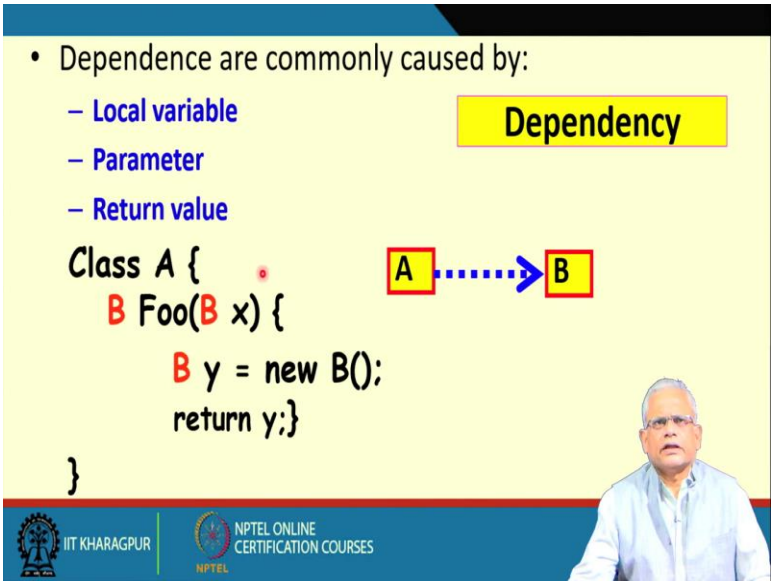
Object-Oriented System Development Using UML, JAVA and Patterns
Professor Rajib Mall
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture 15
Dependency Relation

Welcome to this lecture.

In the last lecture we were looking at the dependency between classes. The dependency between classes is represented by a dotted arrow and we had discussed about several causes for dependency. Now, let's look at the programming aspect of it. The commonly occurring reasons for dependency, and then how is it present in the program code.

(Refer Slide Time: 0:52)



The commonly caused dependencies are: An object of another class which appears as a local variable in a class, another reason that the object of one class is used as a parameter in the dependent class, the return value is another reason where a method returns an object as a return value and this object is a instantiation of an independent class. So, the dependent class uses an object of the independent class as a return value. These are three common reasons for dependency.

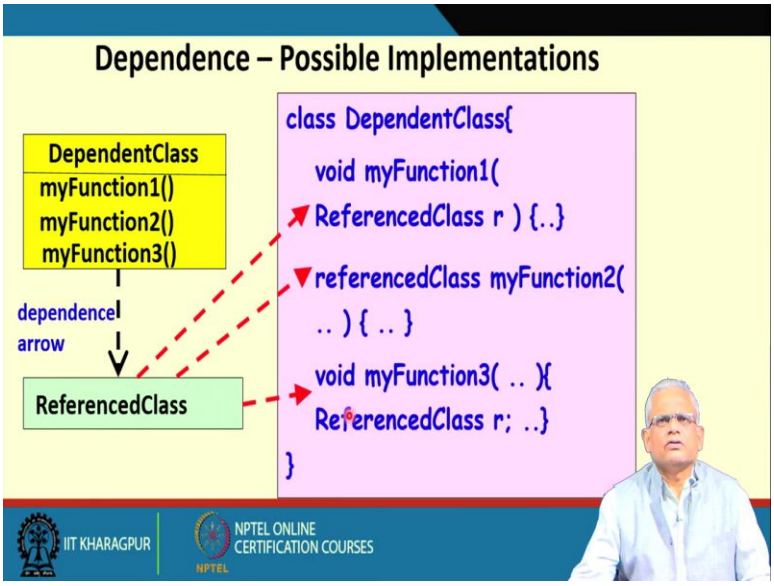
Now, let's look at this how it occurs in the program code (in the above slide). Let's look at this class A. Now, class B object is x. So, the class B object is used as a parameter in a

method of class A. This is one reason for dependency. Another reason of dependency is that an object of class B is returned by the method ‘Foo’ of class A. The third reason is that ‘y’ is an object of class B and it return in class A.

Any of these occurs in a dependent class, it will cause the dependency on an independent class. We also said that if there is a dependency, then any change to the independent class would require a change in the dependent class. It may lead to some code changes or may just require recompilation. In this example, at least recompilation of class A is required because class A is the dependent class.

So, just to recapture three common reasons for dependency between classes is that an object is a parameter in a dependent class method, the second reason is that in a dependent class, an object of another class, which is the independent class, is used as a local variable, and the third reason is that an object of an independent class is returned as the result of some method. So, these are three commonly occurring reasons for dependency between classes.

(Refer Slide Time: 4:31)



Now, let say we have this class diagram (in the above slide) that we have a dependent class ‘DepemdentClass’, which is depending on some reference class and we had said

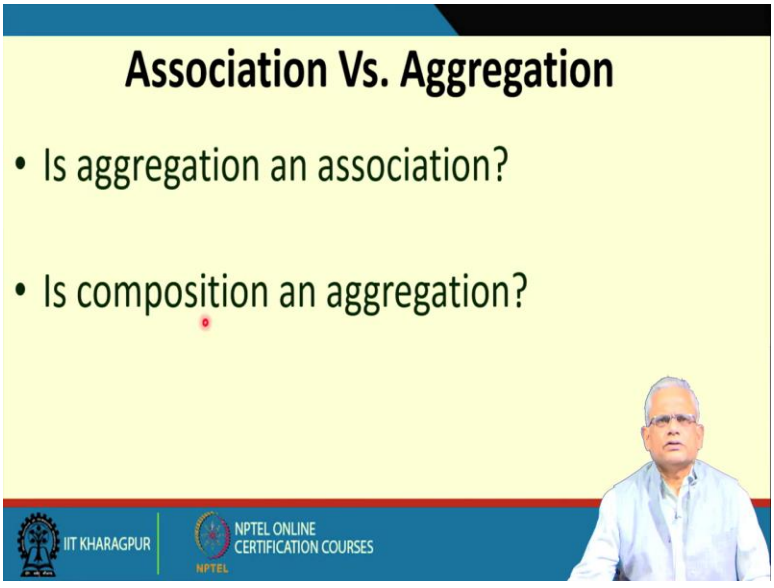
that this dotted arrow is a dependence arrow (black dotted arrow in the above slide diagram). Now, how do we write the Java code for this?

There are three common reasons for dependency and we don't know really which is present, but assuming that all are present. We might have a situation like the dependent class, reference class object as a parameter. Here, myFunction1 of DependentClass has a parameter 'r' of ReferencedClass.

Another reason maybe that some function returns as a parameter of the reference class (myFunction2 has return type referencedClass) and the third reason is that some function has a local variable of the reference class (myFunction3 has a local variable of reference class). Just to summarize given a code we can determine if there is a dependency between two classes, but given a dependency model here we cannot uniquely write the code.

The dependency might be caused due to various reasons and it is normally not possible to come up with a unique code for such a model. So, we have just seen an example of possible code from a given dependency diagram.

(Refer Slide Time: 6:50)



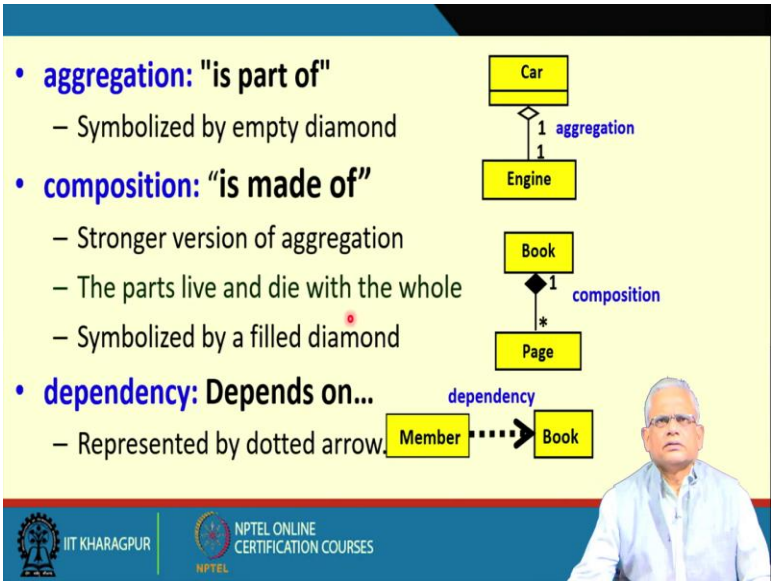
Now let me just ask one question, between association, aggregation and composition. Is aggregation an association relation?

The answer is that, yes. An aggregation is an association relationship, but it is a special type of association relationship. It means something more than plain association. We had said that not only that aggregation implies having the ID's of the associated class stored as implicit local variables, but also it implies that the aggregate creates the component classes.

Now, is composition an aggregation?

The answer to this is also, yes. The composition is an association relationship between two classes and also composition implies that the composite creates the components. Additionally, composition also implies that the lifelines of the component and composite are the same that is the components are created along with creation of the composite and also the components are destroyed when the composite gets destroyed. So, a composition relation is an aggregation relation but something more than that.

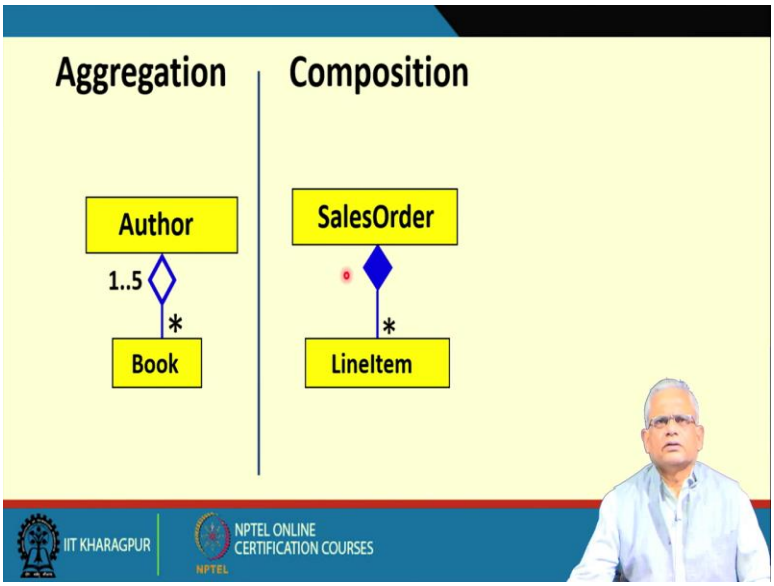
(Refer Slide Time: 8:50)



Now just to recapture what we discussed so far. An aggregation relation is a part of relation between one class and other, and we had said when something is aggregation means something aggregate something and when something is a composition means something composed by something. the plane aggregation is represented by a diamond symbol. A composition typically is 'made of' relation. It is a stronger form of aggregation

in that the parts live and die with the whole and is represented using a filled diamond. The dependency relation is represented by a dotted arrow and just implies that the dependent class depends on the independent class, in the sense that any changes to the independent class would need changes to the dependent class.

(Refer Slide Time: 10:04)



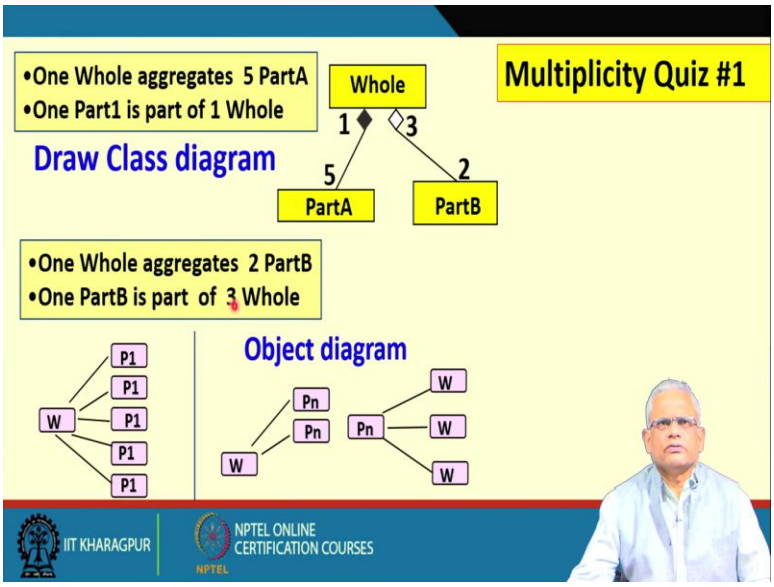
Now, let just make a comparison between aggregation and composition.

Let see the above example of the slide. The aggregation relation of the slide says that an author has many books. An author aggregates books means writes many books, but if we read from another side, a book is written by one to five authors. So, this is a correct model of the situation, when author writes many books, but it may possible a book is written anywhere from one to five authors.

But when we have a composition relation, we cannot have anything other than ‘1’ on the composite end because the component belongs to exactly one composite. Here a sales order contains many LineItem, but a Lineitem belongs to exactly one sales order. We cannot have more than ‘1’ on the composite side that would be incorrect because in composite relation LineItem belongs to exactly one sales order.

I hope that is clear about the multiplicity on the aggregator end and the composite end. In the aggregation relationship we can have any multiplicity here, on the other hand in the composition relation we cannot have anything other than ‘1’ on the composite end.

(Refer Slide Time: 12:20)



Now, let's just try to do a small quiz here (in the above slide). How do we represent one whole? whole is the class name here. One whole aggregate five PartA and one PartA is part of one whole (in slide instead of Part1, it will be PartA in text description). So, 'whole' is a class and 'PartA' is a class. So, one whole aggregate five PartA, but each PartA is part of exactly one whole.

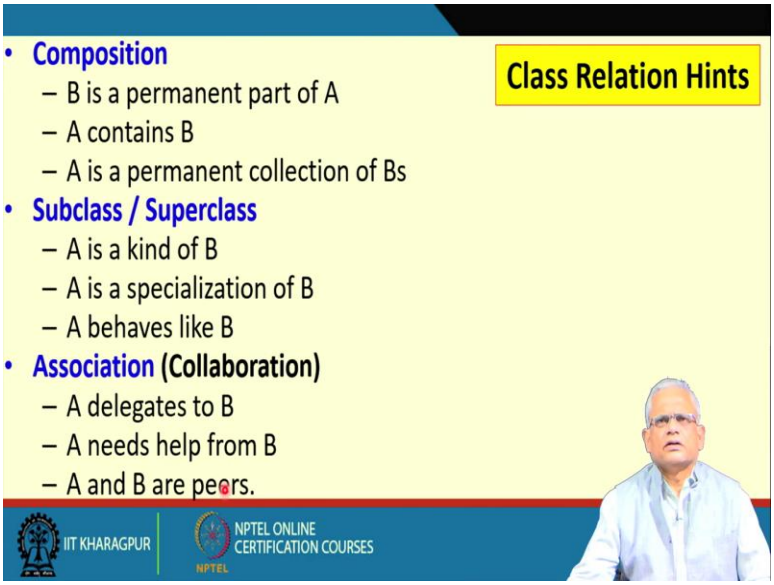
If we draw the class diagram, this would look like this (class diagram in the above slide). If we say here one whole is made up of five PartA then we will use a filled diamond but if we just say aggregates we can as well use an empty diamond also. So, the point is one whole is made up of five PartA then, we will use a filled diamond but right now with one whole aggregate five PartA we can use an empty diamond. In the slide, filled diamond shown but instead of that we can use empty diamond because in the statement it says 'One whole aggregate five Part A'.

And now, let's do another part of this exercise, one whole aggregate 2 PartB. In the first part of the question if it is written here that one whole is made up of five PartA then the composition relation is okay, otherwise we could have used a simple aggregation relationship that's an empty diamond. Now, here in the second question it is written that one whole aggregate 2 PartB and one PartB is part of 3 whole.

So, here one whole aggregate 2 PartB and each PartB is part of 3 whole (shown in the above slide class diagram).

If we look at the object diagram of the above slide, then we will see that for the first question, one whole is associated with five parts and each part is associated with one whole (left side first object diagram in the slide), but for the second one, one whole is associated with 2 Part B and for each of this part it can be associated with 3 wholes. So, for the second one two object diagram shown in the above slide.

(Refer Slide Time: 15:43)

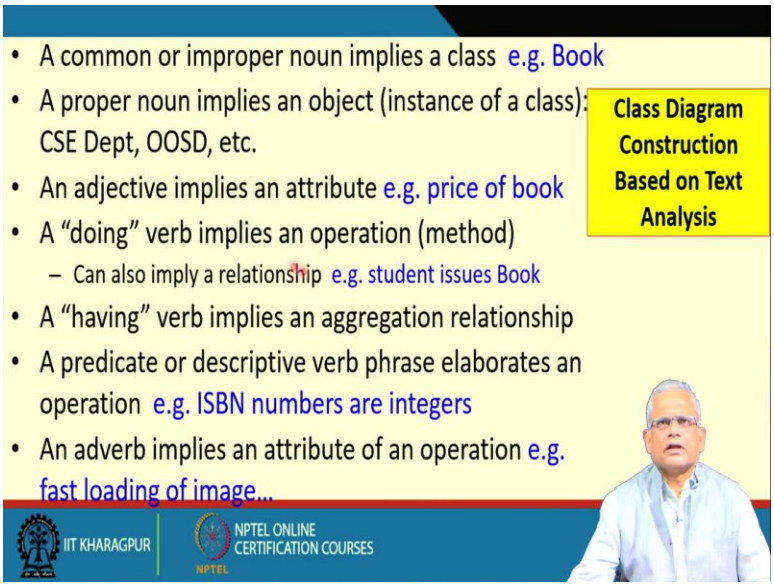


Now, given a text description (some example in the above slide) how do we know what kind of relationship exists between classes?

We can infer about composition relation: if the text contains statement like B is a permanent part of A, A is made up of B or A contains B, A is a permanent collection of B's then it indicates composition relations. On the other hand, an inheritance relation is indicated by statements like is kind of, is specialization of, behaves like, etc. Association is indicated by phrases like delegates, responsibility to B, A needs help from B, A and B are peers and help each other, and so on.

Later will do some exercises, we have some text descriptions we like to identify the classes and the class relations from that.

(Refer Slide Time: 17:04)



Now, let see a text analysis of a problem (in the above slide) to be solved. Using text analysis, we can create a model and later we would like to write the program for that.

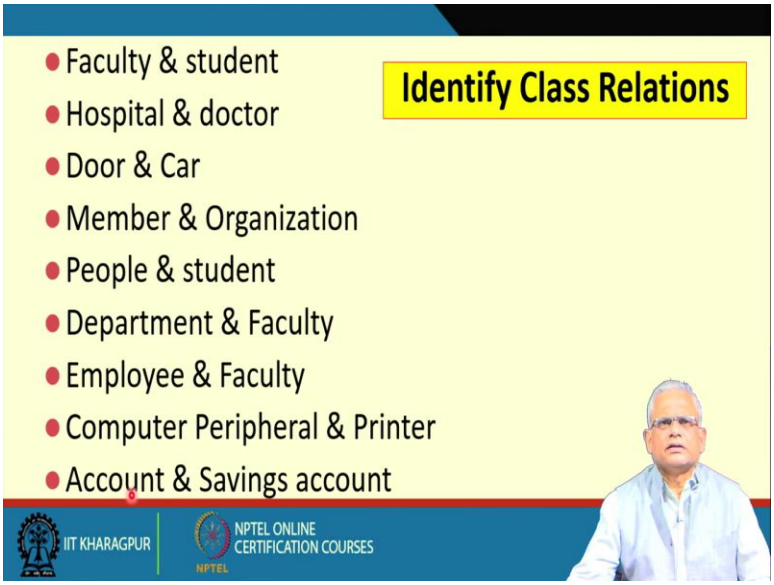
How do we analyze the text description to come up with meaningful class model?

An improper noun or a common noun implies a class, for example, a book, a member, register and so on. All these are improper nouns or common nouns. On the other hand the proper noun like name of a person, CSE Department, OOSD course and so on -- these are specific objects or instances of class. These appear as proper nouns and adjectives indicates an attribute. For example, price of book, book is a class and price is an attribute of the book. If there is some action represented in the text, for example, doing, then it represents a method or an operation.

Of course, we have to be careful that it can also imply association relation, for example, student issues a book. Normally methods are represented as verbs, but then some of the verbs may represent association relation. 'Something has or having,' this represents aggregate relationship. A predicate or descriptive verb represents an operation. It's basically gives an elaboration of an operation. For example, ISBN numbers are integers. So, here we know that it deals with integers, some method deals with an integer or we can also represent this as an attribute and say that ISBN numbers represented as

attributes. An adverb implies an attribute, for example, fast loading of an image; it is an attribute of an image.

(Refer Slide Time: 20:06)



Now, let's do some exercise (in the above slide) because based on exercise we will develop the important skill of identifying classes and relationship from text description.

Faculty and student: assuming that these are two classes. What is the relationship between faculty and student? If you can think of the relation that faculty teaches student, then this is an association relation.

Hospital and Doctor: Here the relation is that the hospital employees many doctors, so this is an aggregation relationship between a hospital and a doctor.

Door and car: These are two classes and the car have four doors. So, it is a composition relation.

Member and organization: An organization has many members and therefore it is an aggregation relation.

People and student: Here a student is a special type of person or people, a student is a special type of people, and therefore ‘student’ is the derived class and ‘people’ is the base class. So, it’s an inheritance relationship.

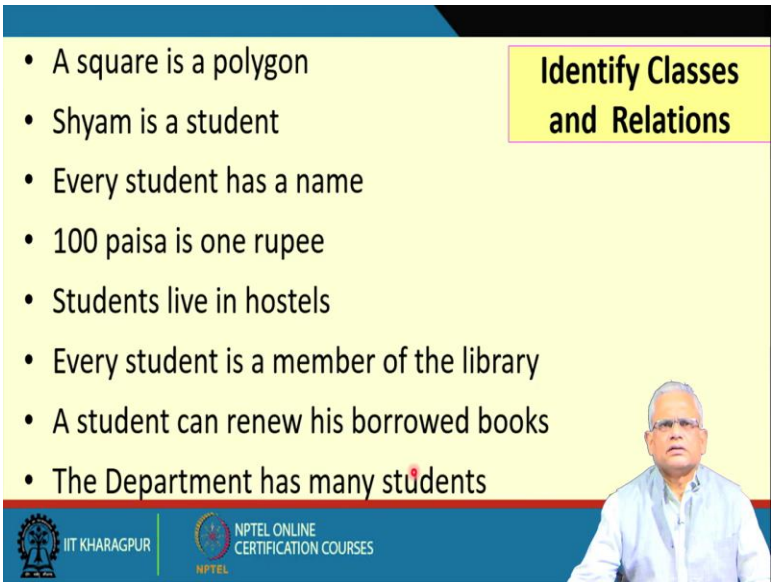
Department and faculty: A department has many faculties and therefore the relationship between these two classes is aggregation. Department is the aggregate class and faculty is the component class.

Employee and faculty: A faculty is an employee or we can say that a faculty is a special type of employee, there are many employees and faculty is a special type of employee. So, this is a ‘IS A’ relation. faculty is an employee indicates that it is an inheritance relation. Employee is the base class and faculty is the derived class.

Computer peripheral and printer: A printer is a special type of computer peripheral which indicates that computer peripheral is the base class and printer is the derived class.

Account and savings account: There are many types of account, checking account, savings account, deposit account and so on. So, savings account is a special type of account and therefore the relation is an inheritance relation.

(Refer Slide Time: 23:30)



Now, let’s do some more exercise (in the above slide). If you have a sentence, how you can identify the relations? Let’s do some example exercise.

A square is a polygon: So, what is the relation between the square class and the polygon class? A square is a special type of polygon. So, we can say that polygon is the base class and square is the derived class.

Shyam is a student: Shyam is a proper noun and therefore, represents an object of the student class. So, here student is the class and Shyam is an object of student class.

Every student has a name: Here name is an attribute of the student and student is the class.

100 paisa is one Rupee: This is the composition relation. The Rupee is made up of 100 paisa because we cannot have a Rupee with a 99 paisa or 101 paisa. So, a Rupee will always have 100 paisa and therefore it is a composition relation.

Students live in hostels: Students and hostels are two classes and lives in is the association relation between these two classes.

Every student is a member of the library: So here student and library are two classes and ‘is member of’ is the association relation.

A student can renew his borrowed book: So here student and the book are the two classes and renew represents the association relationship between these two classes.

The department has many students: Here the ‘has’ indicates an aggregation relation and department is the aggregate class and student is the component class.

Later we will do analysis of larger text, that’s normally the situation when you solve a problem that is you want to write a program to implement some problem, then the problem is described in the form of a text description spanning few paragraphs or pages.

We have been trying to develop our skill by working out on small sentences and later we will have paragraphs and we try to develop the class diagram based on that and of course,

solving real-life problems will require reading multiple paragraphs and developing the class diagram based on that.

We are almost at the end of this session. We will stop here and continue in the next session.

Thank You.