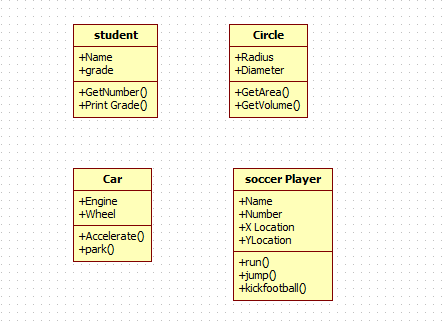
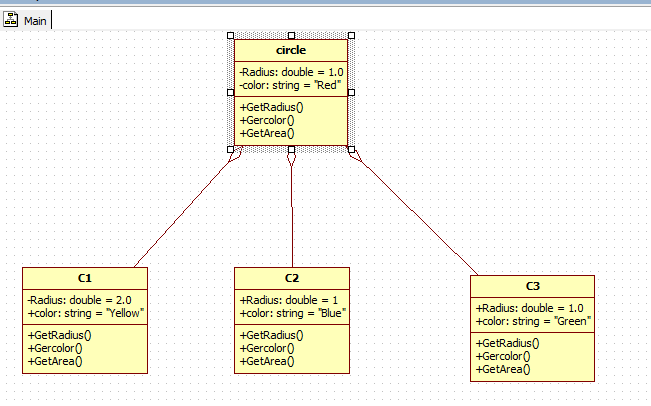
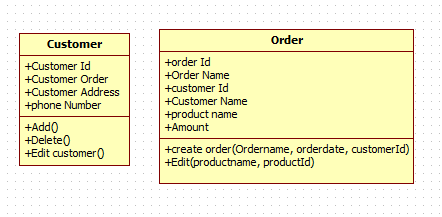
Question 1:

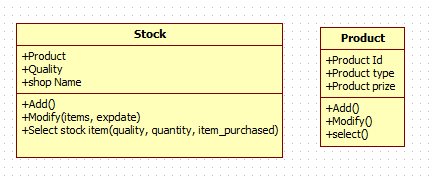


Question 2:

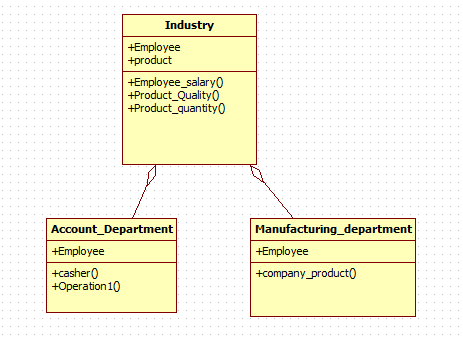


Question 3:





Question 6:



Question 7:

Class :

An object is any person, place, thing, concept, event, screen, or report applicable to your system.

Objects both know things (they have attributes) and they do things (they have methods).

A class is a representation of an object and, in many ways, it is simply a template from which objects are created. Classes form the main building blocks of an object-oriented application.

# Example

Although thousands of students attend the university, you would only model one class,

called Student, which would represent the entire collection of students.

Object :

An object in oops, is an abstract data type created by a developer. It **can** include multiple properties and methods and may even contain other **objects**. In most **programming** languages, **objects** are defined as classes. **object** provide a structured approach **to programming**

Assosiation and its types:

Association

An association is a "using" relationship between two or more objects in which the objects have their own life time and there is no owner.

Example: A patient may visit one or many doctors and same way, a doctor can be associated with multiple patients. If a patient dies, existence of doctor will not be vanished and similarly if doctor dies patient will remain patient.

Association is represented as thin line connecting two classes. Association can be unidirectional (shown by arrow at one end) or bidirectional (shown by arrow at both end) or without arrow.

Aggregation

Aggregation is a special form of association. It is also a relationship between two classes like association,

however, it’s a **directional** association, which means it is strictly a **one way association, meansunidirectional association.** It represents a **Has-A** relationship.

Example: Consider two classes Student class and Address class. Each student must have an address so the relationship between student and address is a Has-A relationship. But if you consider its vice versa then it would not make sense as an Address doesn’t need to have a Student necessarily.

Composition

Composition is a special case of aggregation. In a more specific manner, a restricted aggregation is called composition. When an object contains the other object, if the contained object cannot exist without the existence of container object, then it is called composition.

For **Example**: Consider the same scenario with some modifications. In this scenario, a student has address and each student has different address (Please keep sibling relationship argument apart). So, when a student record is added his house number and street number will be entered. And if I delete the record of a particular student, then his/her record will be of no use.