	Total Marks	Remarks	Partial Marks
Correct Implementation	4	Should show correct output and never crash for all given test cases. No need to check error handling. **Little bit of variations in output is fine as long as its not incorrect.**	3
		Special condition -> To check for insufficient balance of customer and remove food items from cart (in any manner)	1
Identifying Interface	2	An Interface should exist (User) and at least one identified common method	1+1
Identifying Classes/With Inheritence	2	Restaurant classes and special Restaurant classes	1
		Customer classes and special Customer classes	1
Identifying Methods	3	Students must have identified bare minumum and important methods in each of the above 4 classes (below method names are a guidance, look if respective tasks are categorized in each class)	
		Restaurant's base class - addFoodItems, editFoodItems, printRewards, discountBillValue	1
		Customer's base class - printRewards, printRecentOrders, discountBillValue	1
		Main Class :- All queries (inputs) must only be handled here	1
Polymorphism	2	Use of method names from parent class (with same parameter types) with different implementation in subclasses	
		Restaurant's subclasses - calculateRewards, discountBillValue (indicative, atleast 1)	1
		Customer's subclasses - discountBillValue	1
Encapsulation/correct use of modifiers	3	Use of private and protected member variables appropriately (there should be shared variables across base and subclass)	1
		Public getter/setter methods must be used to access/modify member variable of a class object.	2
	2	Use of override annotation for inherited methods	2
Class Relationships	2	Making use of super() (parametrized so as to make use of parent class' constructor) in constructor in child class	1
		Declaring objects for both base class and subclasses (Instantiation)	1
			20