Advanced Programming Assignment-2 (RUBRIC, Follow Binary Marking for all Parameters)		
Criteria	**Description**	**Marks**
Correct Implementation	Code should give correct output for given test case. Evaluator can check for 1-2 small modifications as well. Binary marking to be followed. No marks to be awarded for partially correct outputs. The output format may vary a little but it should be complete. Note that small assumptions regarding discounts and coupons by students are permissible as long as they can explain it	2
Functionality	code should have sign-in and sign-up feature for customers and sign-in for admins using correct password, note that no sign in should happen with an incorrect password	1
	customers should be able to browse and see products without logging in but should not be able to buy them	1
	admin should be able to add / delete new / existing products and categories	0.5+0.5
	customer should not be able to checkout if the balance in their wallet is insufficient	1
	customer should not be able to buy a product / deal if it is out of stock	1
Inheritance	The code should exhibit inheritance. There can be many places where this can be done, like, Customer can be parent class and the types of Customer (Elite, Regular, Prime) can be child classes which inherit from Customer. If the student has any form of IS-A relationship which they can justify, give marks. Follow binary marking - if there is no IS-A relation, give zero marks irrespective of whether test cases give correct output or not.	3
Encapsulation / correct use of modifiers	no data member should be public; ~~**classes should be public**~~	2
	getters and setters should be used	2
Interface	The code should have at least one interface, for example, there can be many possibilities here such as an interface to implement billing/checkout of orders and each customer type implements it differently. Aside this accept any interface as long as it serves as a contract for the classes it is implemented by and students can provide a justification for the same. Follow binary marking.	3
Polymorphism	The code should exhibit polymorphism in the case of customer types - an object of type Elite / Prime / Regular can be referred to using a variable of type Customer but on calling a function, the implementation specific to the actual class (Elite / Prime / Regular) should run, for example - billing/checkout is different for all three, discount, delivery period etc. If students have shown polymorphism in any other manner, give marks as long as they justify it and show a working example of it. Follow binary marking.	3
	Total	**20**