CMSC 132:
Object-Oriented Programming II

UML II

Department of Computer Science

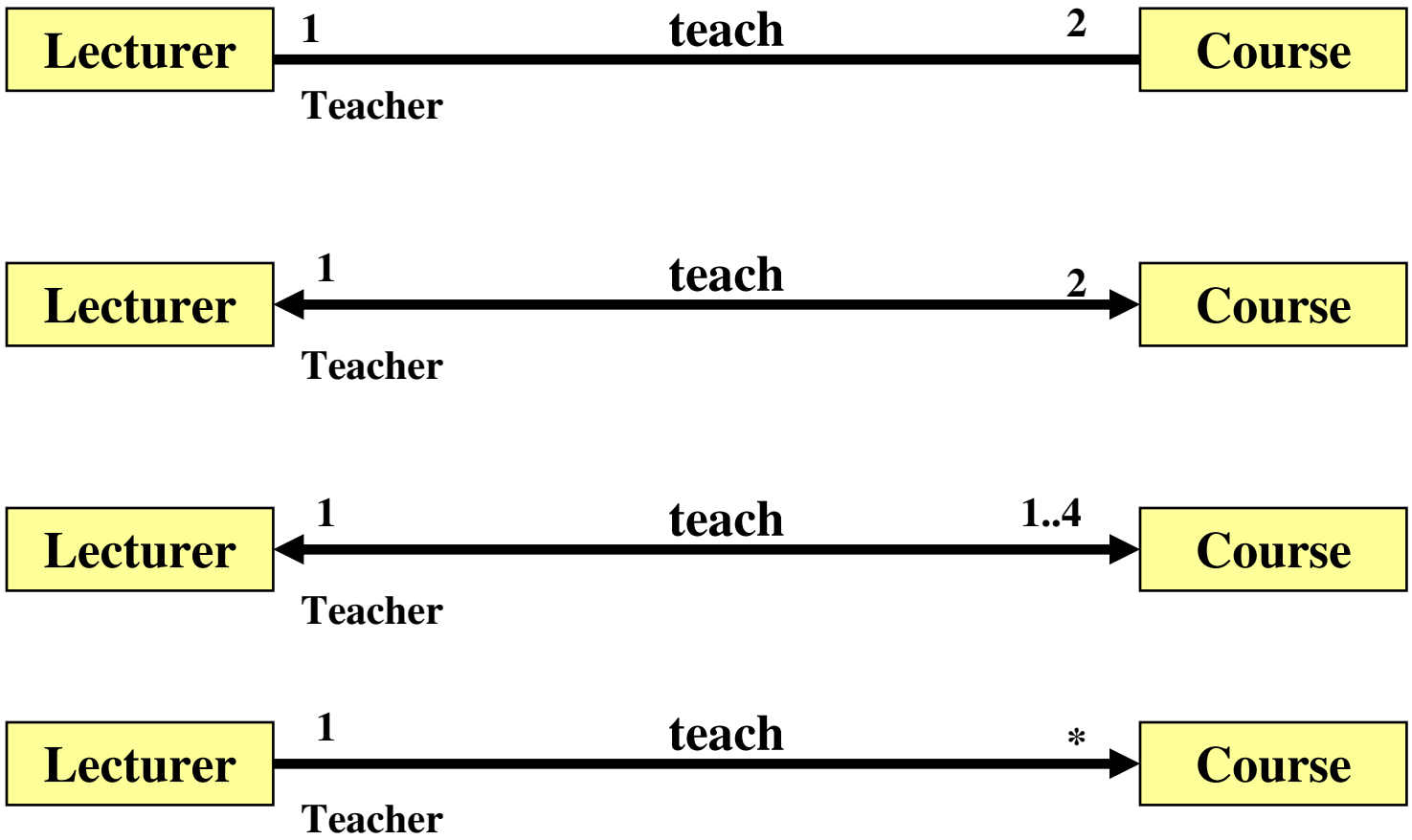
University of Maryland, College Park

Overview

- Use of UML Class Diagrams to represent relationships
  - Association
  - Generalization
  - Dependency
- UML Notation for inner (nested) classes
- UML Notation for Generic classes

Relationships Among Classes
Association

- UML Association Examples

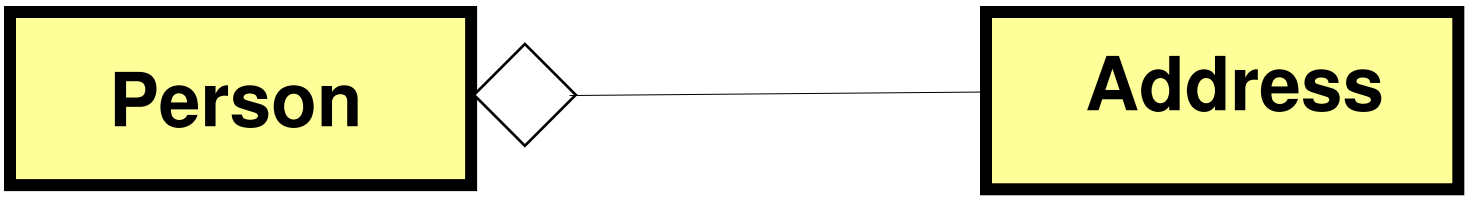


Relationships Among Classes
Association

- **Multiplicity Notation**
  - * ⇒ 0, 1, or more
  - 5 ⇒ 5 exactly
  - 5..8 ⇒ between 5 and 8, inclusive
  - 5..* ⇒ 5 or more

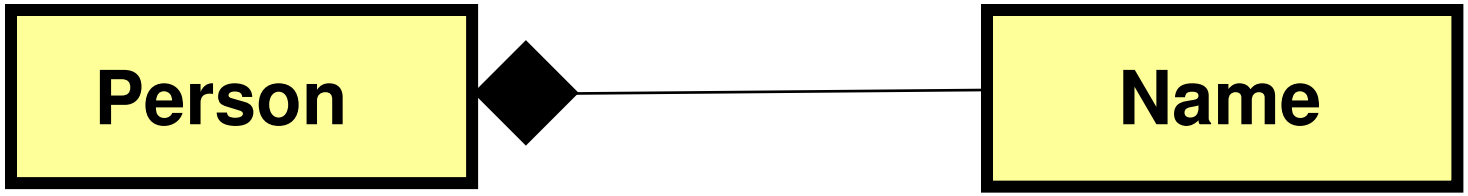
Relationships Among Classes
Association

- **Aggregation**
  - Special kind of association representing an ownership relationship between two classes where the owned object is not exclusively owned
  - Association is represented by a data field
  - Models a HAS-A relationship
  - Example: A person (Person class) has an address (Address class).
  - An open diamond is used to represent aggregation



Relationships Among Classes
Association

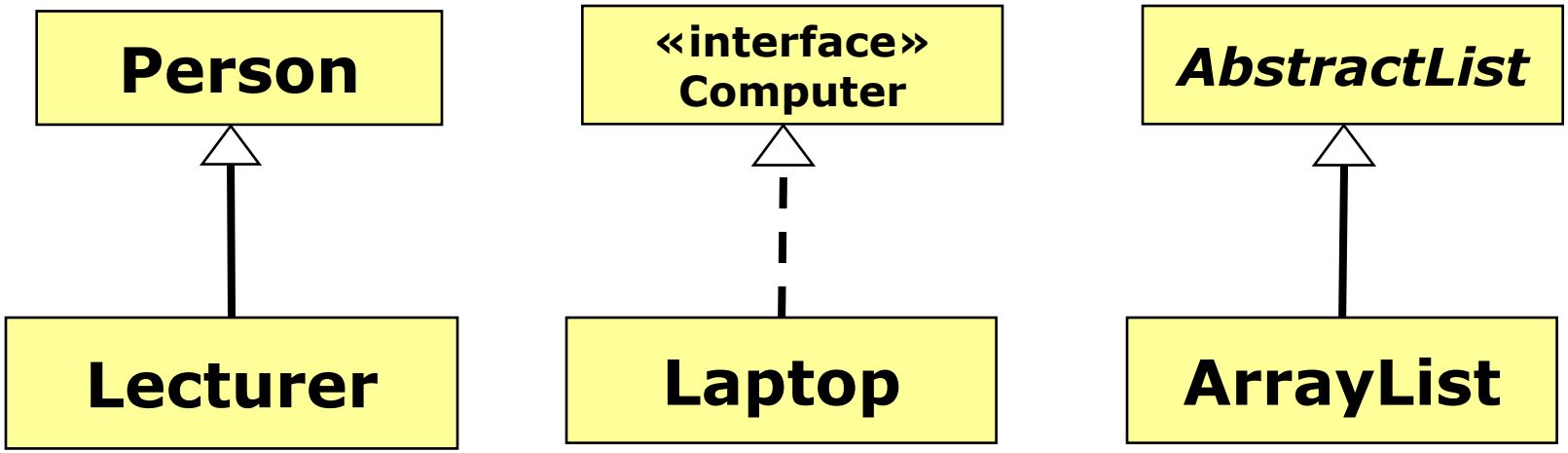
- **Composition**
  - Special kind of association representing an ownership relationship between two classes where the owned object is exclusively owned
  - Association is represented by a data field
  - Models a HAS-A relationship
  - Example: A person (Person class) has a name (Name class).
  - A filled diamond is used to represent composition



Relationships Among Classes
Generalization

- Describes a relationship between a superclass and its subclasses
- Models the IS-A relationship
- Example: A lecturer is a person (Lecturer extends a Person class)
- UML Notation
  - Solid line with open arrowhead pointing to superclass represents a class extending another class
  - Dashed line with open arrowhead represents a class implementing an interface

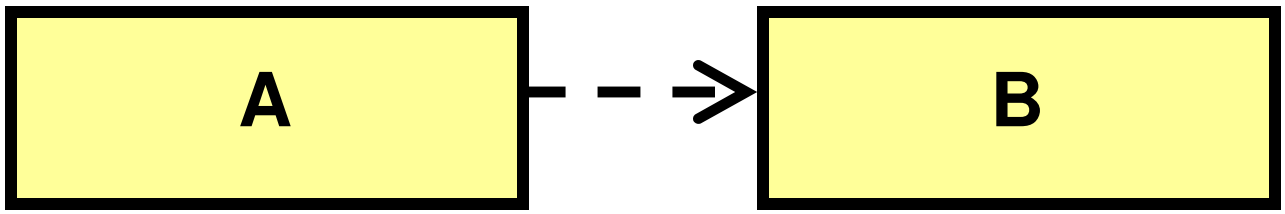
Relationships Among Classes
Generalization



- **Leftmost diagram – Lecturer extends Person**
- **Center diagram – Laptop implements the Computer interface (notice the dashed arrow lines and the guillemets)**
- **Rightmost diagram – ArrayList extends the abstract class AbstractList (notice italics indicate abstract classes). An alternative notation is to use { } (e.g., {AbstractList})**

Relationships Among Classes
Dependency

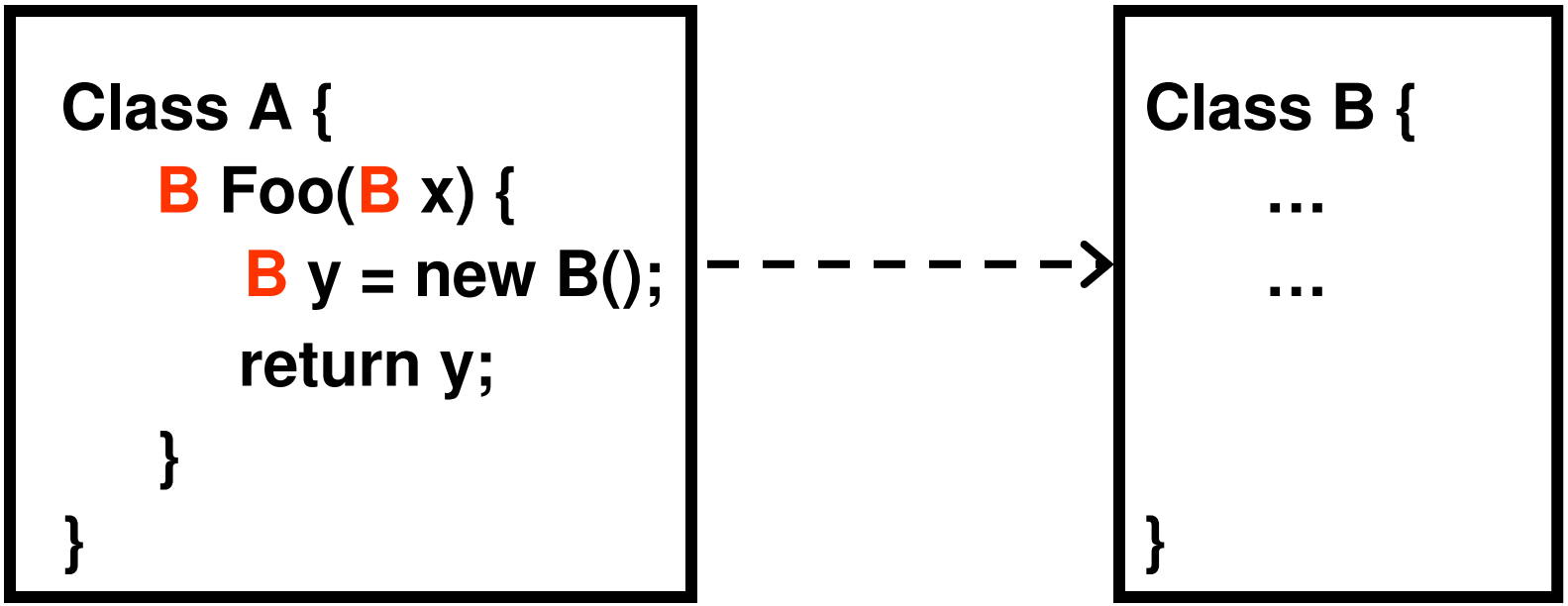
- **Dependency – When a class A uses a class B in some way other than generalization or association.**
- **Example: If class A has a method that creates an instance of class B, then class A depends on class B**
- **Always directed (Class A depends on B)**
- **UML Notation - dashed line with arrowhead**



A depends on B

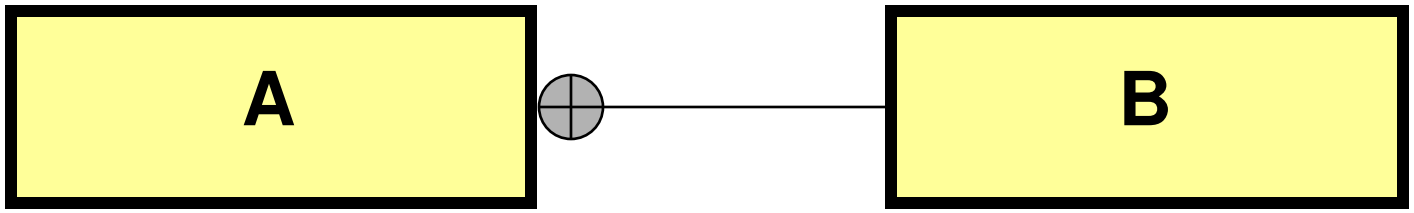
Relationships Among Classes
Dependency

- Dependency caused by
  - Local variable
  - Parameter
  - Return value
- Example



UML Notation for Inner (Nested) Classes

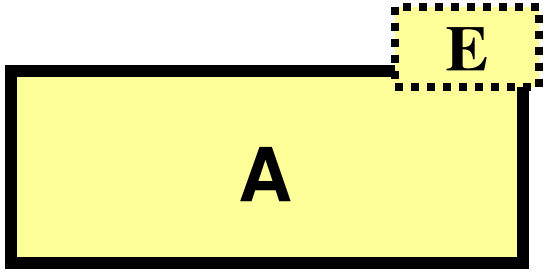
- Inner class – class declared within the body of another class
- UML Notation



B is an inner class of A

UML Notation for Generic Classes

- Generic parameter(s) will be placed in the upper right corner of the rectangle modeling the class
- Example



Example

Let's go over an example …