their punch card back to 0*
- *check Pointcard status to determine if the customer can redeem points. If they cannot, skip to displaying the final bill amount*

- *if the customer is silver tier or above, prompt user asking how many of their points they want to use to offset their final bill*
- *redeem points, if necessary*
- *display the final total bill amount*
- *prompt user to press any key to make payment*
- *increment the punch card for every ice cream in the order (if it goes above 10 just set it back down to 10)*
- *earn points*
- *while earning points, upgrade the member status accordingly*
- *mark the order as fulfilled with the current datetime*
- *add this fulfilled order object to the customer's order history*

(b) Display monthly charged amounts breakdown & total charged amounts for the year

- *prompt the user for the year*
- *retrieve all order objects that were successfully fulfilled within the inputted year*
- *compute and display the monthly charged amounts breakdown & the total charged amounts for the input year*
- *Sample mock up screenshot:*

```
Enter the year: 2023

Jan 2022:    $32.00
Feb 2022:    $64.00
.
.
.
Nov 2022:    $128.00
Dec 2022:    $256.00

Total:       $4534.63
```

(c) Recommend a feature to be implemented (bonus marks are only awarded if advanced feature is completed)

- *you may gain up to 5 bonus marks if you propose and successfully implement an additional feature.* ***You are REQUIRED to check with your tutor about your idea before implementing.***

IMPORTANT INSTRUCTIONS:

- *A student is to implement feature (a) and another implements feature (b).*
- *Individual student without a team can choose to implement either one of the features.*
- *Please note that you should implement the advanced features only AFTER all the basic features have been fully implemented and working.*
- *NO MARKS will be awarded for the advanced features if the basic features have NOT been fully implemented and working.*
- *Marks will be deducted if you are not able to show your understanding of the program, both basic and advanced features (if applicable), during the presentation.*

5. ACTIVITY PLAN

Suggestions for Getting Started

There are many ways that you could complete this assignment. It is most important part to think about the entire assignment first so that it is easy to integrate the various parts.

a) Analysis

1. Understand the program specification and the requirements before attempting the assignment.
   e.g. the relationships between the classes
   the use of the attributes in each class

b) Program Design

2. Work out the User Interface needed to get the user input for suitable output.
3. Work out the main logic of the program using Object-Oriented programming techniques;

   i.e. use the inheritance and the association of the classes properly.
4. You are required to use suitable classes and methods appropriately for this assignment.

   Marks will be deducted for inefficient use of the classes/methods or improper use of classes/methods

c) Implementation & Testing

5. Determine the order in which the classes are to be implemented. (Certain classes need to be implemented before other classes can be implemented)
6. Implement the classes **ONE** at a time.
7. Test your program logic to make sure that it works.

   You must prepare test data to see that your program works correctly. All data entry should be validated and illegal data entry should be highlighted to the user so that the user can enter correct data.

6. DELIVERABLES

You are to develop the project using Console App. You should name your project as **`Snnnnnnnn_PRG2Assignment`** *(note: **`Snnnnnnnn`** is your Student Number)*.

You are required to submit your work in **TWO** stages:

Stage 1 – Due date: Wednesday, Week 15 (24 January 2024) @ 23:59 hours

- ALL the 9 classes (source files) shown in the class diagram and at least 1 basic feature to your **Brightspace Stage 1 Submission**.

  In **EACH** of your source files, you **MUST** include a block comment at the top stating your **`student number, student name,`** and **`partner name`** as shown below:

```
//=======================================================
// Student Number : S12345678
// Student Name   : Michael Jordan
// Partner Name   : Scottie Pippen
//=======================================================
```

  Note: If a student did not submit his/her stage 1 work by the deadline, 20 marks will be deducted.

Stage 2 – Due date: Wednesday, Week 16 (31 January 2024) @ 23:59 hours

- ALL the classes (source files) that you have written for the whole assignment to your **Brightspace Stage 2 Submission**.

  In **EACH** of your source files, you **MUST** include a block comment at the top stating your **`student number, student name,`** and **`partner name`** as shown below:

```
//=======================================================
// Student Number : S12345678
// Student Name   : Michael Jordan
// Partner Name   : Scottie Pippen
//=======================================================
```

Presentation – During PRG2 lessons

- You are to present your application to your tutor during the PRG2 classes right after the submission deadline.

7. GRADING CRITERIA

This assignment constitutes 30% of this module.

Performance Criteria for grading the assignment is as described below. Marks awarded will be based on **program code** as well as student's degree of understanding of work done as assessed during the **presentation**.

Grading criteria for the program is given below.

A Grade

- Program implements the *Basic Features* successfully and efficiently
- Program implements all the basic *input validations* successfully
- Program implements *Advanced Features* successfully and efficiently
- Program demonstrates good design with the correct use of methods
- Program provides strong evidence of good programming practice
- Program has been tested adequately

B Grade

- Program implements the *Basic Features* successfully
- Program implements some basic *input validations* successfully
- Program attempts to use methods
- Program provides sufficient evidence of good programming practice
- Program has been tested adequately

C Grade

- Program implements the *Basic Features* successfully
- Program provides some evidence of good programming practice
- Program has been tested adequately

D Grade

- Program implements the *Basic Features* successfully
- Program has been tested adequately
- Presentation is awarded at least a 'D' grade

NOTE

- *Evidence of good programming practice include appropriate use of methods, the use of meaningful variable names, proper indentation of code, appropriate and useful comments, adoption of standard naming conventions etc.*
- *Basic Input validation refers to the checking of the inputs entered by the user. e.g. invalid option, invalid date*