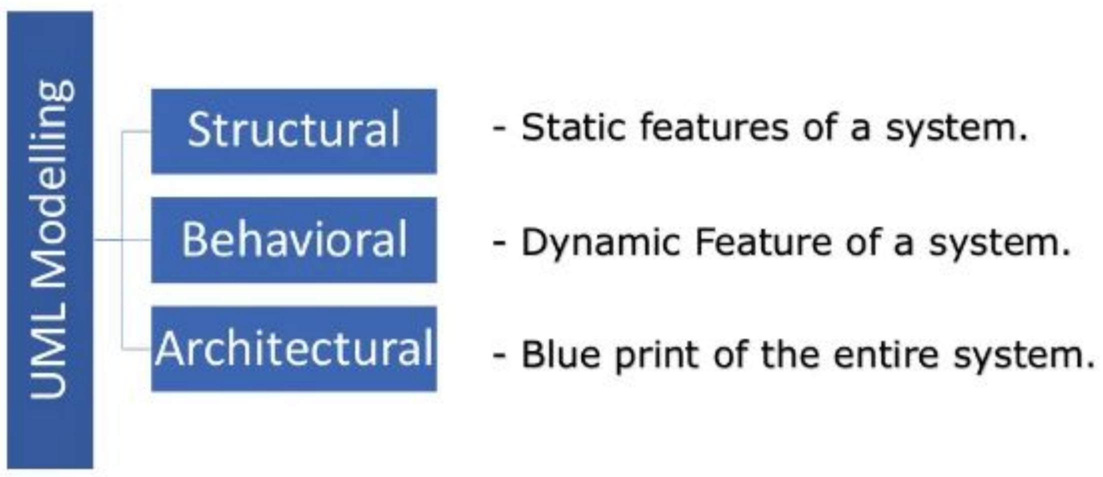
UML: Unified Modeling Language

- Class Diagram
- Object Diagram
- Use Case Diagram

UML: Unified Modeling Language

- UML stands for Unified Modeling Language
- It is a Visual Language
- It is Industry Standard Graphical Language for specifying,visualizing, constructing and documenting the artifact of the system.
- UML mostly uses graphical notations to express Object Oriented Analysis and Design of the software
- It simplifies the complex process of the software design

UML: Unified Modeling Language



UML Modelling

Structural - Static features of a system.

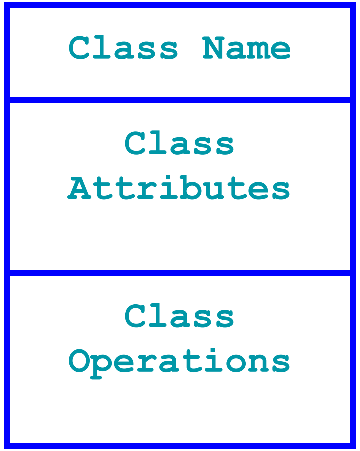
Behavioral - Dynamic Feature of a system.

Architectural - Blue print of the entire system.

Class Diagram

- It shows the classes of the system, their inter-relationships, the operations and the attributes of the classes.
- Explore domain concepts in the form of domain model.
- Analyze requirements in the form of conceptual/analysis model.
- Depict the detailed design of object oriented or object based software.

Class Diagram

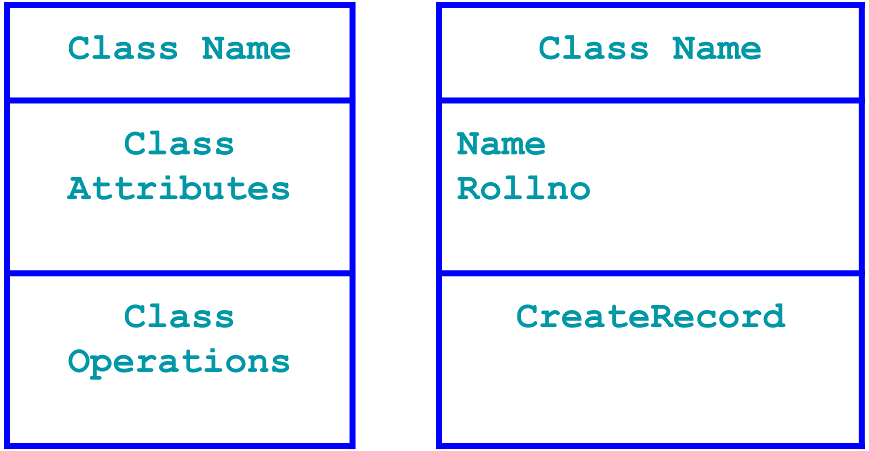


Class Name

Class Attributes

Class Operations

Class Diagram



Class Name

Class Attributes

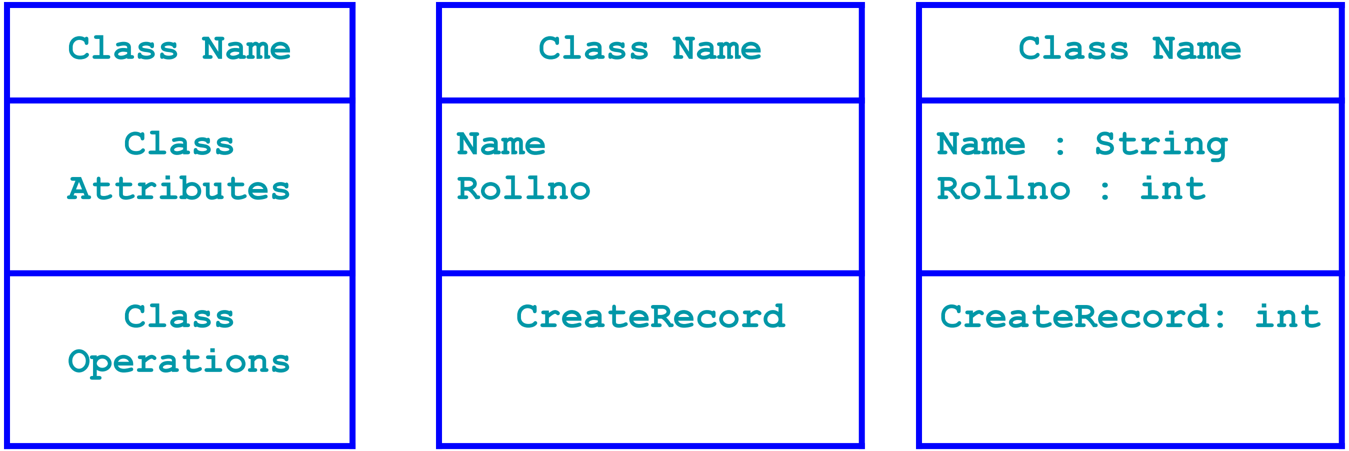
Class Operations

Class Name

Name
Rollno

CreateRecord

Class Diagram



Class Name

Class Attributes

Class Operations

Class Name

Name
Rollno

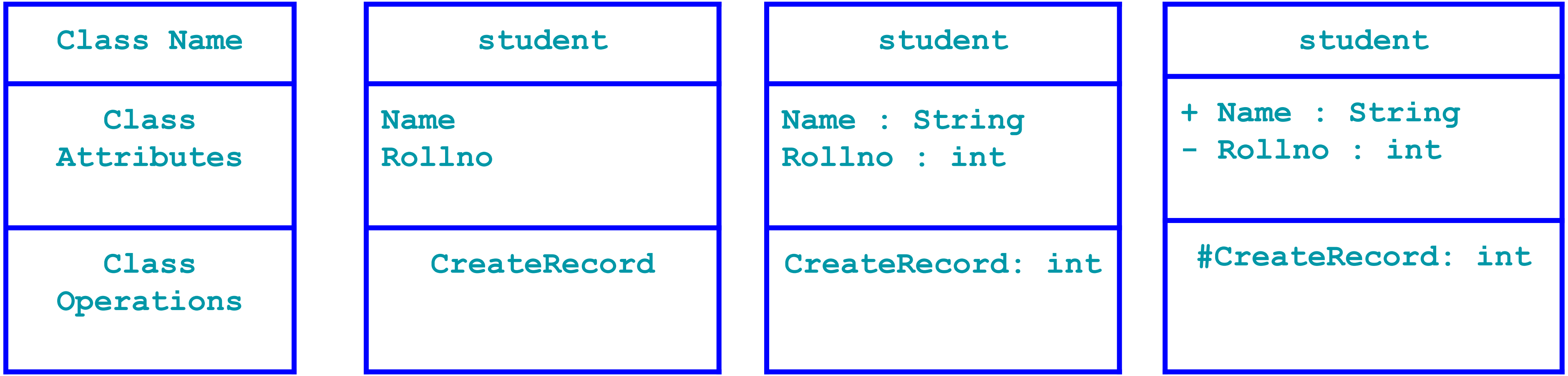
CreateRecord

Class Name

Name : String
Rollno : int

CreateRecord: int

Class Diagram



Class Name

Class Attributes

Class Operations

student

Name
Rollno

CreateRecord

student

Name : String
Rollno : int

CreateRecord: int

student

+ Name : String
- Rollno : int

#CreateRecord: int

Visibility Notations:

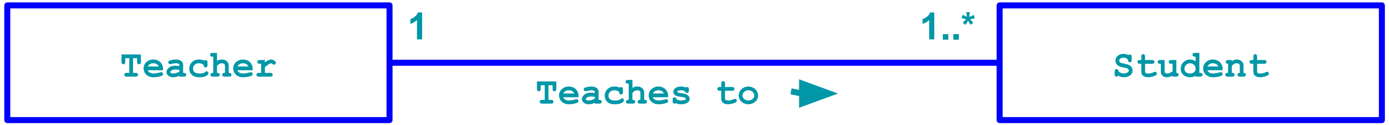
+ Public
- Private
# Protected

Class Diagram

- It has relationships between the classes
  - Association
  - Dependency
  - Aggregation
  - Composition
  - Generalization

Class Diagram

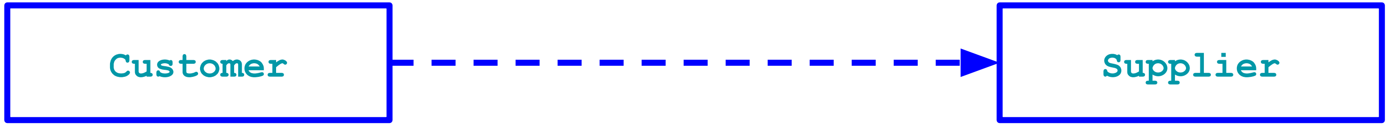
- **Association**



1
0..1
0..*
1..*
Or Exact numbers 3,4,5,..

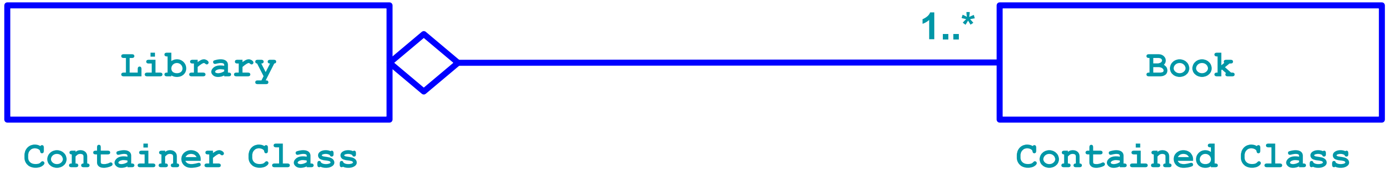
Class Diagram

- Dependency



Class Diagram

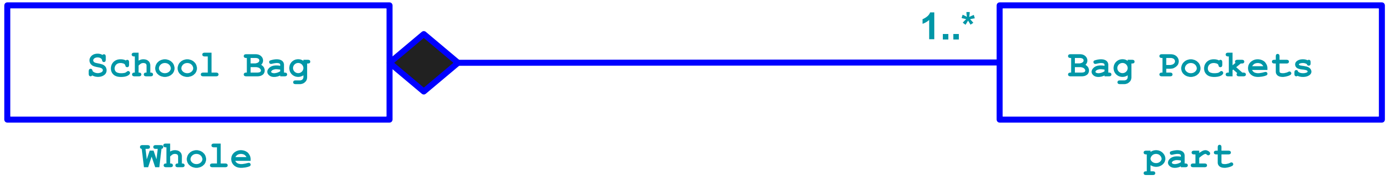
- **Aggregation**



Here, book class is not strongly depend on library class i.e. contained class is not strongly dependent on container class.

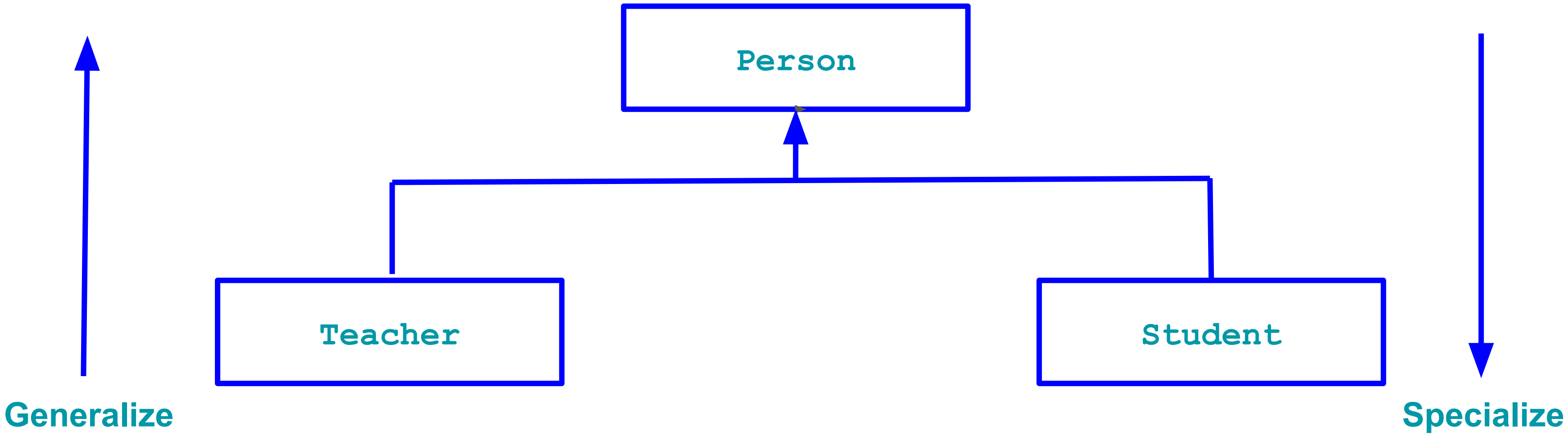
Class Diagram

- **Composition**
  - It is like aggregation.
  - Example: A school bag and its pockets.
  - If school bag is destroyed, then pockets automatically destroyed.
  - I.e. Contained class has strongly dependency on contained class.



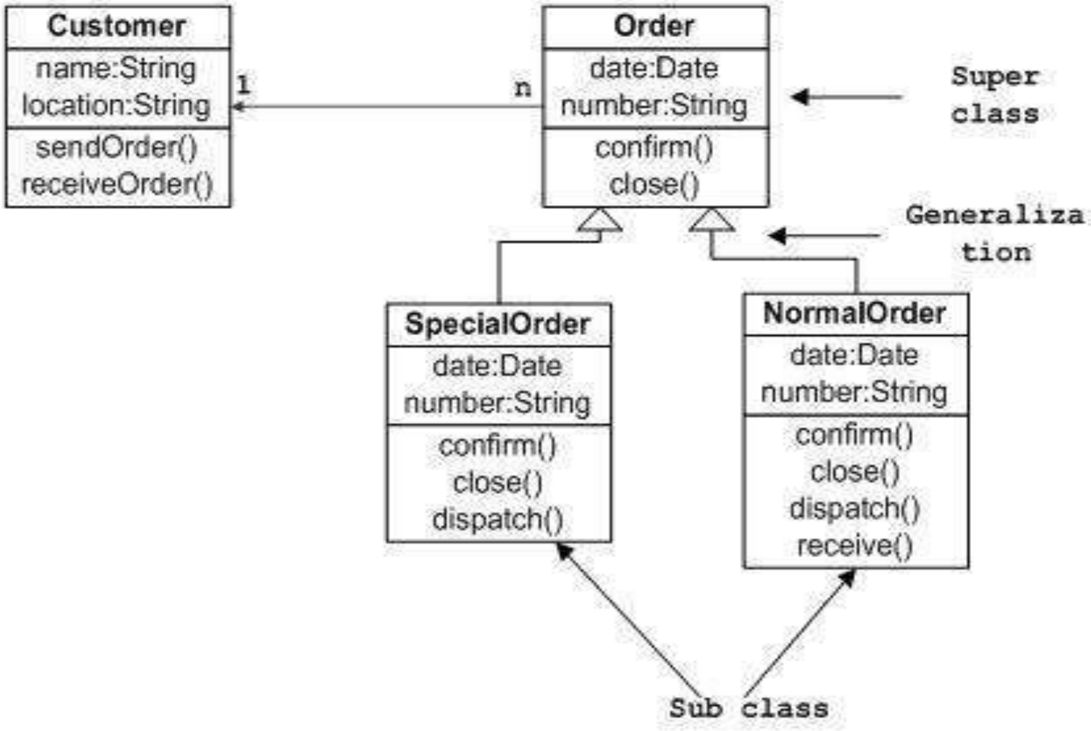
Class Diagram

- Generalization



Class Diagram

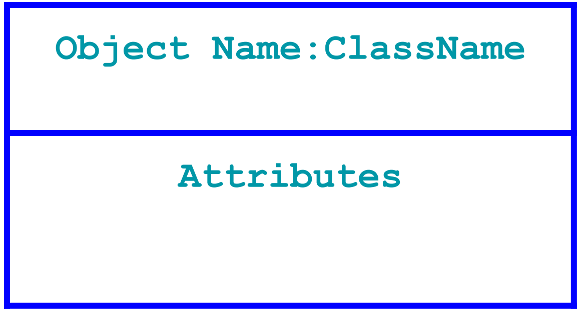
Sample Class Diagram



Object Diagram

- It shows instances instead of classes.
- They are useful for exploring small pieces with complicated relationships, especially recursive relationships.

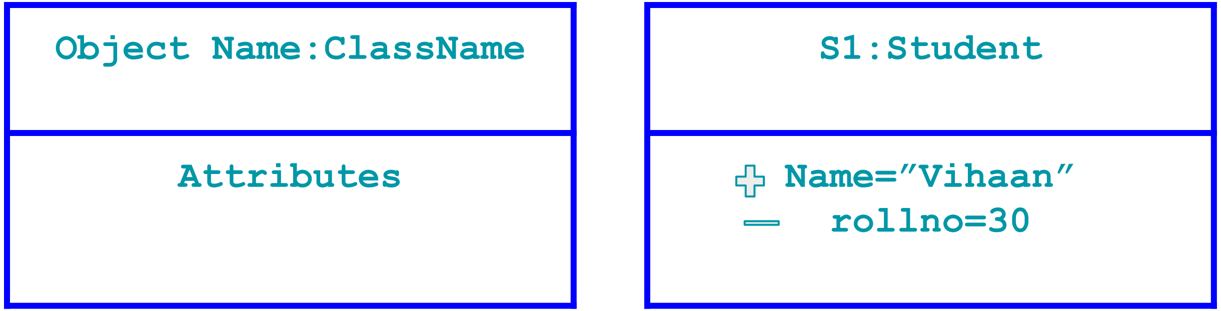
Object Diagram



Object Name:ClassName

Attributes

Object Diagram



Object Name:ClassName

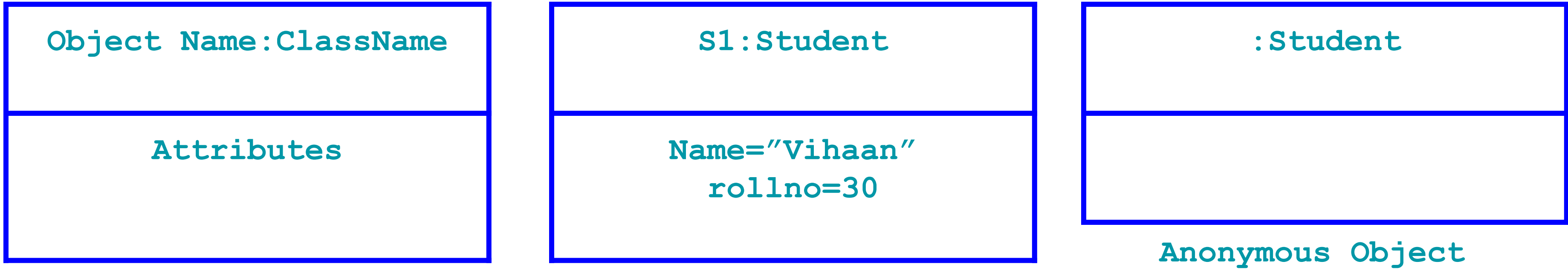
Attributes

S1:Student

+ Name=”Vihaan”

− rollno=30

Object Diagram



Object Name:ClassName

Attributes

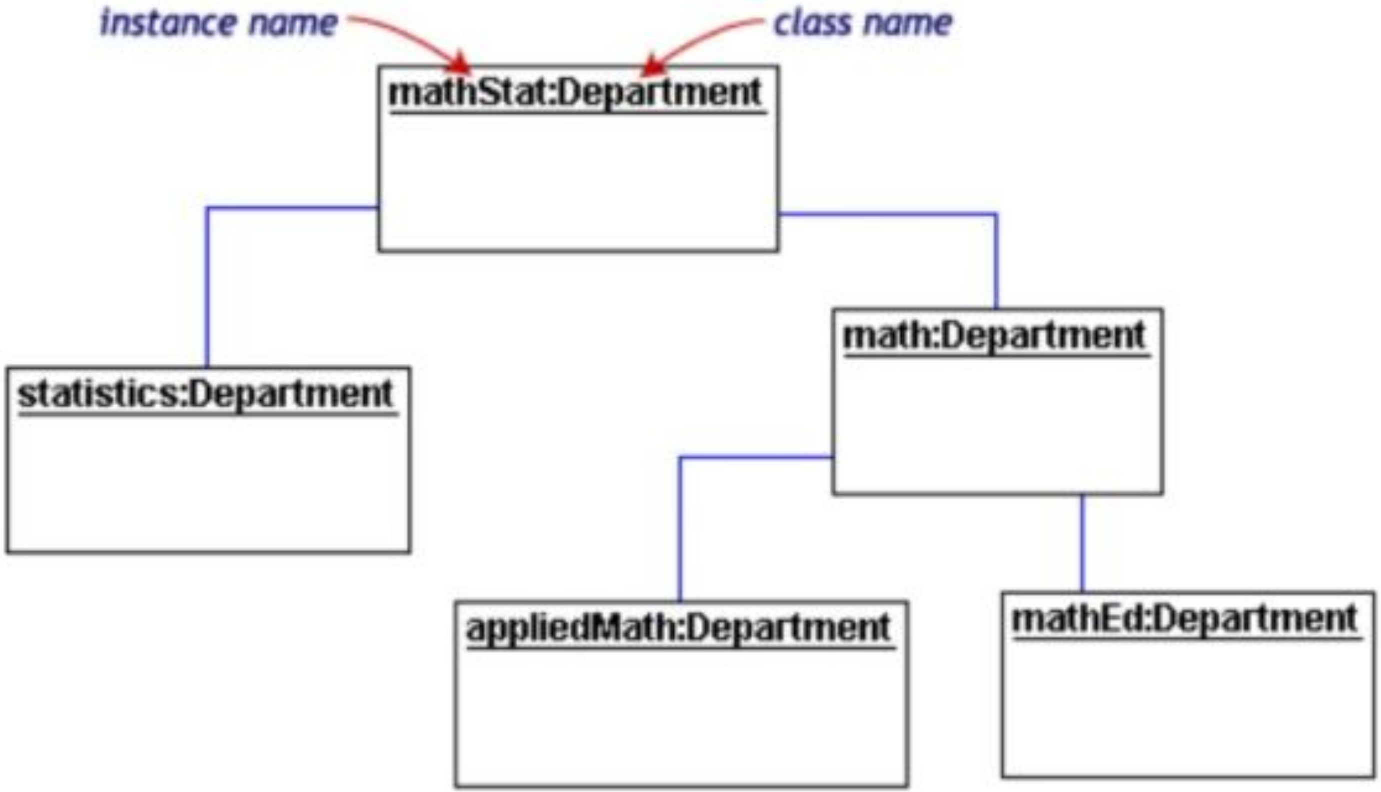
S1:Student

Name=”Vihaan”
rollno=30

:Student

Anonymous Object

Object Diagram


instance name
class name
mathStat:Department
statistics:Department
math:Department
appliedMath:Department
mathEd:Department