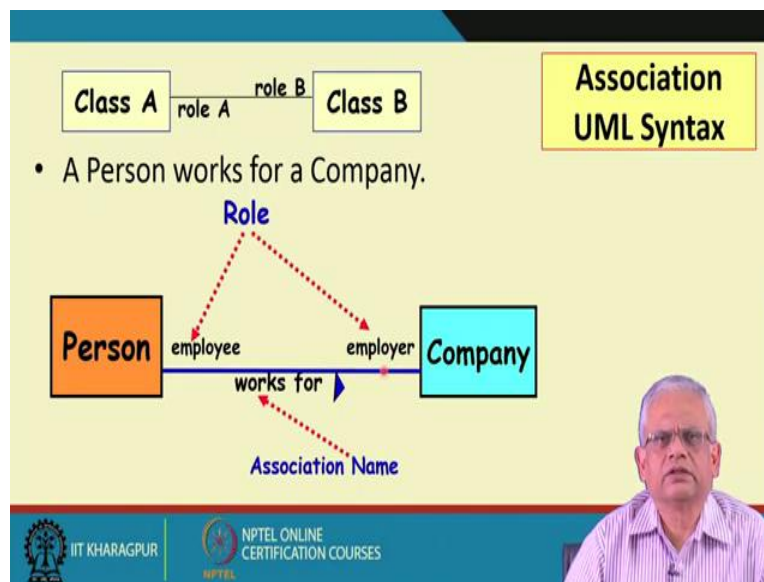


Object Oriented System Development using UML, Java and Patterns
Professor Rajib Mall
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Binary and Unary Associations
Lecture 09

Welcome to this session! Over the last few sessions, we had discussed about the UML use case modeling and after that we had taken up the class diagrams. We had looked at the basic syntax of class diagrams. Then, we were looking at the class relations, how to represent and implement them in Java. We first started with the inheritance relation and we found that, it is very easy to implement using the extends keyword of the Java and then we were looking at association relation, the basic syntax for the association relation. Today, we will continue with this syntax and also will look at how to have the Java implementation for the association.

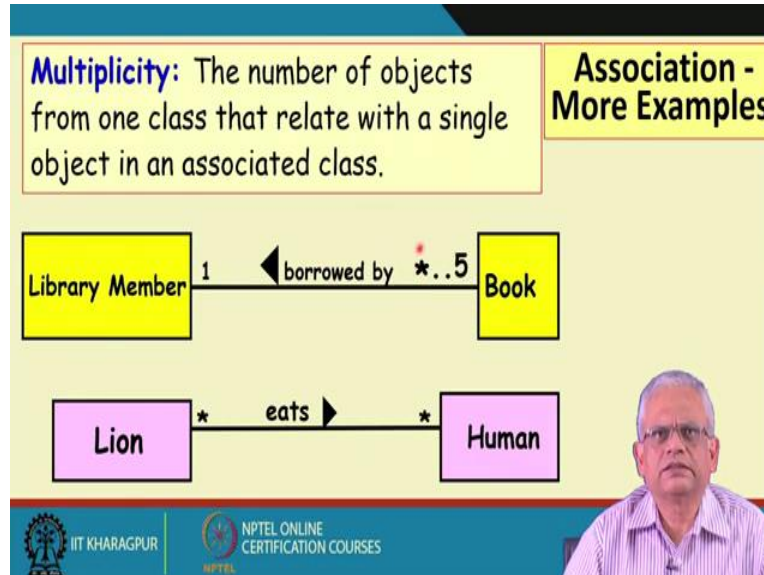
(Refer Slide Time: 01:18)



Let us continue from this point. We have a class A and we have a class B which are associated, we write the name of the association on a straight line. The straight line in the above first diagram joining Class A and Class B represents the association relation between these two classes. We write the name of the association and then we can optionally write the role of each class in the association relation. And then we were taking an example of person working for a company and here we represent the two classes, Person and Company. We write the name of the association relation is the association name works for, we give the reading direction and we write

the name of the role in the association, in the works for association. The role of the person is an employee and the role of the company is the employer that is the basic syntax we are discussing.

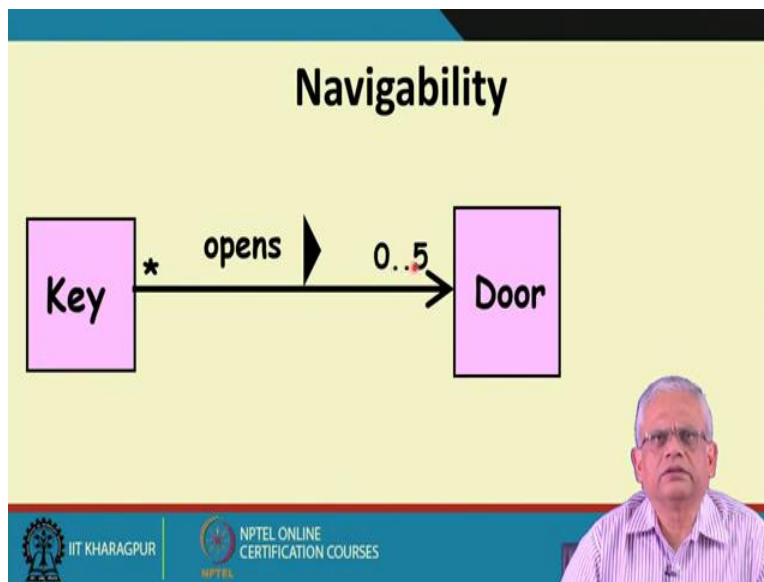
(Refer Slide Time: 02:44)



Now let us look at these examples, we have a Library Member and we have a Book. These two classes are associated that is shown by the straight line between the two classes and we write the name of the association that is borrowed by. We have here the multiplicities or the cardinalities which basically says how many of one object of book class is associated with how many objects at the library member class. We read by saying, a book is book is borrowed by a library member that is 1. A library member borrows up to 5 books. Actually the * in the above diagram should be 0 here, A library member borrows 0 to 5 books. So one object of the library member class can be associated with 5 book objects.

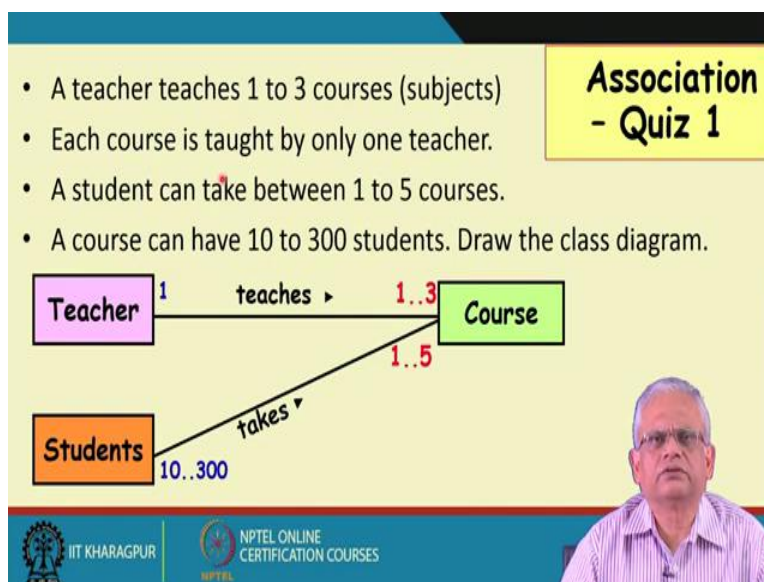
Similarly, we have another example here, we read this by saying a lion eats many humans. We can read it in other ways because the association here bidirectional, simple straight line here means bidirectional association we can read here, a human is eaten by many lions. So one element of the human, one object of the human is associated with many lion objects. Similarly, a lion object is associated with the many human objects is represented by the star. Just to summarize this we say that the number of objects from a one class that related to the objects of the other class is represented as the multiplicity. A single object of one class is related to how many objects of the other class is represented as the multiplicity.

(Refer Slide Time: 05:36)



Now let us try to read this, in the above diagram, it is a unidirectional association represented by the arrow relation. A straight line without arrows is a bidirectional association. Here, the direction of the association is unidirectional and we read here by saying a key opens up to 5 doors. A single key opens up to 5 doors and a door is opened by many keys. We will see the unidirectional association, how it is implemented in Java and how is it different from bidirectional association little later.

(Refer Slide Time: 06:36)



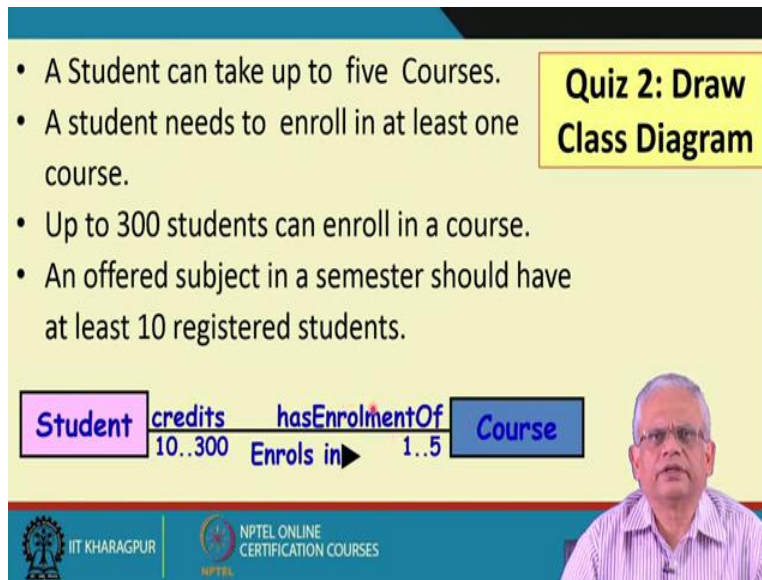
Now let us have a small quiz. Let us try to identify the classes, association relation and represent that in a UML class syntax. In any design activity identifying classes from a problem description and the relations among classes is a very vital skill. Now let us do this small exercise. A teacher teaches 1 to 3 courses or subjects whatever we may tell either we can say courses or subjects. A teacher teaches anywhere between 1 to 3 courses and each course is taught by only a single teacher. A student can take between 1 to 5 courses and a course can have anywhere between 10 to 300 students. Now we need to draw the class diagram. We need to identify the classes and then we need to identify the association relation followed by representation in the UML syntax.

We need to identify the multiplicities those are given here and represent that in the UML syntax. Now what are the classes here? That is the first question. The classes are of course the teacher, the course and the student. Course and subject are synonyms, so we have 3 classes here. The teacher, course and student: these appear as nouns in a description. Later as we proceed with the course, we will pick up more higher skills in identifying the classes from a problem description. Now after identifying the classes, we need to identify the association relation and represent them. And here teacher is associated with courses because he teaches, that is the name of the association relation he teaches.

And the cardinality given here is 1 to 3 courses, a single teacher is associated with 1 to 3 courses. So this is the cardinality on the course side and a course is taught by only one teacher. So this is the cardinality on the teacher side and student is associated with a course because credits the course, takes the course. And the cardinality is given here this is on the course end is 1 to 5 courses. A single student can take 1 to 5 courses. And a course can have 10 to 300 students, so this is the cardinality on the student side.

Now let us represent that in the UML diagram you can try representing yourself and compare your answer with the one that we provide here. So teacher teaches is the reading direction, teacher teaches 1 to 3 courses. A course is taught by a single teacher and a student takes 1 to 5 courses and a course is taken by 10 to 300 students. I hope that given a problem description you can identify the classes, the association relations between the classes and also identify the cardinalities and represent them correctly in a UML diagram.

(Refer Slide Time: 11:05)



The slide features a yellow background with a blue header and footer. A yellow box in the top right corner contains the text "Quiz 2: Draw Class Diagram". A list of four bullet points is on the left. In the center, a UML class diagram shows a "Student" class (purple box) connected to a "Course" class (blue box) by a solid line. The "Student" side has the role "credits" and cardinality "10..300". The "Course" side has the role "hasEnrolmentOf" and cardinality "1..5". An arrow labeled "Enrols in" points from the "Student" side to the "Course" side. A small video inset of a man is in the bottom right. Logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES are in the footer.

- A Student can take up to five Courses.
- A student needs to enroll in at least one course.
- Up to 300 students can enroll in a course.
- An offered subject in a semester should have at least 10 registered students.

Quiz 2: Draw Class Diagram

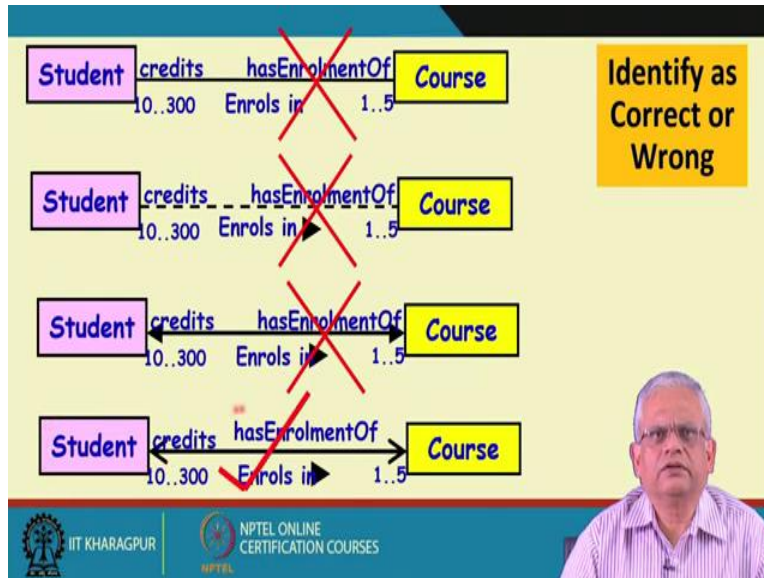
Student credits 10..300 Enrols in hasEnrolmentOf 1..5 Course

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

Now let us draw another class diagram based on a problem description. This is quiz 2 as described in the above diagram. Let us say we just changed the earlier problem little bit like we say that a student can take up to 5 courses that means 0 to 5 courses. Students needs to enroll in at least 1 course that means that a student can take 1 to 5 courses because he needs to enroll in at least one course. Up to 300 students can enroll in a course and offered subject should have at least 10 registered students. So that means that a course should have between 10 to 300 students.

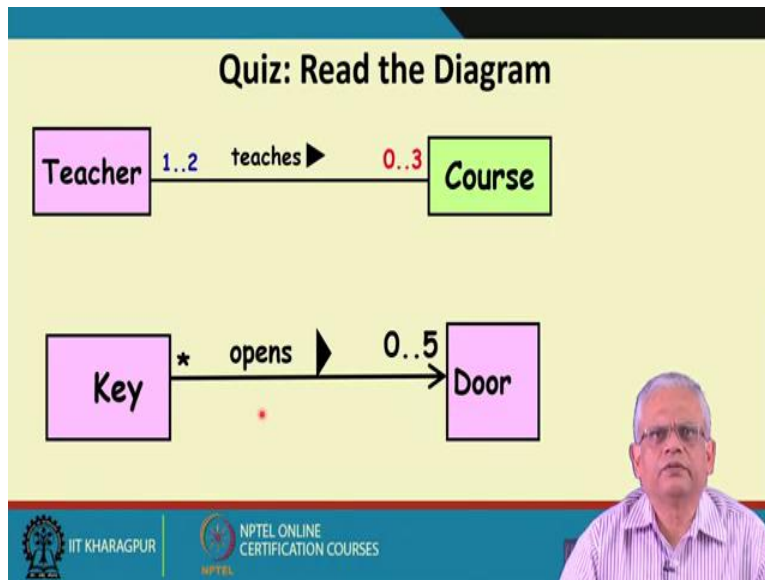
Now the classes that are here as you can see in the above diagram are the student and the course. And we need to represent this properly in a UML class diagram with proper cardinalities. So here we have also written the role of each class. A student enrolls in 1 to 5 courses; a course has enrolment of 10 to 300 students. So here the role of the student is the student credits the course and course has the role as has enrolment of.

(Refer Slide Time: 13:09)



Now let us do another small exercise. Let us try to see the UML diagram for the same problem drawn by different learners and see if what they have drawn is correct or not. Now, just observe the first diagram in the above figure, student enrolls in course, student and course are the 2 classes and we have the association name cardinality and the roles. Are there any mistakes that you can plot, you can identify or is it okay? It is a mistake because we do not have the reading direction. In UML syntax along with a name of the association relation we should have given the reading direction. Now what about the second problem in this diagram? Is this diagram at the problem drawn by a learner okay? Here, still there is a problem, the problem is that the association relationship is not represented by a dotted line, it is represented by a full line, normal line with arrow heads, so this is again problem. Now what about the third diagram? The bidirectional association can also be drawn by arrows on both ends but is this okay? We have the association name, have the reading direction, cardinality properly specified roles but still there is a problem, the arrow type is not proper, this is a filled arrowhead whereas UML syntax wants an open arrowhead. Now what about the fourth one? We have the open arrowhead here, we have the association name, we have the cardinality, we have the normal line here, we have the role of each class in the association. So this looks alright, so this is correct notation in UML.

(Refer Slide Time: 16:13)



Now let us have one more quiz. We want to read this diagram wherein association relationship is mentioned, how do we read this? We can read it if we know that the cardinality here is for a single object of the teacher class. So we can read here like a teacher teaches up to 3 courses and a course is taught by 1 or 2 teachers. Let me repeat that, the cardinality mentioned at this end is the number of objects of this class that can be associated with a single object of this class. So we need to read here as a teacher teaches up to 3 courses and if we read in the other direction we read here like a single object of the course that is a course is taught by 1 or 2 teachers. Now let us read the second one. A key opens up to 5 doors and a door is opened by many keys.

(Refer Slide Time: 17:41)

- **A link:**
 - An instance of an association
 - Exists between two or more objects
 - **Dynamically created and destroyed as the run of a system proceeds**
- For example:
 - An employee joins an organization.
 - Leaves that organization and joins a new organization.

The slide features a yellow background with a blue header and footer. The title 'Association and Link' is in a yellow box. The class diagram shows 'People' and 'Tax_files' classes with an association labeled 'tax_file'. The 'People' class has three objects: Rakesh Shukla, V. Ramesh, and Keshab Parthi. The 'Tax_files' class has two objects: 760901-1234 and 691205-5678. Red lines connect Rakesh Shukla to 760901-1234 and V. Ramesh to 691205-5678. A video inset shows a man speaking. The footer contains logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES.

So far we have been looking at association between classes but what about the objects? We said that if the association cardinality is given, we know one object of a class is associated with how many elements of the other class that we call as link. Here in the above class diagram, if this is the association relation then the cardinality here is 1 and therefore one object of the people class is associated with one element of the tax file class. But the links they get formed and dissolved as the program runs, these are dynamic in nature. For example, a person may get out of the tax bracket and his tax file link may get dissolved and there may be people who are not in the tax bracket and for whom there is no tax file.

So here even though it is written 1-1 but then it may so happen that at some instant, some of the objects do not have an associated object here but an object typically has one associated object. We call this as a link between objects. When there is association relationship between two classes links between the corresponding objects get formed. A link is actually an instance of an association. It exists between 2 or more objects and these links as the system runs, may get formed for example one person who did not have a tax file becomes eligible to pay taxes and then tax file for him is created.

Another example may be that we have person with company, person and company are the classes and employs is the association relation. Now an employee works for a company, a company employs many people but then it may so happen that a person has a link, a person A that is an

object of person has a link with a company, let us say Infosys or something. Now there is a link between A and Infosys but let us say A resigns from Infosys and joins another company let us say TCS. So the link between A and Infosys gets dissolved as the system runs and he joins TCS. So the link between A and TCS gets established. We will see that how in the program it happens a little later.

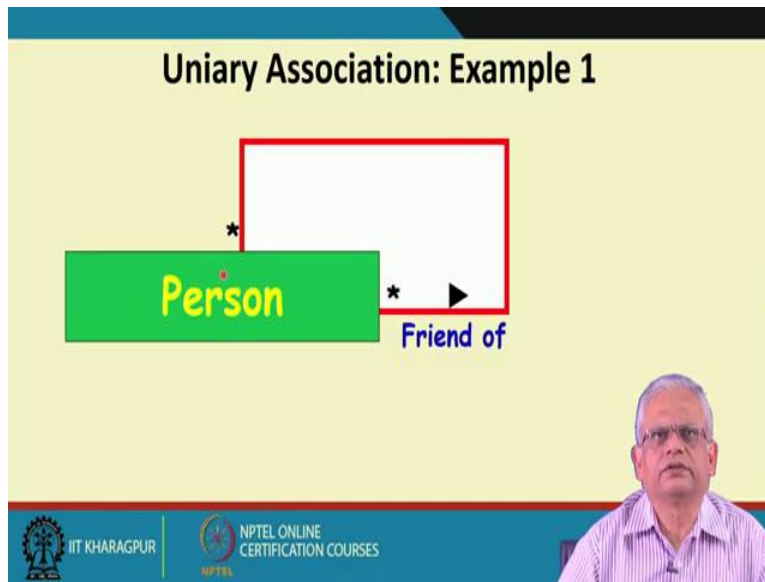
(Refer Slide Time: 21:30)

The slide features a yellow background with a blue header and footer. A yellow box in the top right corner contains the title 'Unary Association'. The main content consists of three bullet points: 'A class can be associated with itself (unary association). -Give an example?', 'An arrowhead used along with name: -Indicates direction of association.', and 'Multiplicity (association cardinality) indicates # of instances taking part in the association.'. A small video feed of a man with glasses is visible in the bottom right corner. The footer includes the logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES.

- A class can be associated with itself (**unary** association).
–Give an example?
- An arrowhead used along with name:
–Indicates direction of association.
- Multiplicity (association cardinality) indicates # of instances taking part in the association.

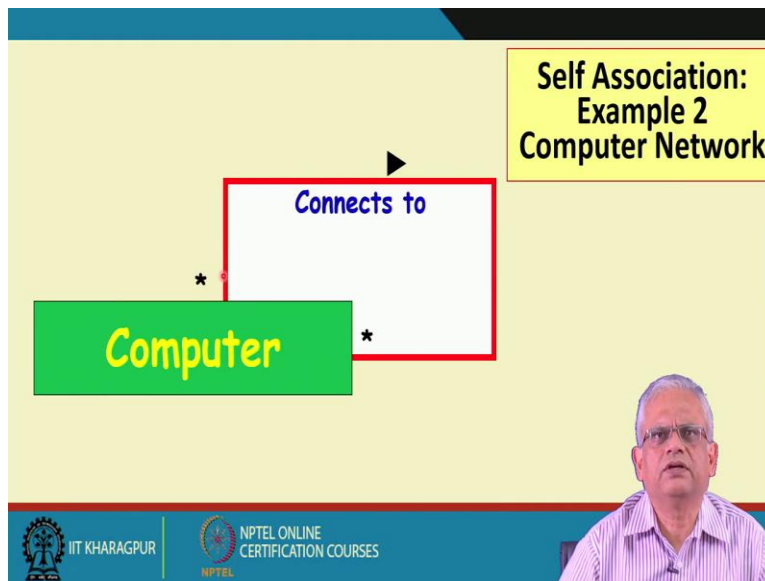
Now we had so far been looking at binary association that is between 2 classes but is it possible that a class, single class is associated with itself, is it possible that a single class is associated with itself and this we call as a unary association. Can you think of a unary association? Give an example of a unary association that is a class is associated with itself. The association relationship exists on one class. Again here we use an arrowhead to indicate the direction of association and we also write the multiplicity but please think of an example where we have a unary association that is an association existing on a single class.

(Refer Slide Time: 22:40)



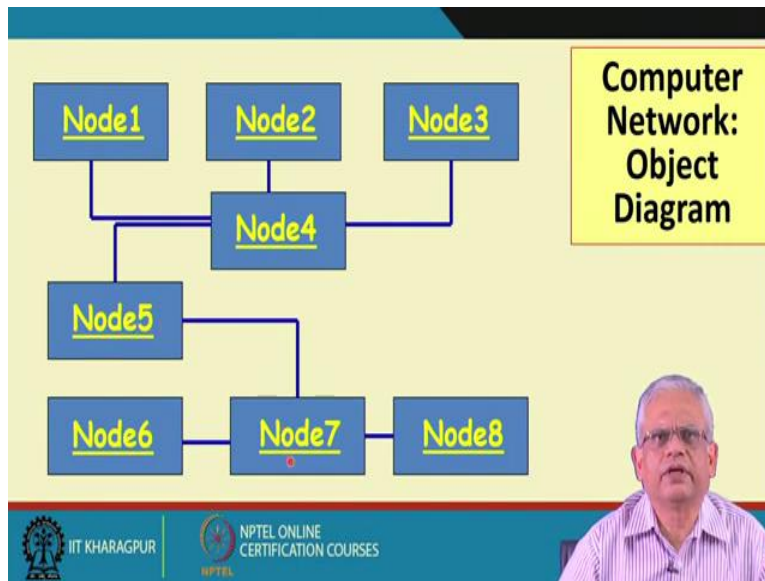
We give the first example here, A person is a friend of another person, A person is friend of many persons. The star implies that it is many, A single person is a friend of many persons. We can read in the other direction, a person has many persons as his friend but then please observe here that the object that we have here we are talking of the object, the friend has other objects in this, the link gets formed with the other objects of the person class. So a person A might have friends B, C, D and so on.

(Refer Slide Time: 23:57)



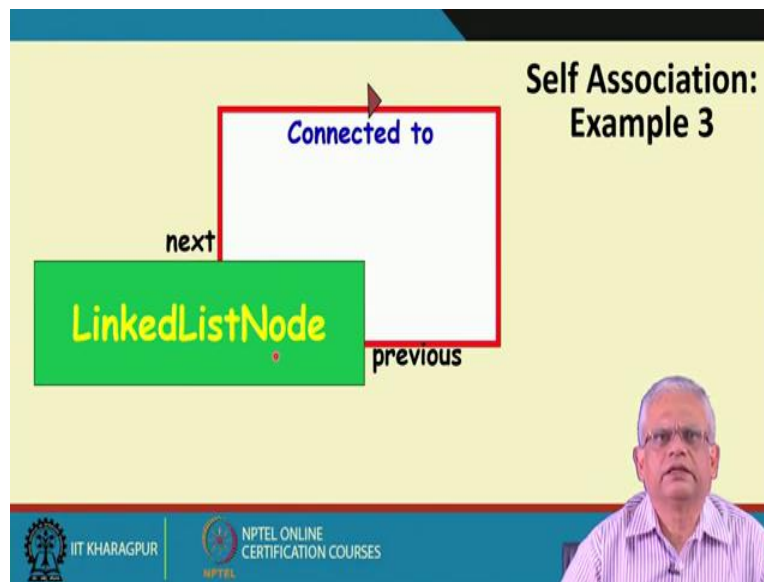
Now let us look at another example, A computer connects to many other computers or A computer is connected to many other computers. So that is indicated by the star here which is many. But then how does it look like in the objects space? We might have a computer A and that may be connected to either B, C and a computer M might be connected to L, M, N, O, L, O, P, Q etcetera and so on.

(Refer Slide Time: 24:50)



If we can represent this in the form of the object diagram may look like this that computer 1 or node 1 is connected to computer 4, so that is a single connection here. But node 4 that is computer 4 is connected to 4 nodes here. 1, 2, 3, 5 and 5 is connected to only 1 node 7 but node 7 is connected to 6 and 8.

(Refer Slide Time: 25:18)



There is another example here, a link list connects to the next node, so next is the reading direction here. A node in a link list is connected to the next node. We can read it in the other direction as A link list node is connected to the previous node. So far we have looked at several examples of unary association. Here the association is defined on a single class that is only the objects of this class participate in the association relation the links exists between different nodes of the same class, different objects of the same class. We are almost at the end of this session, we will stop here and continue in the next session. Thank you.