Topic 6: object -oriented Design Implementation

❖Programming and Development Process

• Coding is an end goal of software development.

• Iterative and incremental development process results in the feeding of prior iteration into the beginning of next iteration, continuously refining the implementation works.

❖ Mapping Design to code

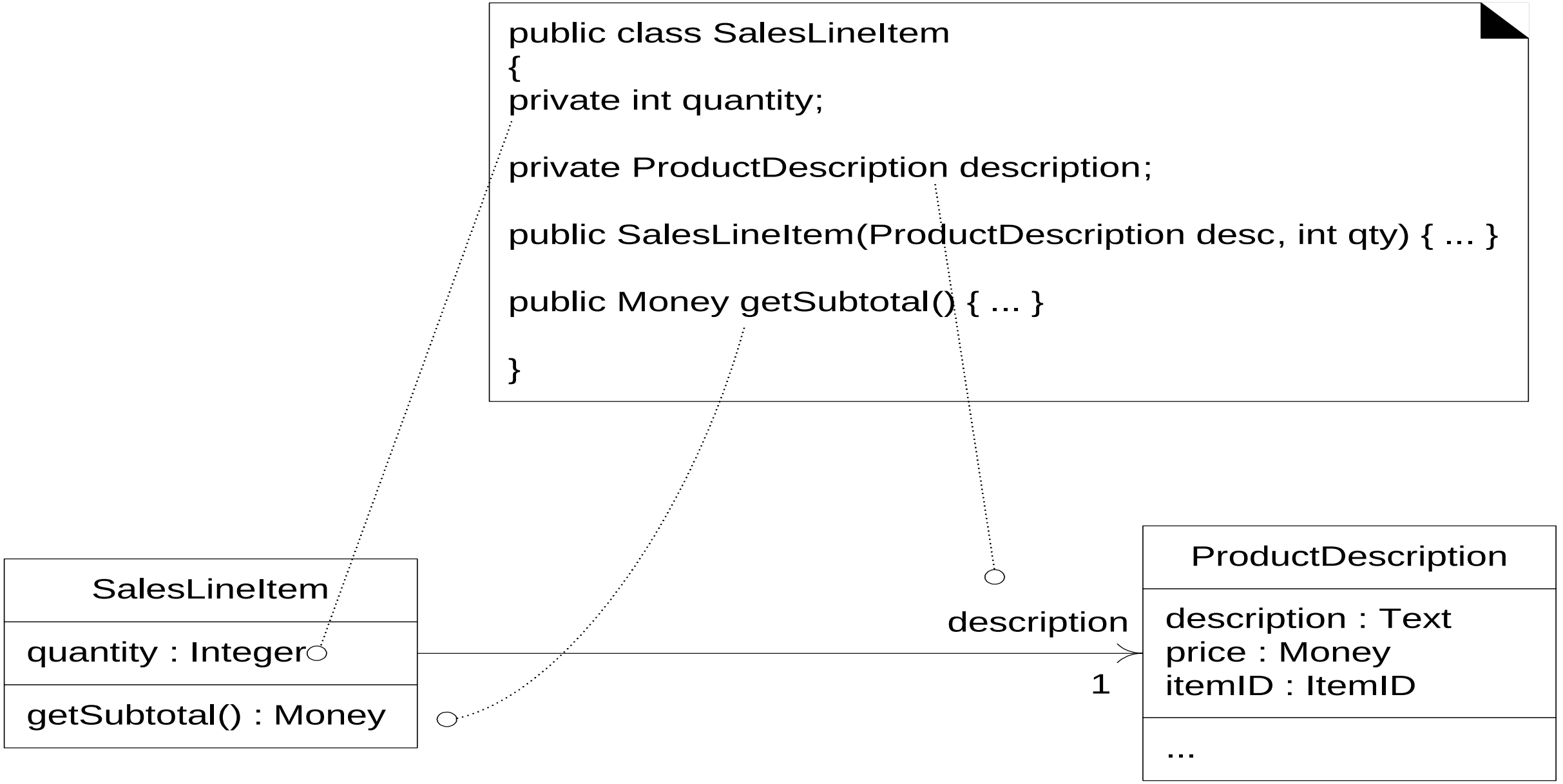
- The goal of mapping design to code in Object-Oriented Analysis and Design (OOAD) is to transform our concepts and blueprints into functional software.
- We convert our designed ideas—such as classes, objects, and relationships—into the programming language. To do this, we must translate our models and diagrams into computer-readable code.

a) Class and interface definitions
b) Method definitions

➢Class definitions are created by mapping design class diagrams to code.

➢Method definitions are created by mapping interaction diagrams to code.

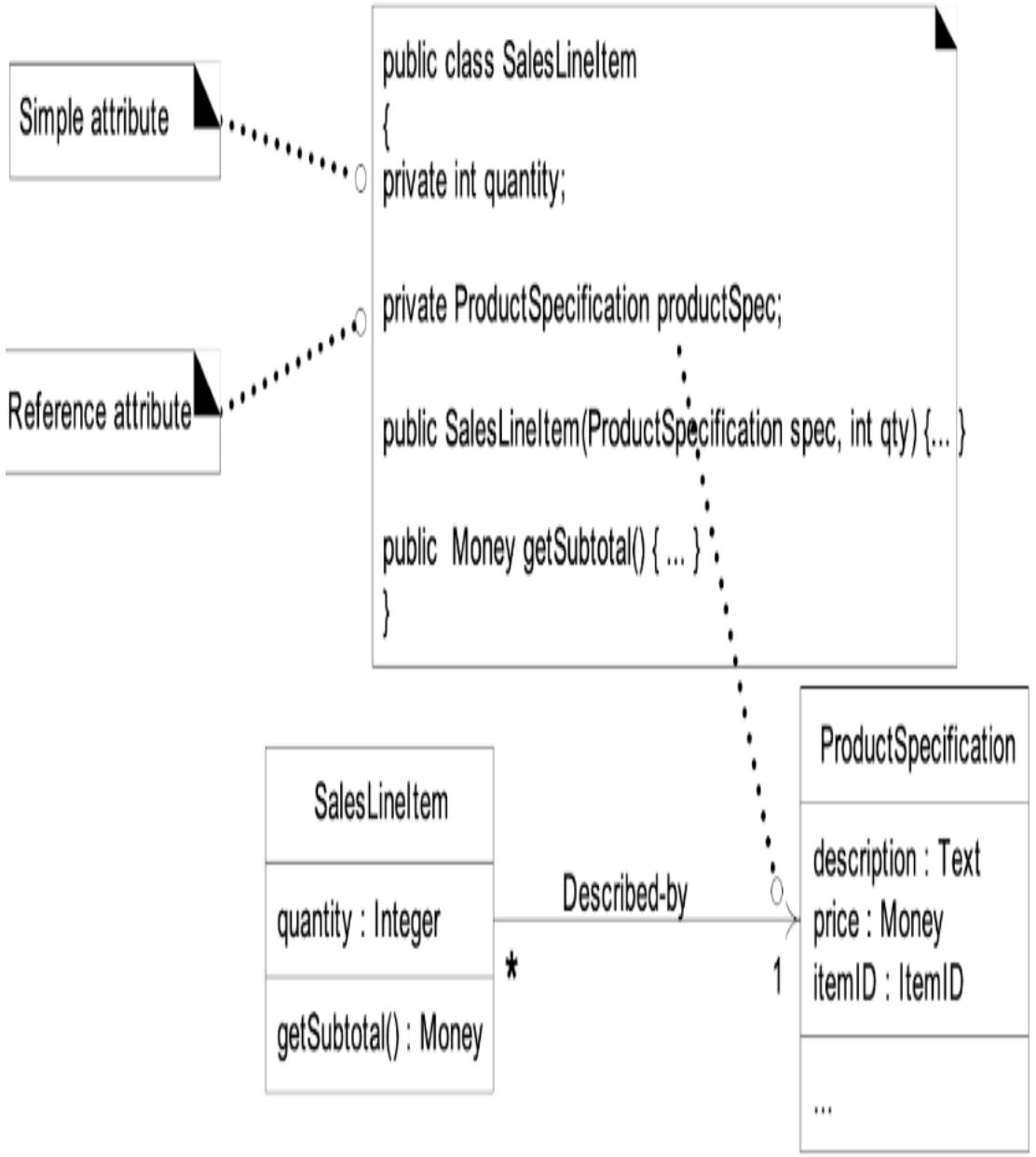
Creating Class Definitions from Design Class Diagram



Adding Reference Attributes

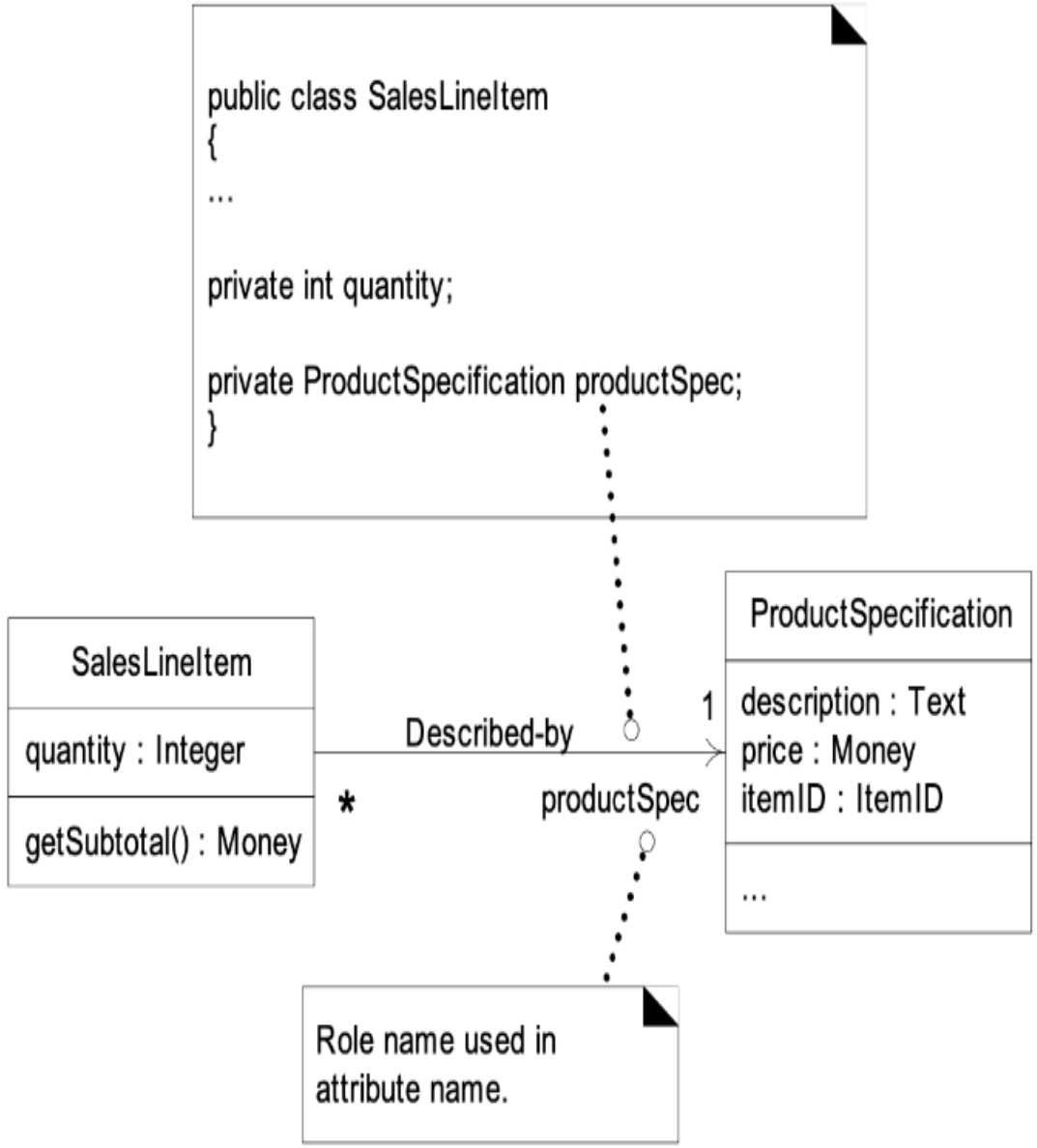
- A reference attribute is an attribute that refers to another complex object, not to a primitive type such as a String, Number, and so on.

- The reference attributes of a class are suggested by the associations and navigability in a class diagram.



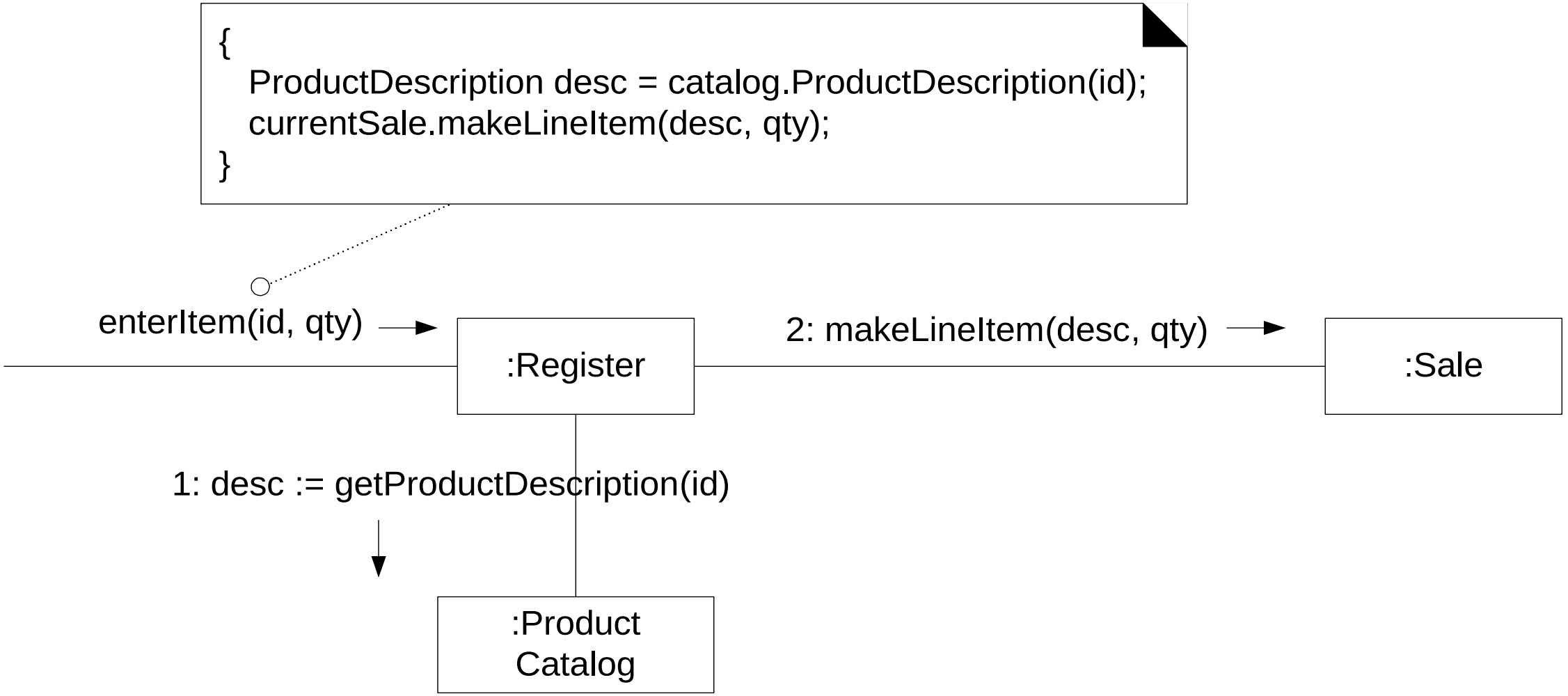
Adding roles names

- The next iteration will explore the concept of role names in static structure diagrams. Each end of an association is called a role. Briefly, a role name is a name that identifies the role and often provides some semantic context as to the nature of the role.
- If a role name is present in a class diagram, use it as the basis for the name of the reference attribute during code generation.



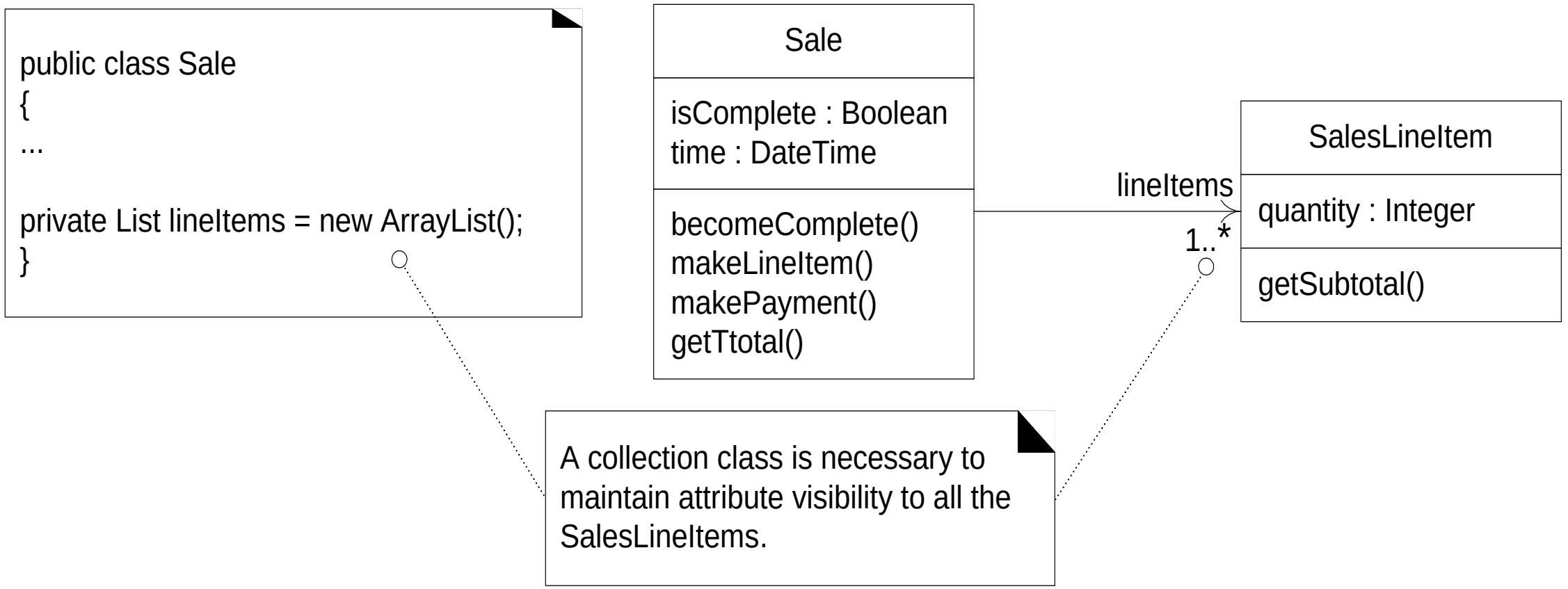
❖ Creating methods from collaboration Diagram

- The sequence diagram consists of sequence of messages which are translated to a series of statements in the method definitions.



❖ Updating class definitions

- One to many relationships are common.
- Such relationships is implemented using collection object such as list, map or array.
- The choice of collection class is influenced by the requirements. i.e. key based lookup requires Map while growing ordered list requires a List.
- If object implements an interface, declare the variable in terms of the interface.


public class Sale
{
...
private List lineItems = new ArrayList();
}
Sale
isComplete : Boolean
time : DateTime
becomeComplete()
makeLineItem()
makePayment()
getTtotal()
lineItems
1..*
SalesLineItem
quantity : Integer
getSubtotal()
A collection class is necessary to maintain attribute visibility to all the SalesLineItems.

❖ Exception and error Handling

Exception

- An exception is a condition that is caused by a runtime error in the program.
- An exception may occur due to following reasons:

a) Invalid data entered by a user.
b) File to be opened can not be found.
c) The network connection has lost in the middle of the communication

Sources for Exceptions

1. User errors
2. Programmer errors
3. Physical resource failure

Categories of Exception

1. Checked Exception

- It is the exception that can not be foreseen by the programmer.
- Eg: FileNotFoundException

2. Runtime Exception

- It is the exception that could be avoided by the programmer.
- It is ignored at the time of compilation.

3. Errors

- They are the problems beyond the control of user and programmer.
- Eg: StackOverflowException

Exception handling process

- In object-oriented programming languages, there is a mechanism to handle exceptions in a proper manner.
- Try, throw and catch are the basic exception handling paradigms used.
- The general code is put in try block. It means try to execute the code.
- If the system succeeds to execute the code, execution flows in general or normal order.
- If something goes wrong while executing the try block, this code throws an exception object and stops executing code of try block.
- The error handler catches the exception object and make necessary actions needed.
- Execution continues with the next instructions following the catch block.

```
try:

    numerator = int(input("Enter the numerator: "))
    denominator = int(input("Enter the denominator: "))
    result = numerator / denominator
except ZeroDivisionError:

    print("Error: Division by zero is not allowed.")
except ValueError:

    print("Error: Invalid input. Please enter numeric values.")
else:

    print(f"The result is: {result}")

finally:

    print("Execution completed.")
```

THANK YOU

MCQ: OOD

1. What does a simple name in UML Class and objects consist of?
a) Letters
b) Digits
c) Punctuation Characters
d) All of the mentioned

2. What Does a Composite name consist of in a UML Class and object diagram?
a) Delimiter
b) Simple names
c) Digits
d) All of the mentioned

3. A Class consists of which of these abstractions?
a) Set of the objects
b) Operations
c) Attributes
d) All of the mentioned

4. A class is divided into which of these compartments?
a) Name Compartment
b) Attribute Compartment
c) Operation Compartment
d) All of the mentioned

5. What should be mentioned as attributes for conceptual modelling?
a) Initial Values
b) Names
c) All of the mentioned
d) None of the mentioned

6. What among the following statement is true?
a) Associations may also correspond to the relation between instances of three or more classes
b) Association lines may be unlabeled, or they may show association name
c) All of the mentioned
d) None of the mentioned

7. What is **multiplicity** for an association?
a) The multiplicity at the target class end of an association **is the number of instances** that can be associated with a **single instance o**f source class
b) The multiplicity at the target class end of an association is **the number of instances** that can be associated with a **number instance of** source class
c) All of the mentioned
d) None of the mentioned

8. Which among these are the rules to be considered to form Class diagrams?
a) Class symbols must have at **least a name compartment**
b) Compartment can be in random order
c) Attributes and operations can be listed at any suitable place
d) None of the mentioned

9. Which of the following statement is true?
a) A transition is a change from one state to another
b) Transitions may be spontaneous, but usually some event triggers them.
c) An event is a noteworthy occurrence at a particular time; events have no duration.
d) All of the mentioned

10. Which of the following determines the state diagram?
a) The UML notation for specifying finite automata is the state diagram
b) In state diagrams, states are represented by rounded rectangles
c) All of the mentioned
d) None of the mentioned

11. Mid -level generation design techniques are classified into which of the following?
a) Creational Techniques
b) Transitional Techniques
c) All of the mentioned
d) None of the mentioned

12. Why does designers look for candidate classes?
a) To model entities in charge of or involved in program tasks
b) To model things in the world that interact directly with the program
c) To model structures and collections of objects
d) All of the mentioned

13. Which of the following is referred for the conceptual modelling?
a) Change actors to interface classes
b) Add actor domain classes
c) Convert or add controllers and coordinators
d) All of the mentioned

14. What is the Interaction diagram?
a) Interaction diagrams are the UML notations for dynamic modeling of **collaborations**
b) Interaction diagrams are a **central focus of engineering design.**
c) All of the mentioned.
d) None of the mentioned

15. What are the different interaction diagram notations does UML have?
a) A sequence diagram
b) A communication diagram

c) An interaction overview diagram
d) All of the mentioned

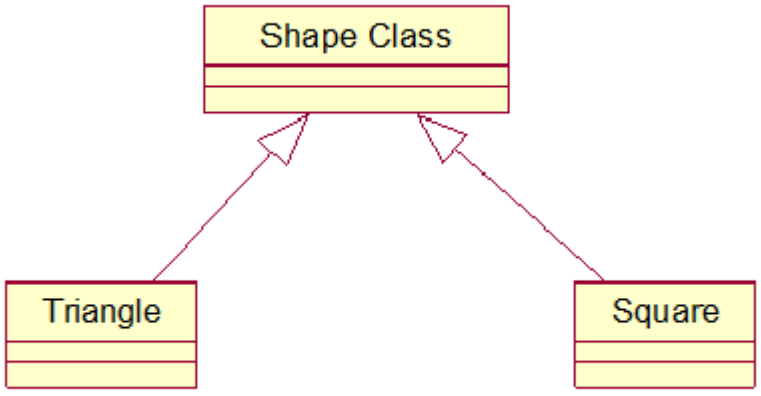
16. What is a sequence diagram?
a) A diagram that shows interacting individuals along the top of the diagram and messages passed among them **arranged in temporal order** down the page
b) A diagram that shows messages superimposed on a diagram depicting collaborating individuals and the links among them
c) A diagram that shows the change of an individual's state over time
d) All of the mentioned

17. What does a message mean?
a) It Passes all communications from one object to another and are represented by message arrows in sequence diagrams
b) The message goes from the sending object's lifeline to the receiving object's lifeline
c) It is a rectangle containing an identifier with a dashed line extending below the rectangle
d) All of the mentioned

18. What are the interaction fragments?
a) A fragment which is a rectangular frame with a pentagonal operation compartment in the upper left-hand corner
b) A fragment which has a marked part of an interaction specification
c) The regions resulting from these divisions will not hold the interaction fragment operations
d) All of the mentioned

19. Which of the following UML diagrams has a static view?
a) Collaboration
b) Use case
c) State chart
d) Activity

20. What type of relationship is represented by Shape class and Square?



a) Realization
b) Generalization

c) Aggregation
d) Dependency

21. Which diagram in UML shows a complete or partial view of the structure of a modeled system at a specific time?
a) Sequence Diagram
b) Collaboration Diagram
c) Class Diagram
d) Object Diagram

22. Which of the following diagram is time oriented?
a) Collaboration
b) Sequence
c) Activity
d) None of the mentioned

23. Classes and interfaces are a part of
a) Structural things
b) Behavioral things
c) Grouping things
d) Annotational things

24. Which of the following term is best defined by the statement: “a structural relationship that specifies that objects of one thing are connected to objects of another”?
a) Association
b) Aggregation
c) Realization
d) Generalization

25.who considers diagrams as a type of Class diagram, component diagram, object diagram, and deployment diagram?

a) structural
b) behavioral
c) non-behavioral
d) nonstructural

26._____ represented by In UML diagrams, relationship between component parts and object.

a) ordination
b) aggregation
c) segregation
d) increment

27.Which of the following is correct list of classifications of design patterns.
a) Creational, Structural and Behavioral patterns

b) Executional, Structural and Behavioral patterns
c) Creational, Executional and Behavioral patterns
d)None of the above

28. Which of the following is a design pattern?
a) Behavioral
b) Structural
c) Abstract Factory
d) All of the mentioned

29. You want to minimize development costs by reusing methods? Which design pattern would you choose?
a) Adapter Pattern
b) Singleton Pattern
c) Delegation pattern
d) Immutable Pattern

30. The recurring aspects of designs are called design
a) patterns
b) documents
c) structures
d) methods

31. Which pattern prevents one from creating more than one instance of a variable?
a) Factory Method
b) Singleton
c) Observer
d) None of the mentioned

32. You want to avoid multiple inheritance. Which design pattern would you choose?
a) Abstraction-Occurrence Pattern
b) Player-Role Pattern
c) General Hierarchy Pattern
d) Singleton Pattern

33. Which design pattern defines one-to-many dependency among objects?
a) Singleton pattern
b) Facade Pattern
c) Observer pattern
d) Factory method pattern

34. Why are Patterns important?
a) They capture expert design knowledge.
b) They make captured design accessible to both novices and other experts
c) All of the mentioned
d) None of the mentioned

35. Which of the following Choices and standardizes patterns for a problem domain promotes software reuse and, hence, quality and productivity?
a) Promoting Communication
b) Streamlining Documentation
c) Increasing Development Efficiency
d) Supporting Software Reuse

36. What is a pattern?
a) It is a model proposed for imitation
b) It solves a software design problem
c) All of the mentioned
d) None of the mentioned

37. Which of the following UML diagrams has a static view?
a) Collaboration
b) Use case
c) State chart
d) Activity

1.(d) 11.(c) 21.(d) 31.(b)

2.(d) 12.(d) 22.(b) 32.(b)

3.(d) 13.(d) 23.(a) 33.(c)

4.(d) 14.(c) 24.(a) 34.(c)

5.(c) 15.(d) 25.(a) 35.(d)

6.(c) 16.(a) 26.(b) 36.(c)

7.(a) 17.(a) 27.(a)

8.(a) 18.(d) 28.(d)

9.(d) 19.(b) 29.(c)

10.(c)20.(b) 30.(a)