Question 01

1) Define the term “Object Oriented Programming” (2 marks)

- Object Oriented programming is a method of implementation in which programs are organized as cooperative collections of objects, each of which represents an instance of some class, and whose classes are all members of a hierarchy of classes united via inheritance relationships.

2) What are the main features of Object Oriented Programming? (2 marks)

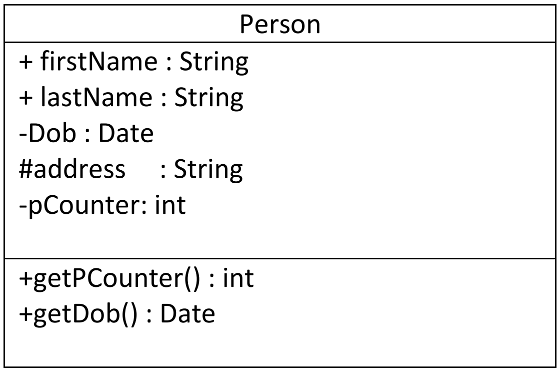
Data Abstraction
Encapsulation
Information Hiding
Inheritance
Polymorphism

3) Explain the relationship between object and a class using a suitable example. (3 marks)

- A class is the abstract definition of the data type. It includes the data elements that are part of the data type, and the operations which are defined on the data type.
Where as an object is an instance of a class.

House house1; Here “House” is the structure which defines the House whereas “house1” is the object which can be used to store data and do operations for a House.

4) Convert the following class represented in UML notation into a proper coding. (4 marks)



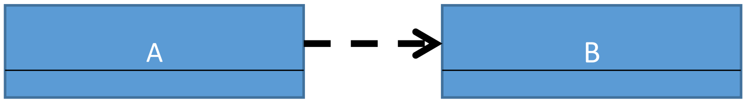
Answer :

```
class Person
{
        private:
                Date Dob;
                int pCounter;
        protected :
                String address[SIZE];
        public :
                String firstName;
                String lastName;
                int getPCounter();
                Date getDob();
};
```

5) What are the relationships that can exist between classes? Show the UML notation for each relationship. (5 marks)

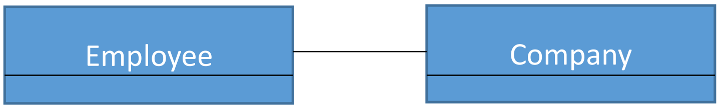
Dependency

- one class depends on another because it uses it at some point in time.



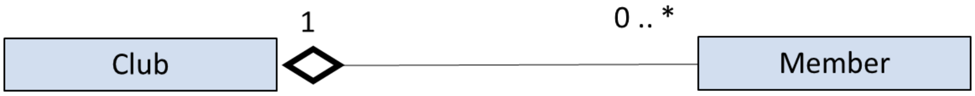
Association

- An association between two classes indicates that objects at one end of an association “recognize” objects at the other end and may send messages to them.



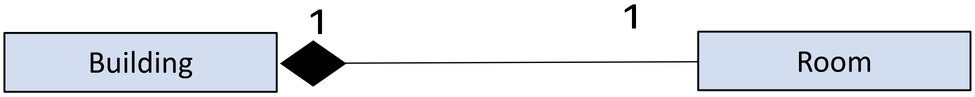
Aggregation

- Aggregation is just a special kind of association
- Aggregation is a weak form of whole-part relationship
- This means that A aggregates B.
- This implies that the whole can exist without the parts.



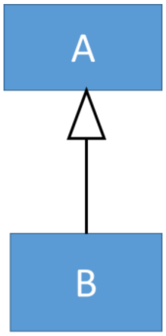
Composition

- Composition is a strong form of whole-part relationship



Inheritance

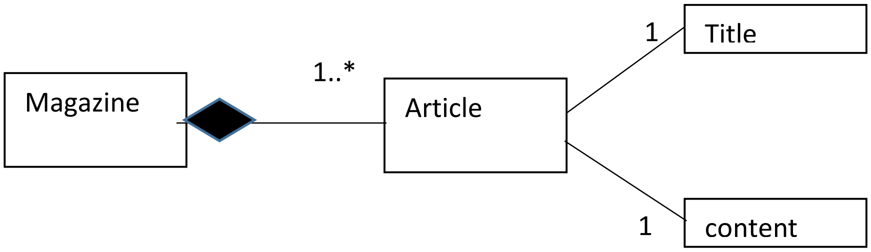
- Otherwise Known as Generalization.
- Inheritance represents a "is-a-kind-of" relationship.
- Inheritance is a relationship between a general thing (superclass/parent) and a more specific kind of a thing (subclass/child).



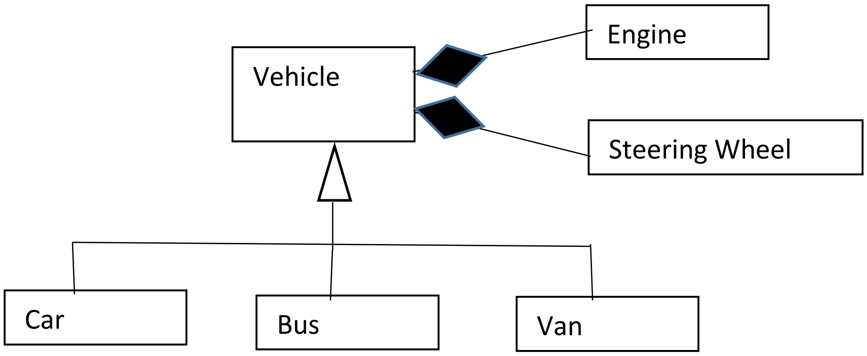
6) Identify the classes and appropriate relationships for the following descriptions. Draw separate class diagrams for each description using UML Notations. (9 marks)

   a. "A magazine consist of several articles. Each article should have a separate title and content"
   b. "Cars, Buses and Vans are vehicles. Any vehicle must have an engine and steering wheel. "
   c. "There are five group members in an assignment group. A group member can be assigned exactly to one assignment group. One of the group members may also work as the group leader. An assignment group is led by only one group leader."

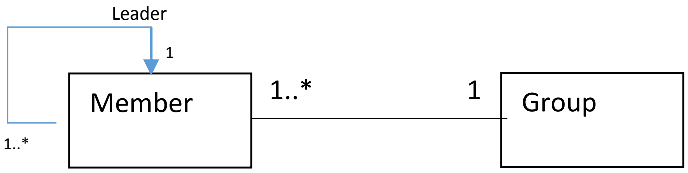
a)


Magazine
1..*
Article
1
Title
1
content

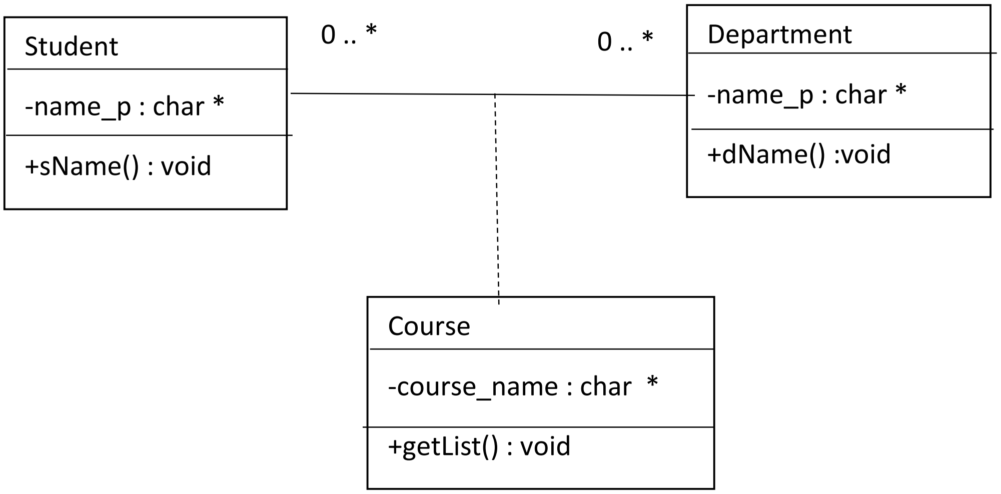
b)


Vehicle
Engine
Steering Wheel
Car
Bus
Van

c)


Leader
1
1..*
Member
1..*
1
Group

7) Consider the following diagram. Write the coding for the three classes. (10 marks)



```
class Student
{
   private:
      char *name_p;

   public :
      void sName();
};


class Department
{
   private:
      char *name_p;

   public :
      void dName();
};


class Course
{
   private :
     Student *std_p;
     Department *dept_p;
     char *course_namel

   public:
     void getList();
};
```

Question 02 (20 marks)

Draw at least 5 CRC cards for the scenario of an Online Super Market given below by identifying the possible classes, their responsibilities and collaborations.

ABC Supermarket has provided customers the ability to purchase goods online. A customer can purchase goods online by providing the delivery address, adding the items to the shopping cart and making the payment. The payment can be made by a Credit Card by providing the credit card number, name of the card holder, the expiry date, the CVC number and the amount to be paid.
A Credit card service provider is used for credit card payments. Initially the credit card details are validated by the credit card service provider. The credit card service provider checks if there is sufficient credit (money) to complete the transaction. If there is insufficient credit an error message is displayed. In addition customers can make payments when goods are delivered.
The manager of the supermarket can access the monthly sales report which is generated by the system. A monthly accounting summary is exported (sent) to the Accounting system used by the supermarket.

Customer	
Register	

Payment	
Store payment details	Customer

Credit Card Service Provider	
Verify credit card details Check availability of credit Display message	Payment Customer

Items	
Update details	

Report	
Monthly Accounting Summary	Payment
Monthly Sales Report	Shopping Cart

Shopping Cart	
Add Items	Customer
Calculate payment	Items

Question 03 (25 marks)

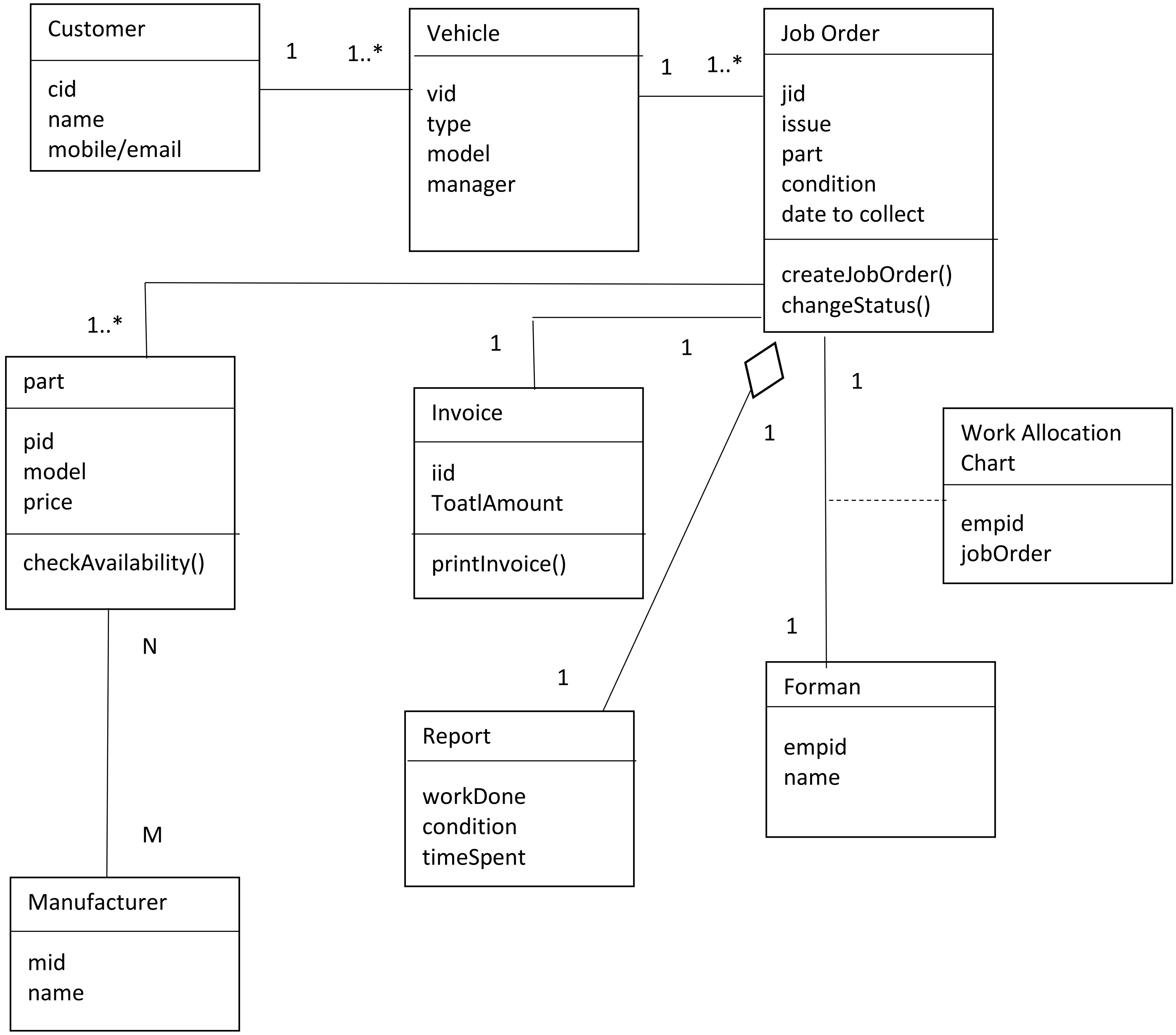
Read the following description and identify the classes, attributes, methods and relationships between classes, and multiplicity specifications. Draw a UML class diagram for the following scenario using the above features.

“Sunshine Service” is a repair station for automobile vehicles such as cars, vans, jeeps, etc… They undertake all types of vehicle repairs at their repair station at Colombo.
Any customer can bring his/her vehicle for repairs to “Sunshine Service”. Once a new repair arrives to the repair station, an Automobile Specialist will be allocated for the job. He will then create a new job order for the corresponding vehicle and will record relevant details such as, the issue, part(s) where the problem exists (eg: engine gearbox, power steering pump, etc…), the condition of the vehicle and the date of hand over requested by the customer. Then the original copy of the job order will be given to the customer and the vehicle will be sent to the Repair Station Coordinator with a copy of the job order.

The Repair Station Coordinator inspects the vehicle to decide whether the requested repair can be done using available spare parts in repair station. If it cannot be done using available spare parts in the repair station, the customer will be notified regarding the situation, and the satus of the vehicle will be mentioned as “ON-HOLD” until repair station receives the required spare parts from the manufacturer/ spare parts dealer. If the repair can be done using available spare parts, the Repair Station Coordinator will create a Work Allocation Chart and the work will be assigned to a suitable Forman.

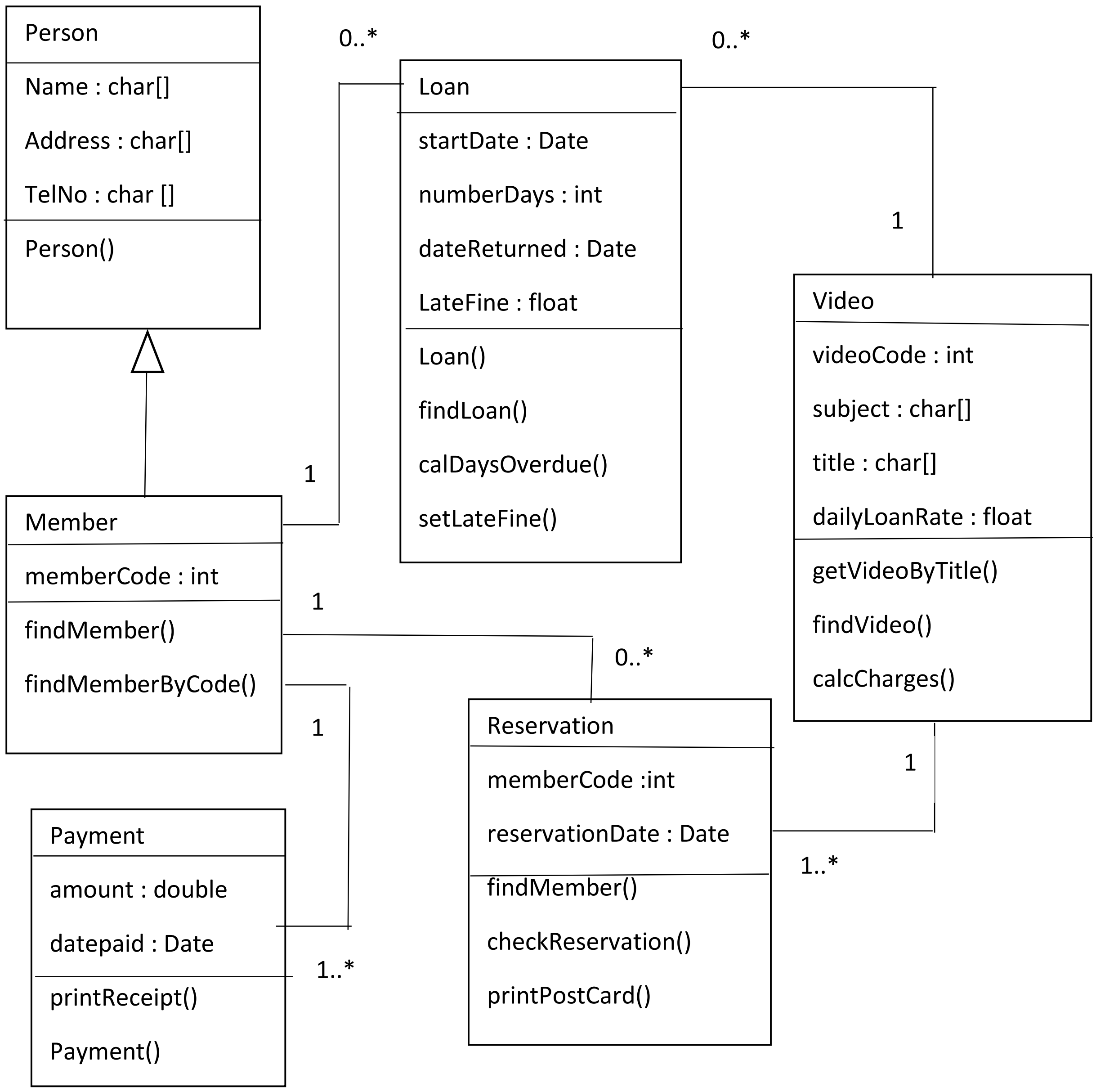
During the process of fixed the problem of the vehicle, if the Forman discovers new problem in the vehicle he will inform it to the Repair Station Coordinator. Then the Repair Station Coordinator will create a new Work Allocation Chart for the newly identified problem and will assign the Forman to carry out the work on it.

When the repairs are completed, the Forman has to submit a detailed Report mentioning the work he did, new spare parts used (if any), the new condition of the vehicle, time spent etc.. This Detailed report will be used by the Automobile Specialist to create the Invoice.



Question 04 (20 marks)

Consider the following scenario of a Video Rental Shop and write the C++ coding to represent the classes. Include the attributes and methods and also show the relationships between classes through the coding.



```
#define SIZE 10
// Assumption SIZE captures the max number of relationships
//  that can be there for simplicity
class Person
{
   protected:
      char name[20];
      char Address[40];
      char TelNo[10];

  public :
      Person();
};

class Member : public Person
{
    private :
      int memberCode;
      Payment *payArr[SIZE];
      Loan *loan[SIZE];
      Reservation *res[SIZE];
    public:
      Member();
      void findMember();
      void findMemberByCode();
};
```

```
class Payment
{
    private:
      double amount;
      Date datepaid;
      Member *mem;
    public:
      Payment()
      void printReceipt();
};


class Loan
{
   private:
       Date startDate;
       int numberDays;
       Date dateReturned;
       float Latefine;

       Member *mem2;
       Video *video;
  public:
       Loan();
       void findLoan();
       void calDaysOverdue();
       void setLateFine();
};


class Video
{
   private:
        int videoCode;
        char Subject[20];
        char title[30];
        float dailyLOanRate;
        Loan *loan[SIZE];
        Reservation *res[SIZE];

   public:
       Video();
       void getVideoByTitle();
       void findVideo();
       float calCharges();
};


class Reservation
{
   private:
      int memberCode;
      Date reservationDate;
      Member *mem;
      Video *vd;
   public:
      Reservation();
      void findMember();
      void checkReservation();
      void printPostCard();
};
```