OBJECT ORIENTED TECHNIQUES USING JAVA(ACSE0302)

Unit: 1

Introduction

Course Details
(B.Tech 3rd Sem /2nd Year)

Mr. Surya Prakash Sharma
Assistant Professor
CSE Department

Evaluation Scheme

Sl. No.	Subject Codes	Subject Name	Periods			Evaluation Scheme				End Semester		Total	Credit
			L	T	P	CT	TA	TOTAL	PS	TE	PE		
WEEKS COMPULSORY INDUCTION PROGRAM													
1	AAS0301A	Engineering Mathematics-III	3	1	0	30	20	50		100		150	4
2	ACSE0306	Discrete Structures	3	0	0	30	20	50		100		150	3
3	ACSE0304	Digital Logic & Circuit Design	3	0	0	30	20	50		100		150	3
4	ACSE0301	Data Structures	3	1	0	30	20	50		100		150	4
5	ACSE0302	Object Oriented Techniques using Java	3	0	0	30	20	50		100		150	3
6	ACSE0305	Computer Organization & Architecture	3	0	0	30	20	50		100		150	3
7	ACSE0354	Digital Logic & Circuit Design Lab	0	0	2				25		25	50	1
8	ACSE0351	Data Structures Lab	0	0	2				25		25	50	1
9	ACSE0352	Object Oriented Techniques using Java Lab	0	0	2				25		25	50	1
10	ACSE0359	Internship Assessment-I	0	0	2				50			50	1
11	ANC0301/ ANC0302	Cyber Security*/ Environmental Science*(Non Credit)	2	0	0	30	20	50		50		100	0
12		MOOCs** (For B.Tech. Hons. Degree)											
		GRAND TOTAL										**1100**	**24**

Syllabus

UNIT-I	Introduction	8 Hours

Object Oriented Programming: Introduction and Features: Abstraction, Encapsulation, Polymorphism, and Inheritance.

Modeling Concepts: Introduction,Class Diagram and Object Diagram.

Control Statements: Decision Making, Looping and Branching, Argument Passing Mechanism: Command Line Argument.

UNIT-II	Basics of Java Programming	8 Hours

Class and Object: Object Reference, Constructor, Abstract Class, Interface and its uses, Defining Methods, Use of "this" and "super" keyword, Garbage Collection and finalize () Method.

Inheritance: Introduction and Types of Inheritance in Java, Constructors in Inheritance.

Polymorphism: Introduction and Types, Overloading and Overriding.

Lambda expression: Introduction and Working with Lambda Variables.

Arrays: Introduction and its Types.

Syllabus

UNIT-III	Packages, Exception Handling and String Handling	8 Hours

Packages: Introduction and Types, Access Protection in Packages, Import and Execution of Packages.

Exception Handling, Assertions and Localizations: Introduction and Types, Exceptions vs. Errors, Handling of Exception. Finally, Throws and Throw keyword, Multiple Catch Block, Nested Try and Finally Block, Tokenizer. Assertions and Localizations Concepts and its working.

String Handling: Introduction and Types, Operations, Immutable String, Method of String class, String Buffer and String Builder class.

UNIT-IV	Concurrency in Java and I/O Stream	8 Hours

Threads: Introduction and Types, Creating Threads, Thread Life-Cycle, Thread Priorities, Daemon Thread, Runnable Class, Synchronizing Threads.

I/O Stream: Introduction and Types, Common I/O Stream Operations, Interaction with I/O Streams Classes.

Annotations: Introduction, Custom Annotations and Applying Annotations.

UNIT-V	GUI Programming, Generics and Collections	8 Hours

GUI Programming: Introduction and Types, Swing, AWT, Components and Containers, Layout Managers and User-Defined Layout and Event Handling.

Generics and Collections: Introduction, Using Method References, Using Wrapper Class, Using Lists, Sets, Maps and Queues, Working with Generics.

Text Books

Text Books:
(1) Herbert Schildt," Java - The Complete Reference", McGraw Hill Education 12th edition
(2) James Rumbaugh et. al, "Object Oriented Modeling and Design", PHI 2nd Edition
Reference Books:
(3) E Balagurusamy, "Programming with Java A Primer", TMH, 4th edition.

Branch-wise Applications

Java can be used :

- Data import and export.
- Cleaning data.
- Statistical analysis.
- Machine learning and Deep learning.
- Text analytics (also known as Natural Language Processing or NLP).
- Data visualization.

Course Objectives

- To understand the object-oriented methodology and its techniques to design and develop conceptual models and demonstrate the standard concepts of object-oriented techniques.

- To understand the fundamental concepts of object-oriented programming in Java language and also implement the Multithreading concepts, GUI based application and collection framework.

Course Outcomes

After completion of this course students will be able to:

CO1	Identify the concepts of object oriented programming and relationships among them needed in modeling.
CO2	Demonstrate the Java programs using OOP principles and also implement the concepts of lambda expressions.
CO3	Implement packages with different protection level and evaluate the error handling concepts for uninterrupted execution of Java program.
CO4	Implement Concurrency control, I/O Streams and Annotations concepts by using Java program.
CO5	Design and develop the GUI based application, Generics and Collections in Java programming language to solve the real-world problem.

Result Analysis

RESULT

SECOND YEAR ODD SEMESTER SESSION 2020-2021(3RD SEM)

sr.number	subject code	subject name	faculty name	pass perrcentage	average marks
1	AAS0301A	Engineering Mathematics-III	MR.RAMAN CHAUHAN/DR KANCHAN THYAGI	100	78.61
2	ACSE0301	Data Structures	DR AMBA / MR NARENDRA / MS NEHA YADAV	90	44.2
3	ACSE0302	Object Oriented Techniques using Java	MR GAURAV KUMAR/MS NANCY KANSAL	100	62.26
4	ACSE0304	Digital Logic & Circuit Design	MS KANIKA TANEJA / MS MD SAJED	100	86.58
5	ACSE0305	Computer Organization & Architecture	DR VIVEK KUMAR/MS SANCHALI	100	81.87
6	ACSE0306	Discrete Structures	MS SHRUTI SINHA / MR JAYCHAND /	94	52.77
8	ACSE0351	Data Structures Lab	DR AMBA / MR NARENDRA / MS NEHA YADAV	100	23.47
9	ACSE0352	Object Oriented Techniques using Java	MR GAURAV KUMAR/MS NANCY KANSAL	100	23.94
10	ACSE0354	Digital Logic & Circuit Design Lab	MS KANIKA TANEJA / MS MD SAJED	100	22.76
11	ACSE0359	Internship Assessment-I	all faculty	100	44.85
12	ANC0301	Cyber Security*	MS HARSHA GUPTA	100	36.4

Prerequisite/Recap

- Students must have basic understanding of computer programming and related programming paradigms

Unit Contents

- Object Oriented Programming
- Introduction and Features
- Abstraction
- Encapsulation
- Polymorphism
- Inheritance
- Modeling Concepts
- Class Diagram
- Object Diagram.
- Control Statements
- Decision Making
- Looping
- Branching
- Argument Passing Mechanism
- Command Line Argument.

Lecture 1

- Object Oriented Programming
- Introduction and Features
- Procedural Vs Object Oriented Programming

Object Oriented Programming

Why Do We Need Object-Oriented Programming?

- Object-Oriented Programming was developed because limitations were discovered in earlier approaches to programming.
- To appreciate what OOP does, we need to understand what these limitations are and how they arose from traditional programming languages.

Object Oriented Programming

Procedure-Oriented Programming

- C, Pascal, FORTRAN, and similar languages are procedural languages.
- Each statement in the language tells the computer to do something:

– get some input

– add these numbers

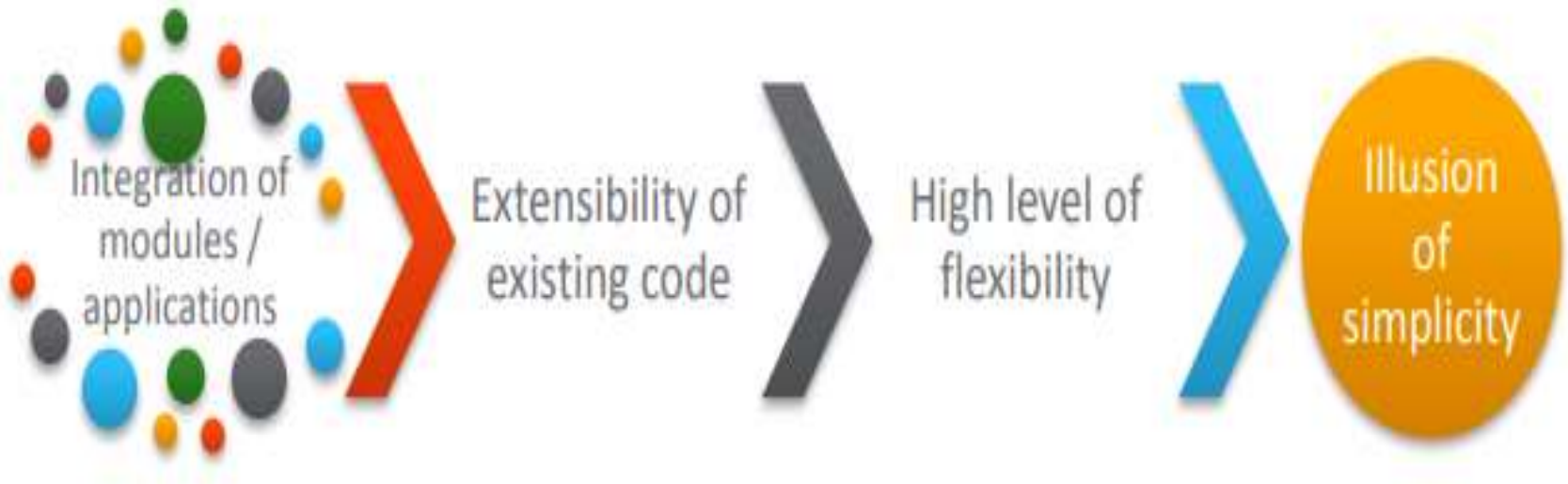
– divide by 6

– display that output

- A program in a procedural language is a list of instructions.

Need for OOP Approach

- **Challenges in developing a business application**.



- If these challenges are not addressed, it may lead to software crisis.

Procedure-Oriented Programming

- Procedural program is divided into functions

•Each function has a clearly defined purpose and a clearly defined interface to the other functions in the program.

•The idea of breaking a program into functions can be further extended by grouping a number of functions together into a larger entity called a module.

Procedure Oriented Programming (Cont.)

Procedure-Oriented Programming Drawbacks:

- Since every function has complete access to the global variables, the new programmer can corrupt the data accidentally by creating function.
- We can access the data of one function from other since, there is no protection.
- In large program it is very difficult to identify what data is used by which function.
- Similarly, if new data is to be added, all the function needed to be modified to access the data.

Procedure Oriented Programming Vs Object Oriented Programming

Procedural Oriented Programming	Object Oriented Programming
Program is divided into small parts called ***functions***.	Program is divided into small parts called ***objects***.
It is follows ***top down approach***.	It is follows ***bottom up approach***.
Access specifier is absent	It has access specifiers like **private, public, protected** etc.
Adding new data and function is not easy.	Adding new data and function is easy.
It does not have any proper way for hiding data so it is ***less secure***.	It provides data hiding so it is ***more secure***.
Overloading is not possible.	Overloading is possible.
Function is more important than data.	Data is more important than function.
Based on ***unreal world***.	Based on ***real world***.
Examples: C, FORTRAN, Pascal, Basic etc.	Examples: C++, Java, Python, C# etc.

Lecture 2

- Abstraction
- Encapsulation
- Polymorphism
- Inheritance

Object Oriented Programming

- In the object oriented paradigm, the attributes and the associated operations are treated as one entity known as an object.
- Data of an object can be accessed only by the functions associated with that object.
- Communication of the objects done through function.

Object Oriented Programming

Object means a real-world entity such as a pen, chair, table, computer, watch, etc.

Object-Oriented Programming is a methodology or paradigm to design a program using classes and objects.

It simplifies software development and maintenance by providing some concepts:

- Object
- Class
- Abstraction
- Encapsulation
- Inheritance
- Polymorphism

Class

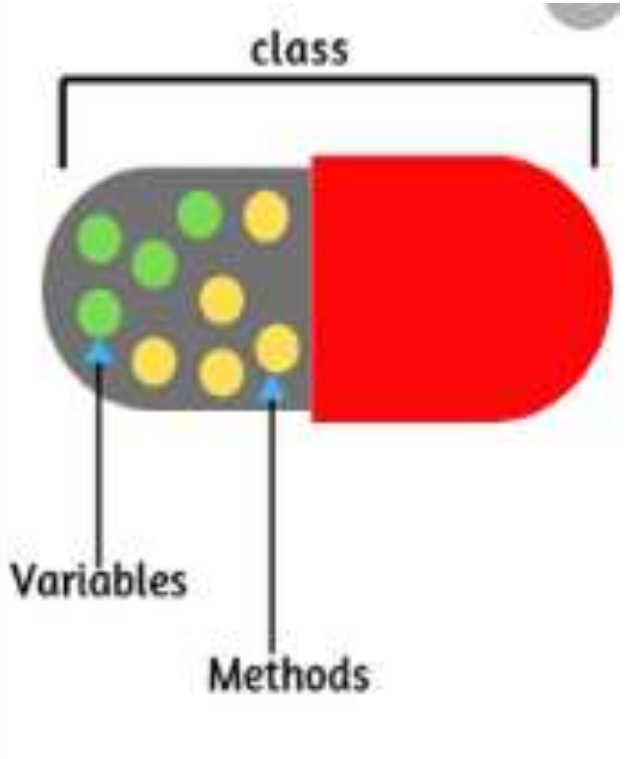
Class: A class is a container of similar objects with same data structure and behavior. A class has infinite number of similar of objects and a object is a instance of a class.

EX. Animal:[class] and Dog , Cat:[objects]

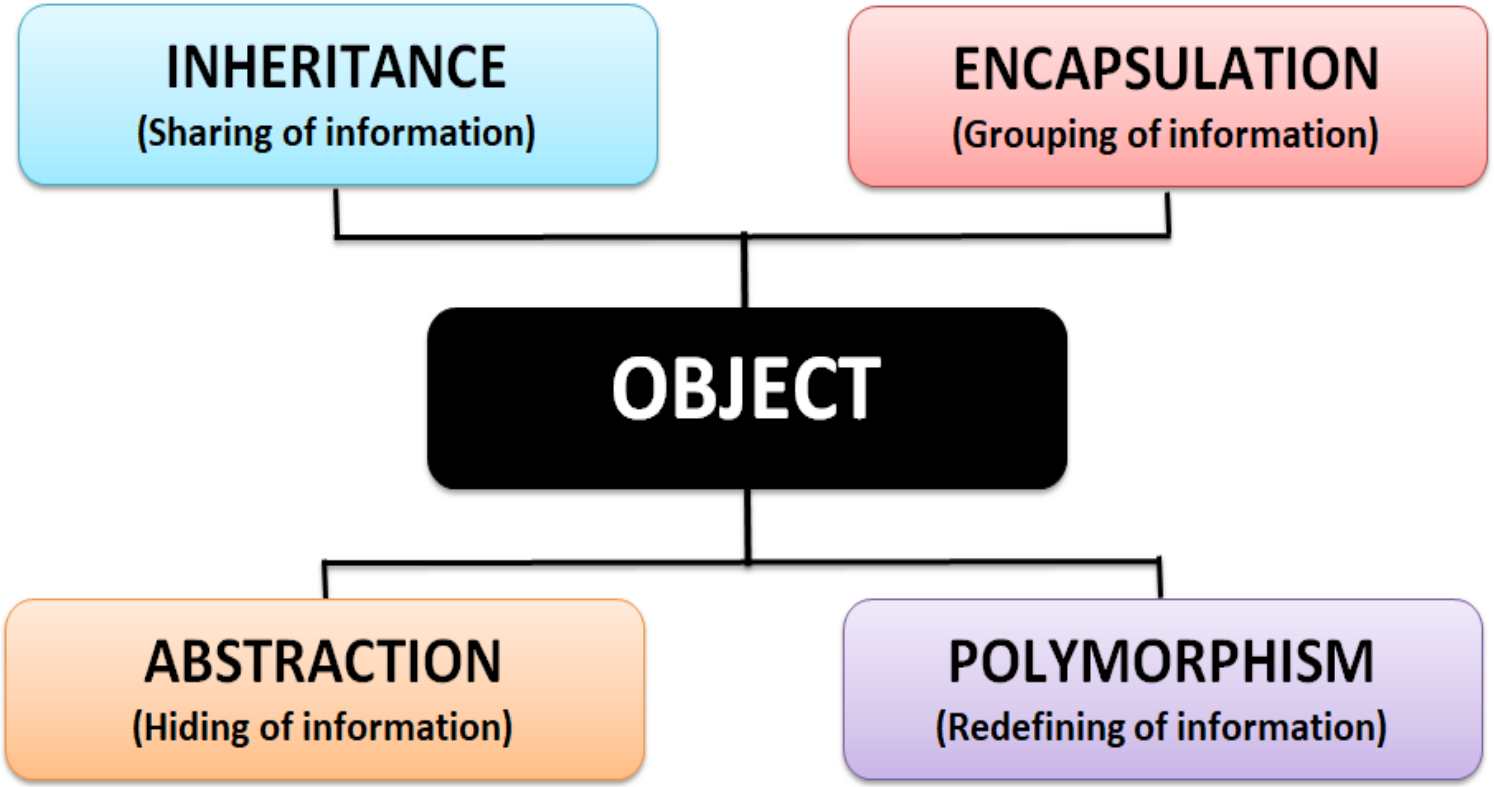
```
class
{

  data members
        +
  methods (behavior)

}
```

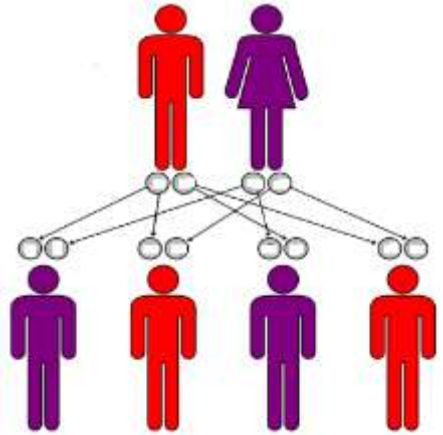


Pillars of OOPs



Inheritance

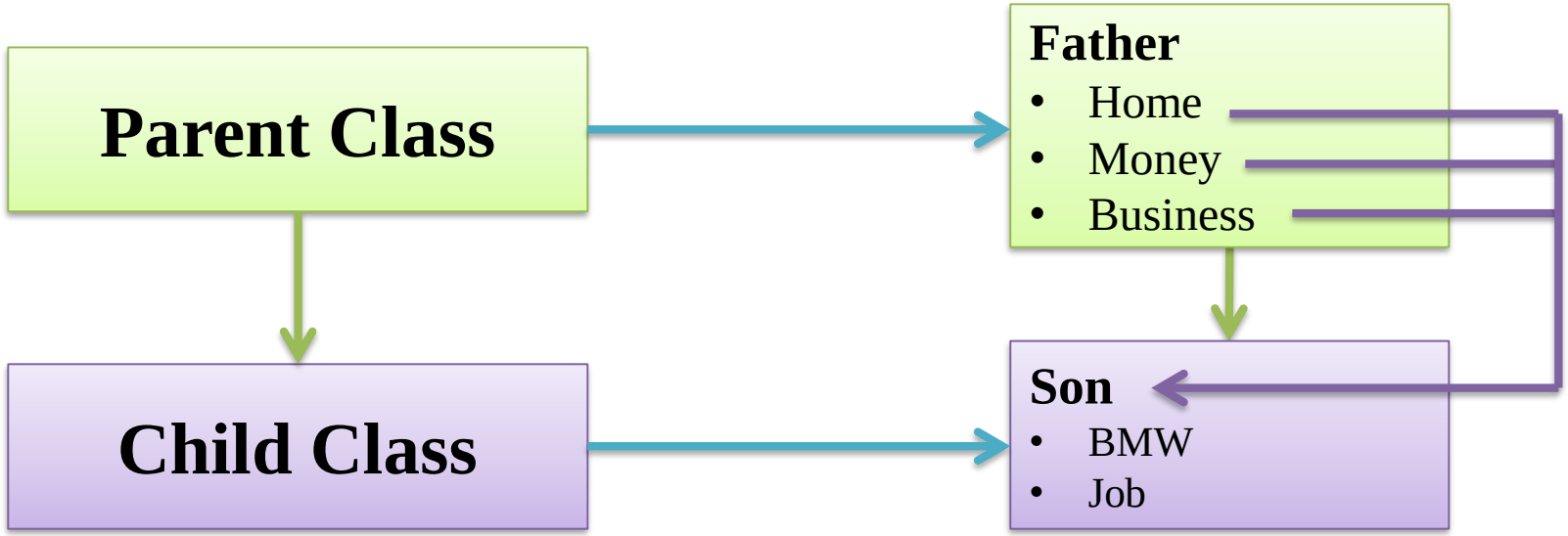
- The mechanism of deriving a new class from an old one (existing class) such that the new class inherit all the members (variables and methods) of old class is called inheritance or derivation.
- A class that is derived from another class is called a **subclass** (also a derived class, extended class or child class)
- The class from which the subclass is derived is called a **super class** or **base class** or **parent class**



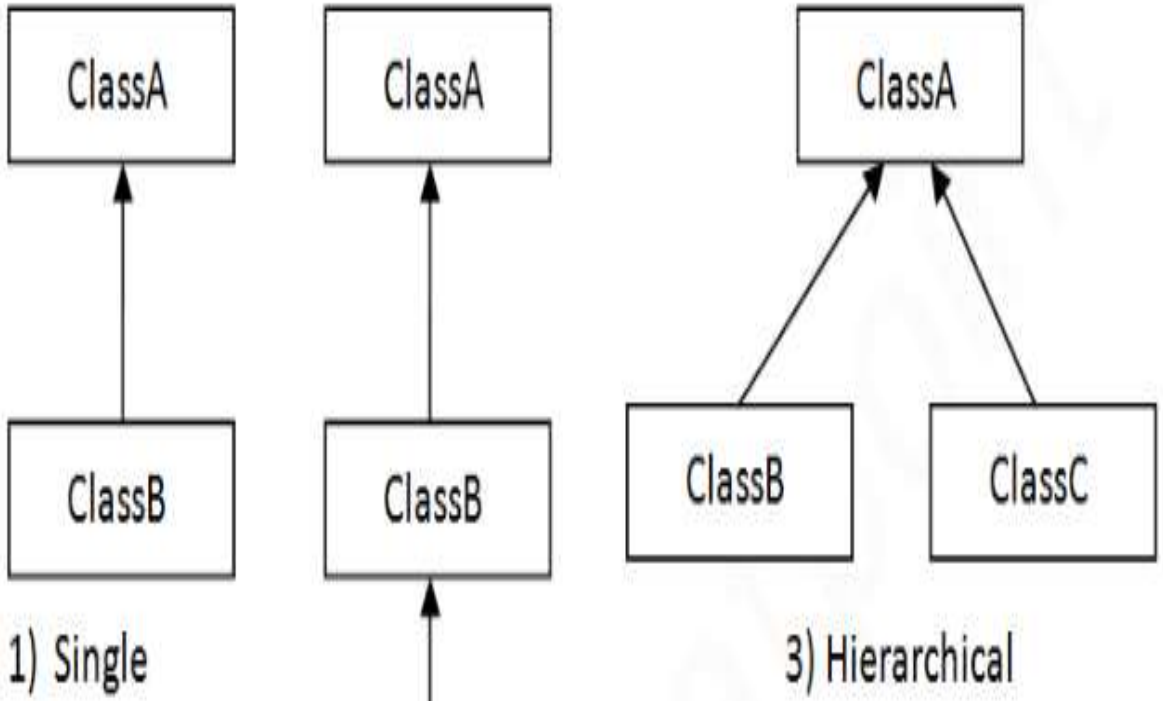
Inheritance

The old class is referred to as the Super class and the new one is called the Sub class.

- Parent Class - Base Class or Super Class
- Child Class - Derived Class or Sub Class



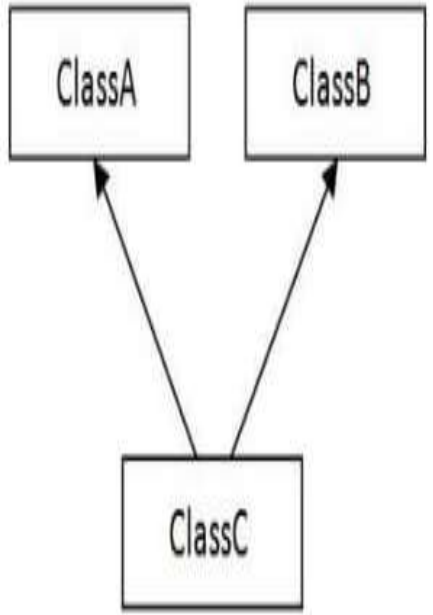
Types of Inheritance



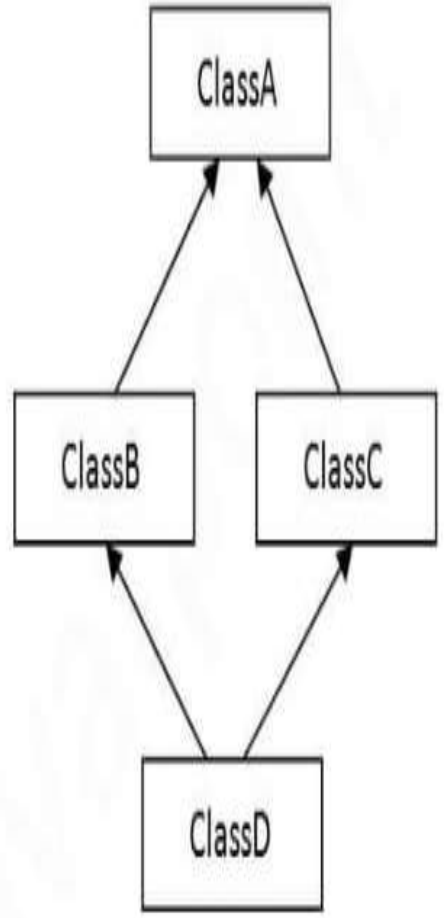
1) Single

2) Multilevel

3) Hierarchical



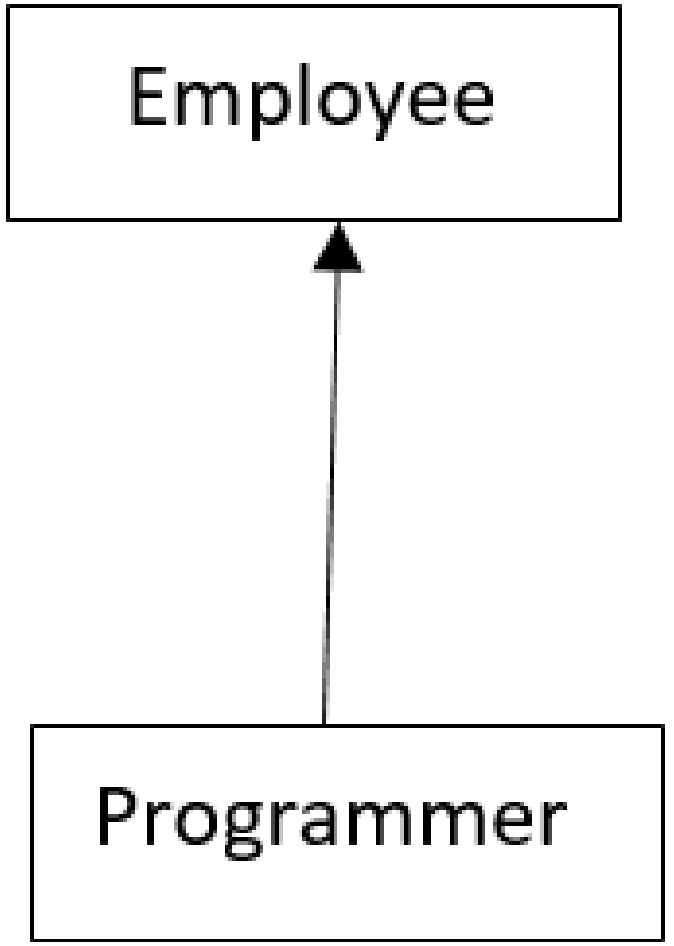
4) Multiple



5) Hybrid

Inheritance Example

```
1. class Employee{
2.    float salary=40000;
3. }
4. class Programmer extends Employee{
5.   int bonus=10000;
6.   public static void main(String args[]){
7.     Programmer p=new Programmer();
8.     System.out.println("Programmer salary is:"+p.salary);
9.     System.out.println("Bonus of Programmer is:"+p.bonus);
10.}
11.}
```



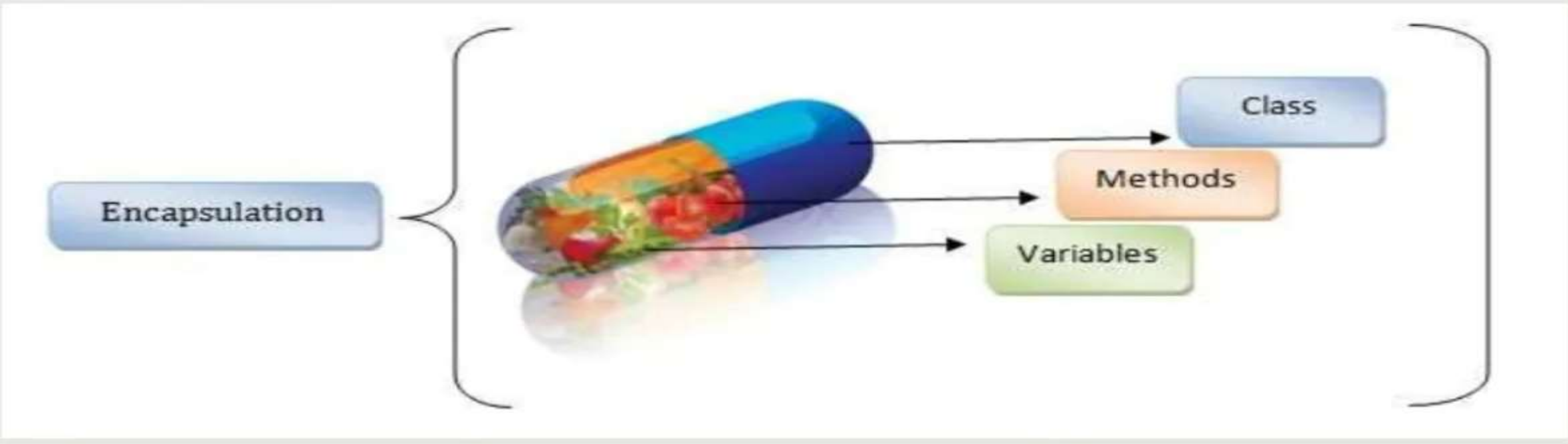
Encapsulation

- Binding the data with the code that manipulates it.
- It keeps the data and the code safe from external interference
- Class provides the encapsulation in OOP.

Example :-

- School bag is one of the most real examples of Encapsulation. School bag can keep our books, pens, etc.

Encapsulation



Abstraction

- Abstraction is a process of hiding the implementation details from the user, only the functionality will be provided to the user. In other words, the user will have the information on what the object does instead of how it does.

- **Example 1**, when you login to your Amazon account online, you enter your user_id and password and press login, what happens when you press login, how the input data sent to amazon server, how it gets verified is all abstracted away from the you.

Abstraction

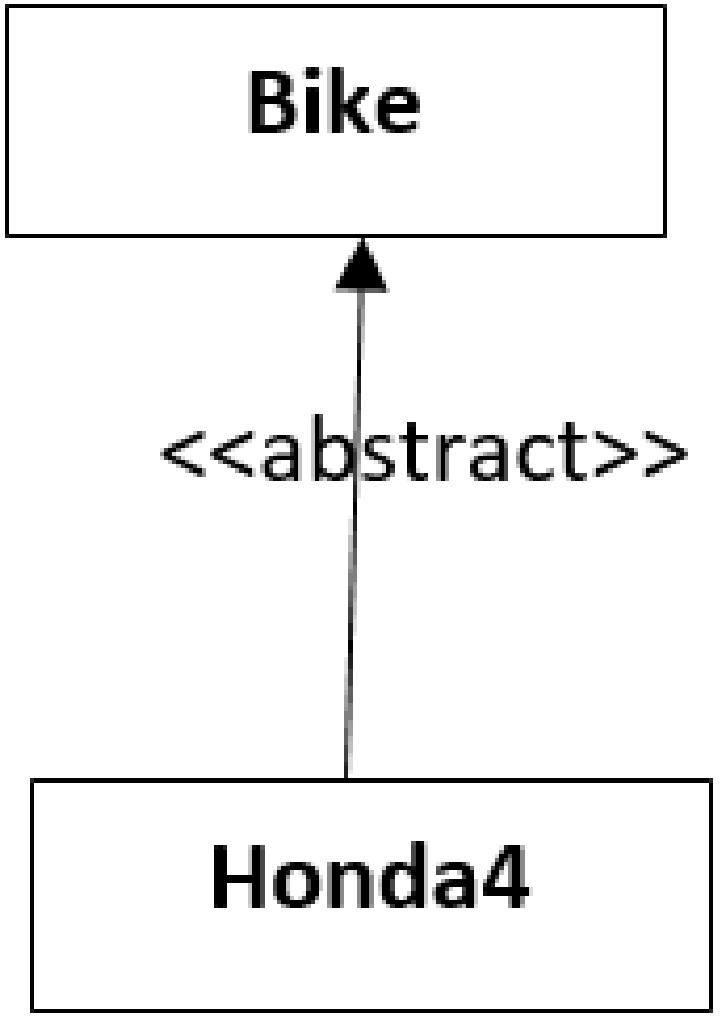
Rules for Java Abstract class

1. An abstract class must be declared with an abstract keyword.
2. It can have abstract and non-abstract methods.
3. It cannot be Instantiated.
4. It can have final methods
5. It can have constructors and static methods also.

Abstraction

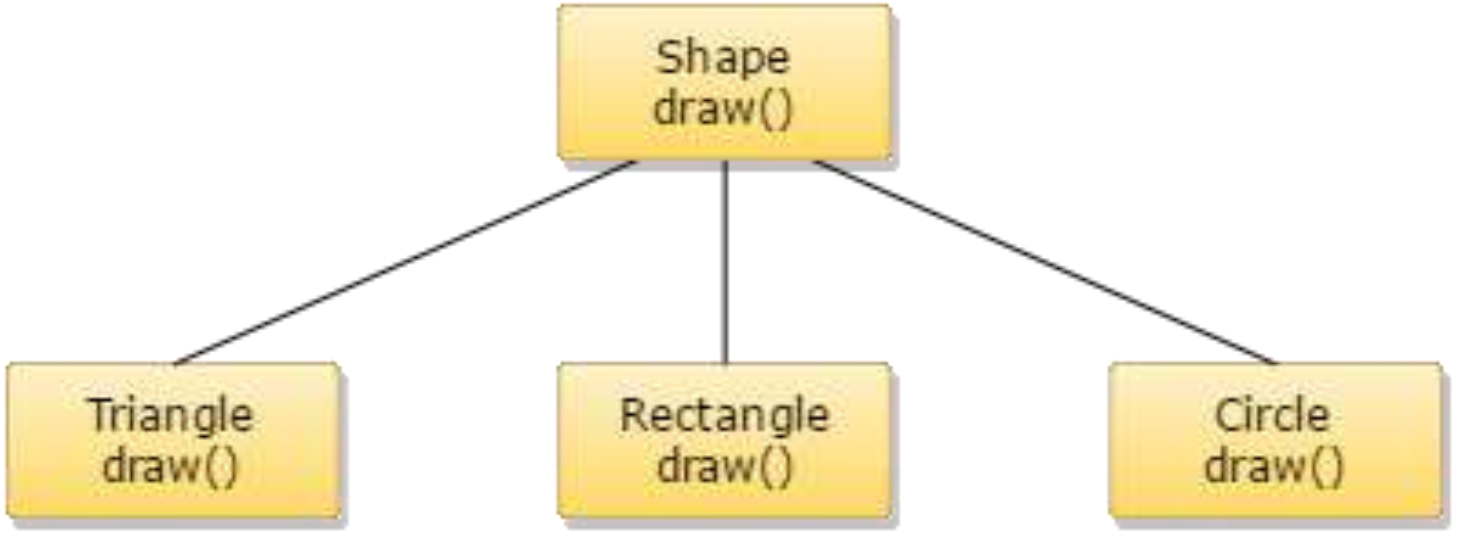
```
abstract class Bike{

        abstract void run();
}
class Honda4 extends Bike{
                void run(){
                  System.out.println("running safely");
                        }
                  public static void main(String args[]){
                    Bike obj = new Honda4();
                   obj.run();
                     }
        }
```



Polymorphism

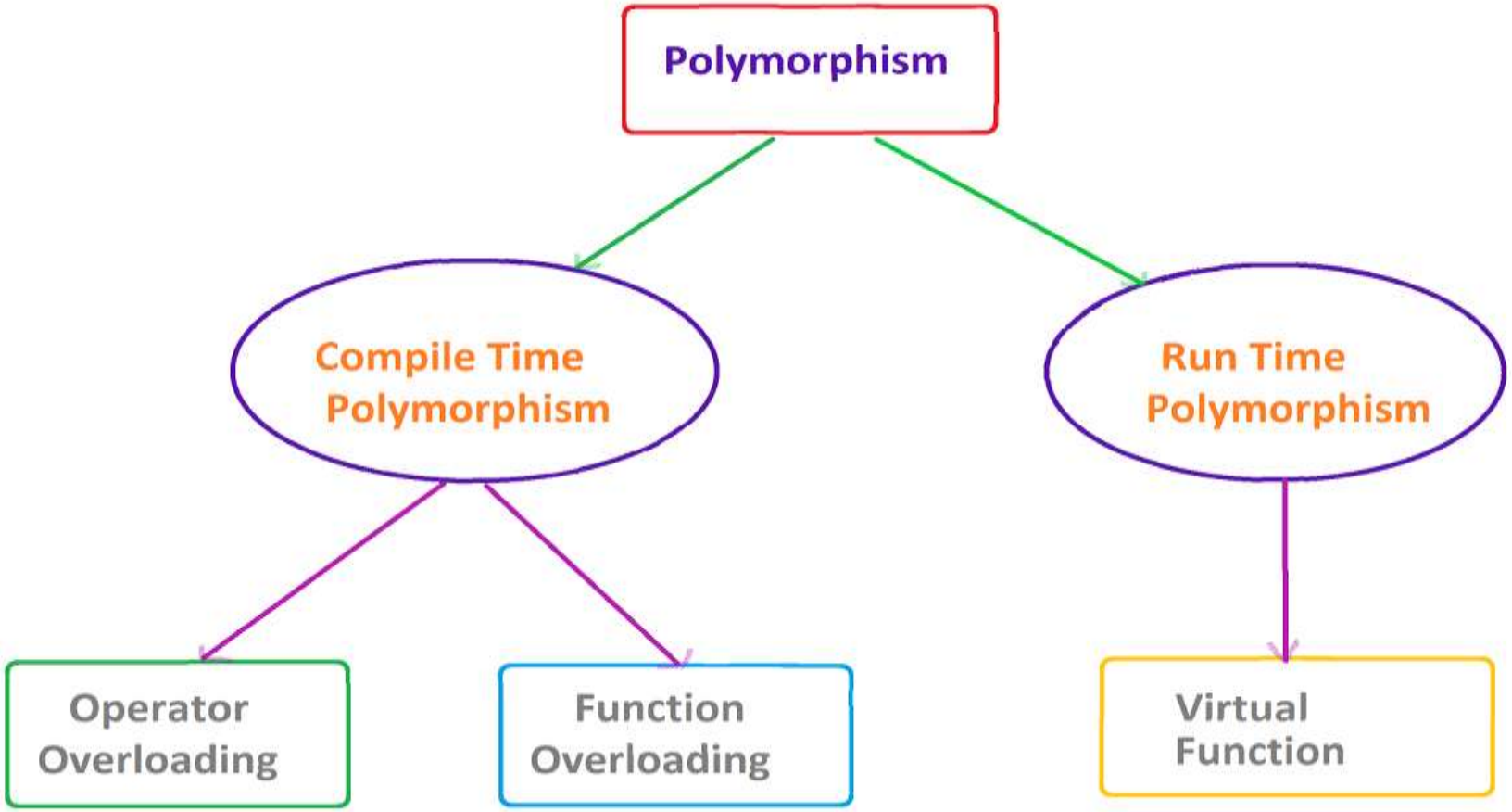
Polymorphism:(poly : many, morphs : form**):** same operation are defined differently for the different classes. EX. draw(operation).



Polymorphism

Type of Polymorphism

- It is the ability to take many form and refers to an operation exhibiting different behavior in different situations.



Lecture 3

- Modeling Concepts

Modeling Concepts

Modeling consists of building an abstraction of reality

Importance of Modeling

•Models help us

- to visualize a system as it is or as we want it to be.
- to specify the structure or behavior of a system.
- in providing a template that guides us in constructing a system.
- in providing documenting the decisions we have made.

Modeling Concepts

What is a Model

•A model is a simplification of reality.

•A model may provide blueprints of a system

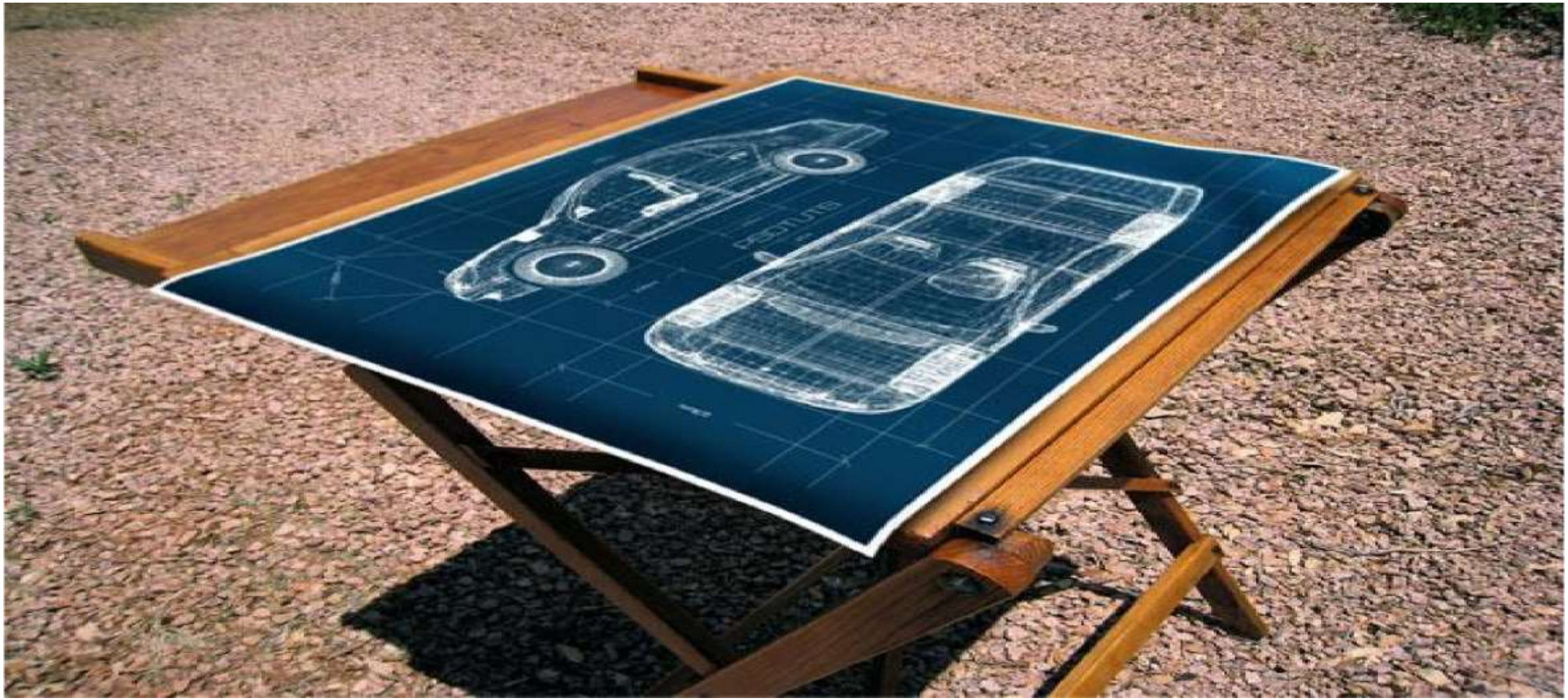
•**A model is an abstraction, before building any system a prototype may be developed**. The main purpose of model is for understanding of the system.

•Designer build different kinds of models for various purposes before constructing things.

•For example car , airplane, blueprints of machine parts, Plan for house construction etc., Models serve many purposes

Modeling Concepts

Modeling Concepts



Modeling Concepts: UML

UML

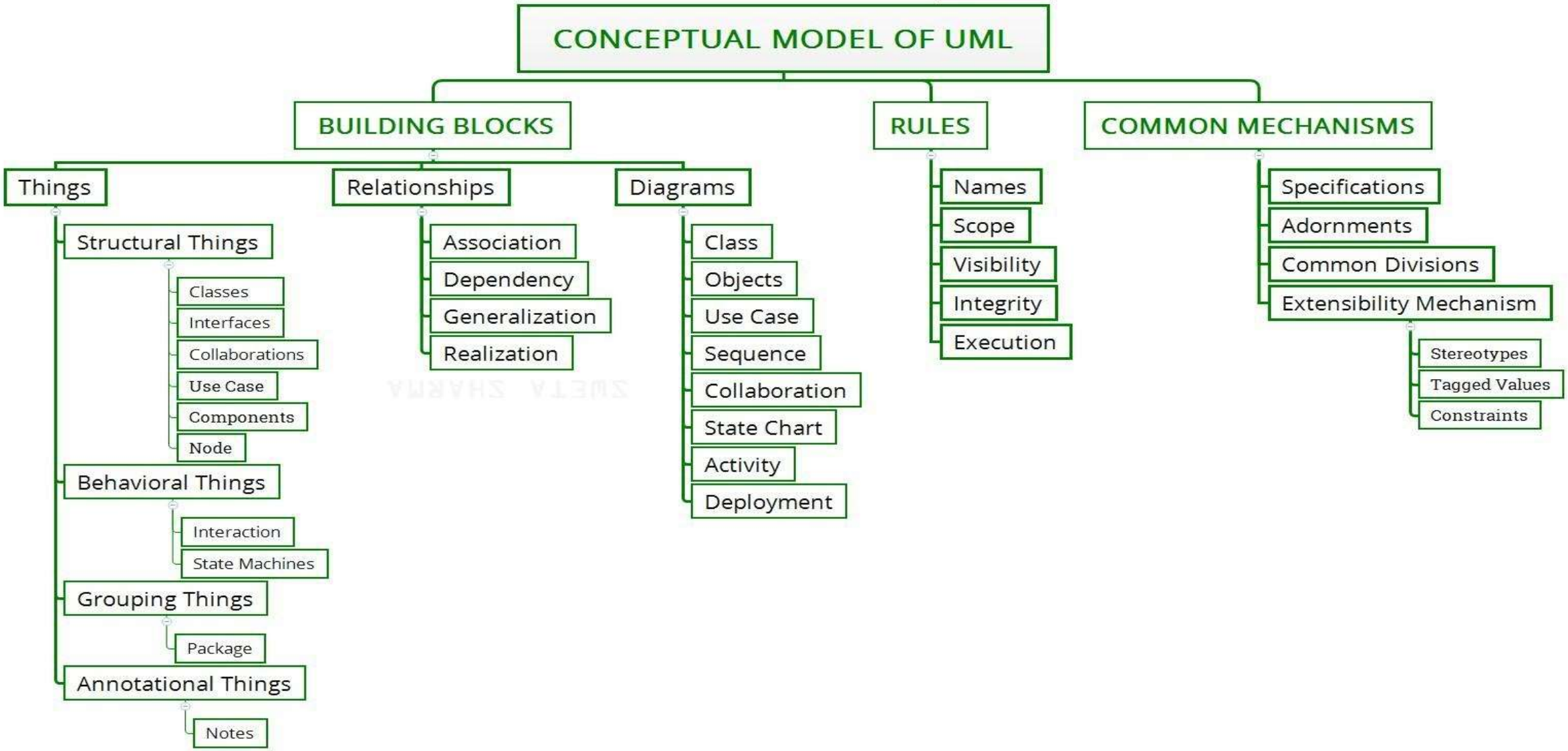
UML (Unified Modeling Language) is a general-purpose, graphical modeling language in the field of Software Engineering.

UML is used to specify, visualize, construct, and document the artifacts (major elements) of the software system.

UML is composed of three main building blocks,

1. Things
2. Relationships
3. Diagrams

Modeling Concepts



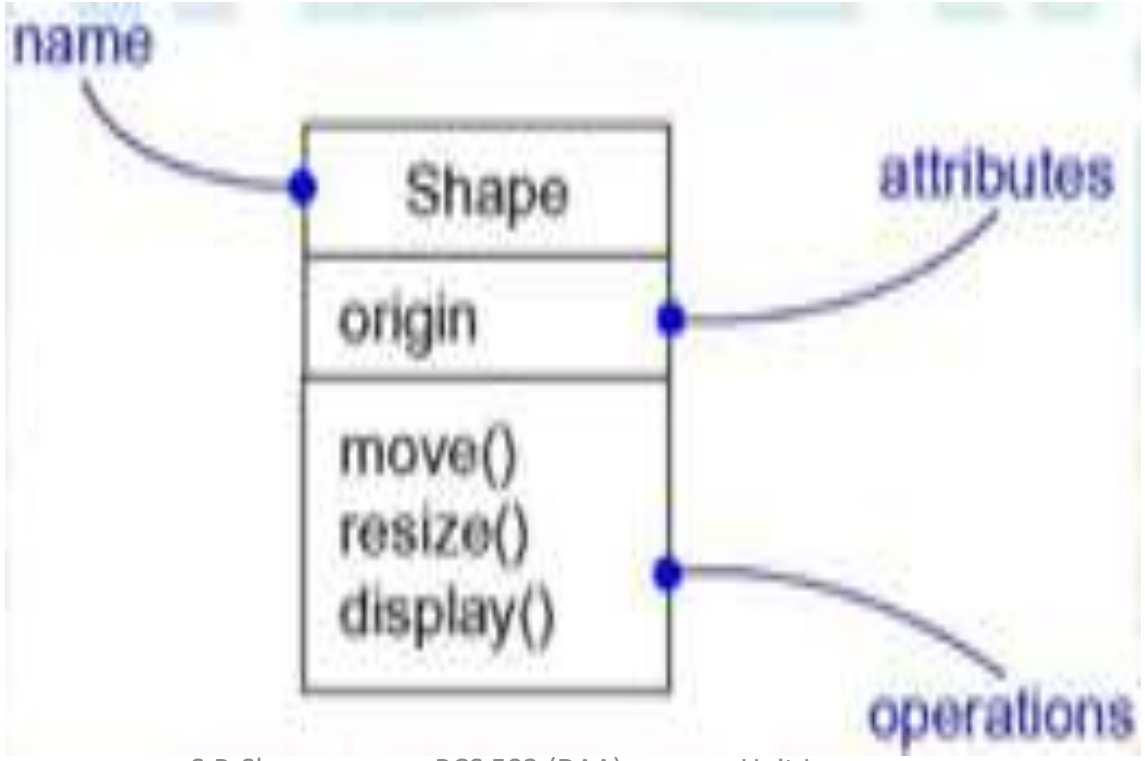
Lecture 4

- Modeling Concepts
- Class Diagram
- Object Diagram

Class Diagram

1.Classes

- A class is a description of a set of objects that share the same attributes, operations, relationships, and semantics. A class implements one or more interfaces.
- classes to represent software things, hardware things, and even things that are purely conceptual.



Class Diagram

Names

Every class must have a name that distinguishes it from other classes.

A *name* is a textual string. That name alone is known as a *simple name;* a *path name* is the class name prefixed by the name of the package in which that class lives.

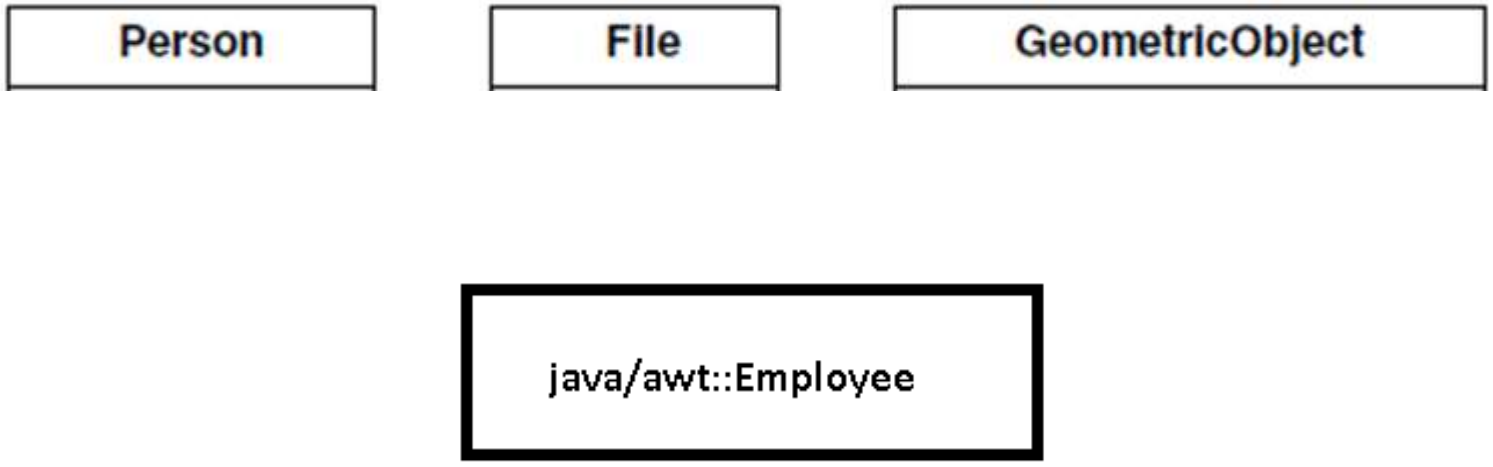
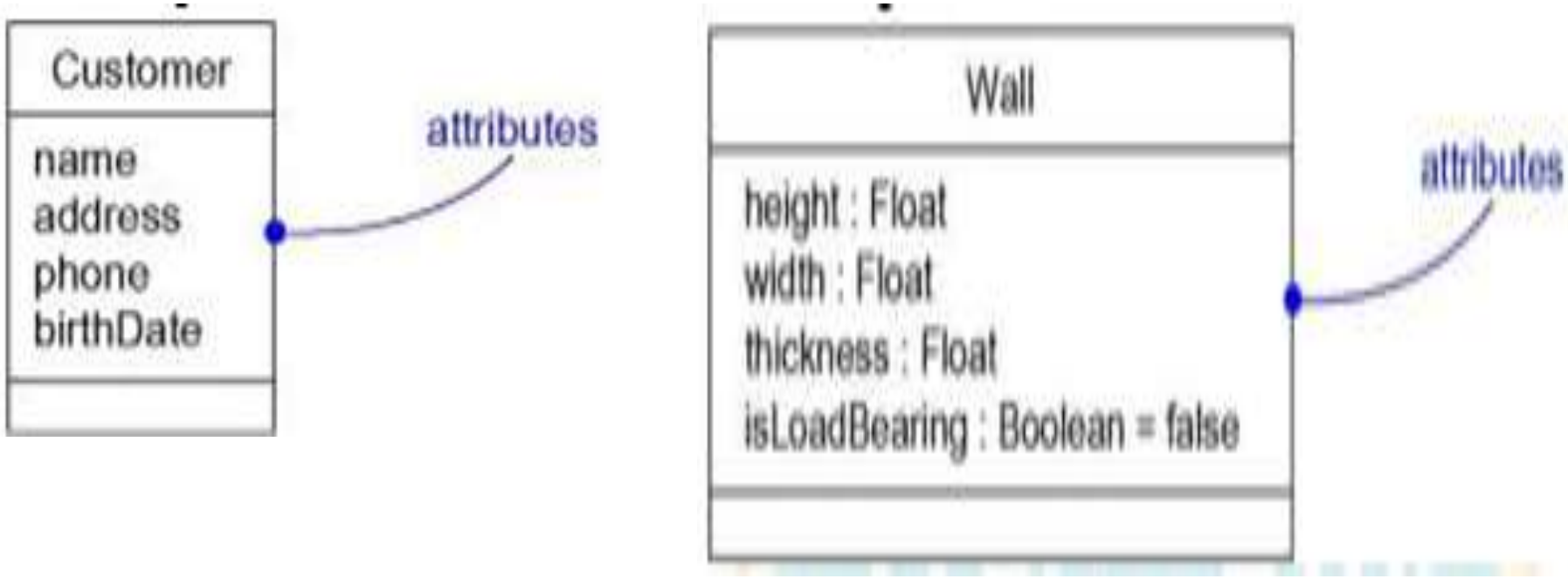


Figure 2

Class Diagram

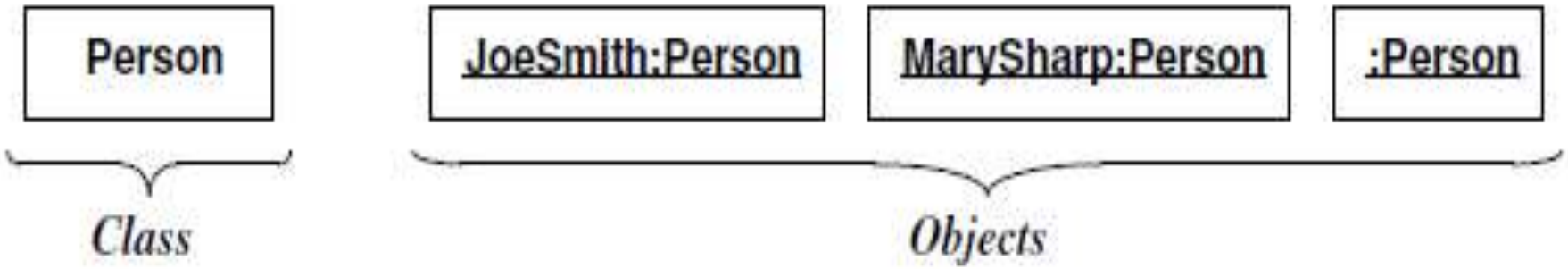
Attributes

- An *attribute* is a named property of a class that describes a range of values that instances of the property may hold.
- A class may have any number of attributes or no attributes at all.
- An attribute represents some property of the thing you are modeling that is shared by all objects of that class.



Object

- An object is a concept, abstraction, or thing with identity that has meaning for an application.
- Objects often appear as proper nouns or specific references in problem descriptions and discussions with users.
- Some objects have real-world entity like table, tree

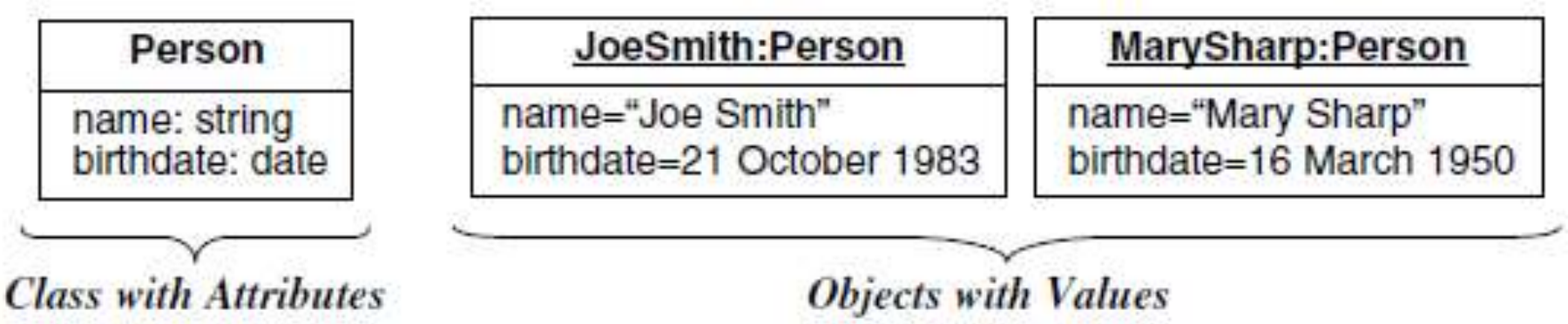


A class and objects. Objects and classes are the focus of class modeling.

Class Diagram

Attribute and Value of an object

- Each attribute is associated a specific value.

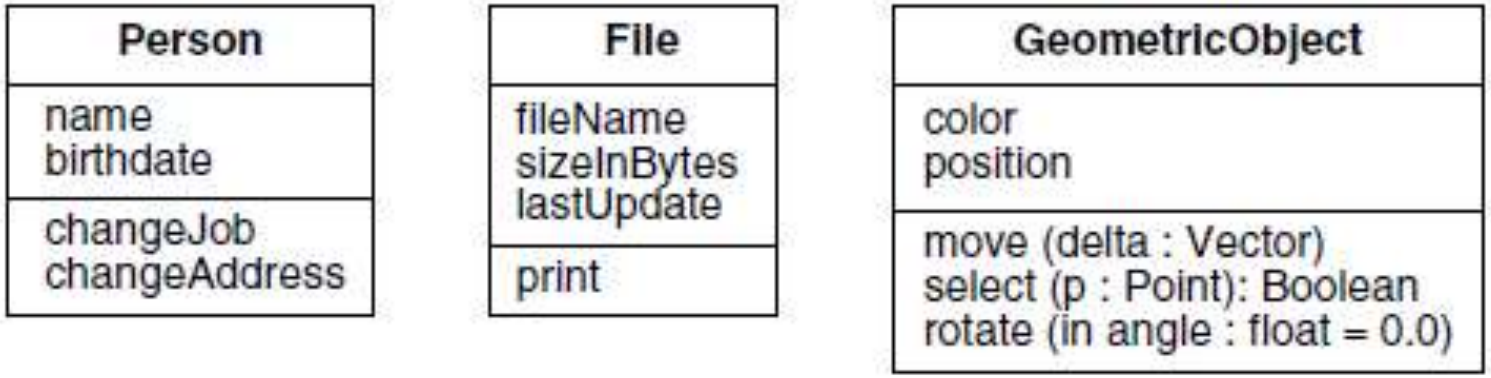


Attributes and values. Attributes elaborate classes.

Class Diagram

Operations

- It is the implementation of a service that can be requested from any object of the class to affect behavior.
- An operation is an abstraction of something you can do to an object and that is shared by all objects of that class.
- A class may have any number of operations or no operations at all.
- An *operation is a function or procedure that may be applied to or by objects in a class*



Operations. An operation is a function or procedure that may be applied to or by objects in a class.

Class Diagram

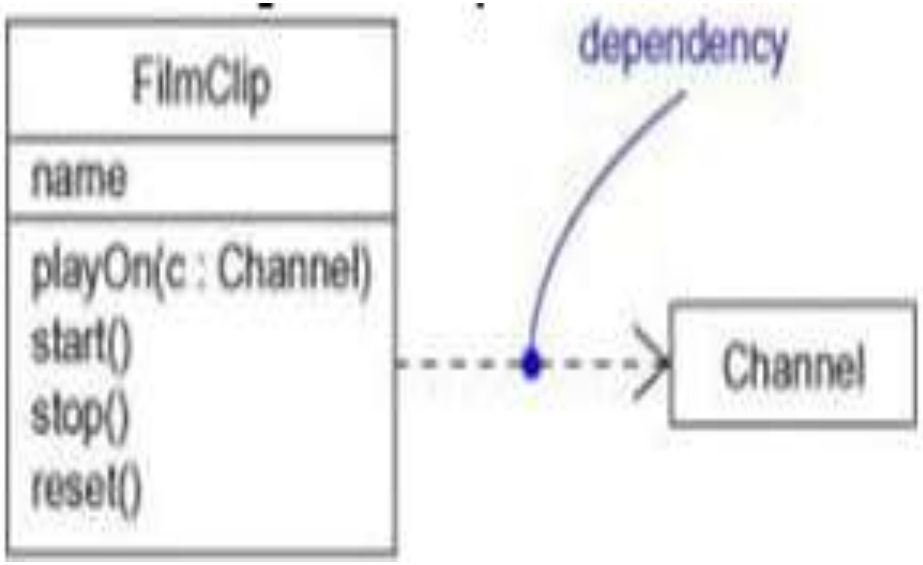
Relationships

- A *relationship* is a connection among things. In object-oriented modeling, there are three kinds of relationships that are especially important:
  - *Dependencies*
  - *Generalizations*
  - *Associations*
    - ✓ *Composition*
    - ✓ *Aggregation*

Class Diagram

Dependency

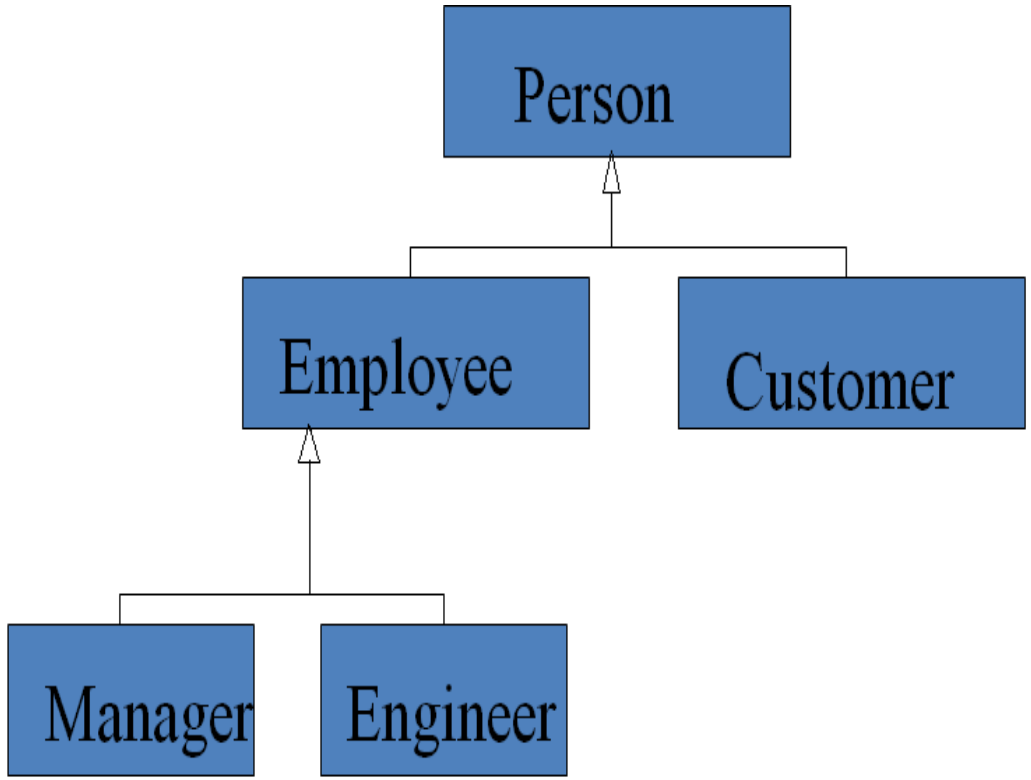
- It states that a change in specification of one thing (for example, class Event) may affect another thing that uses it (for example, class Window),but not necessarily the reverse.
- Graphically, a dependency is rendered as a dashed directed line, directed to the thing being depended on
- A dependency can have a name



Class Diagram

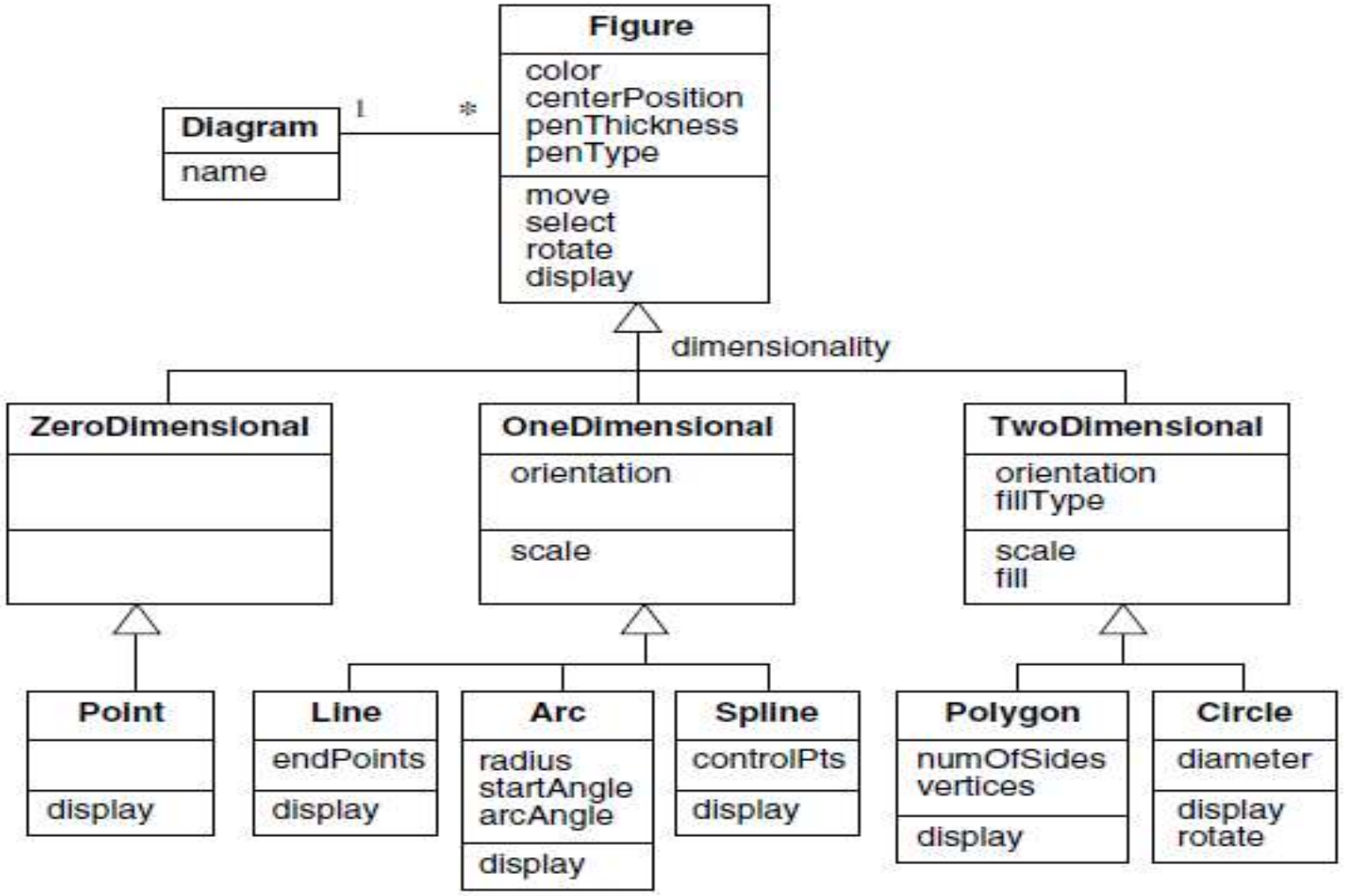
Generalization (parent child relationship)

- It is a"is-a-kind-of" relationship
- In it objects of the child may be used anywhere the parent may appear, but not the reverse.
- Graphically, it is a solid directed line with a large open arrowhead, pointing to the parent,
- It shows parent/child relationships.



Class Diagram

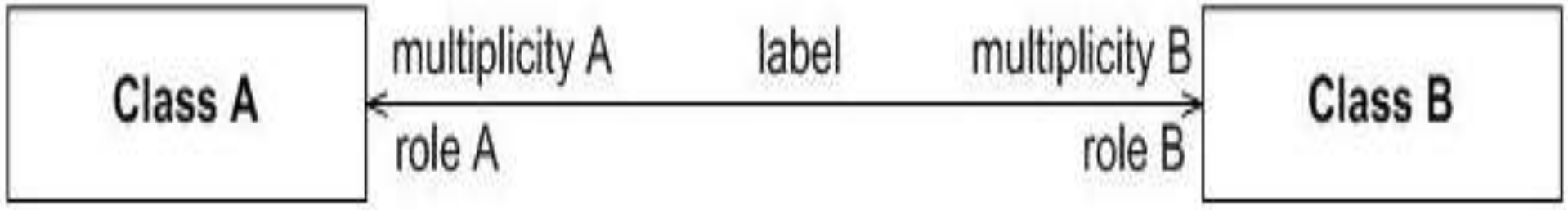
Generalization (Case study)



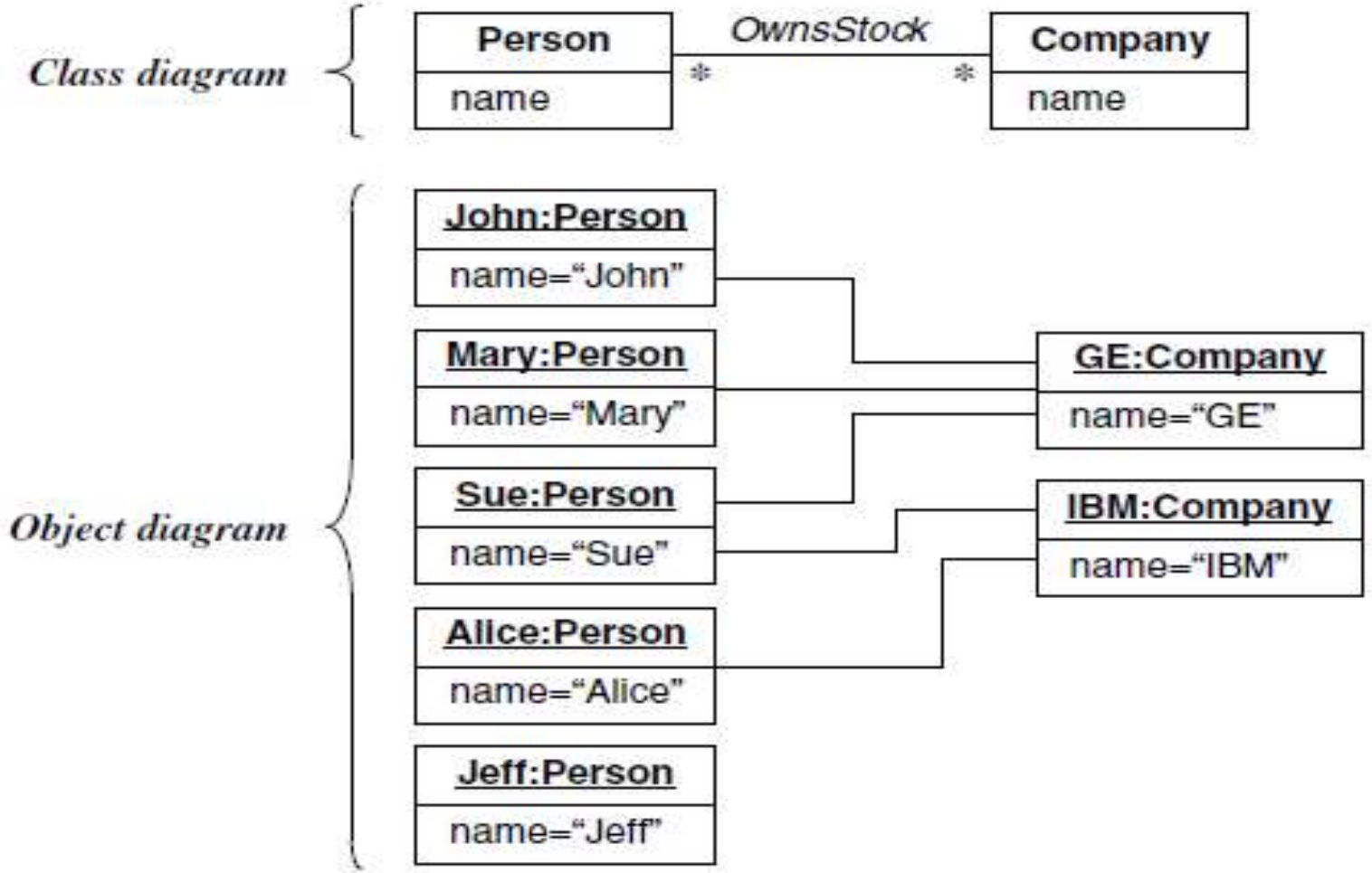
Inheritance for graphic figures. Each subclass inherits the attributes, operations, and associations of its superclasses.

Link and Association Concepts

- A ***link*** *is a physical or conceptual connection among objects*
- An **Association** specifies that objects of one thing are connected to objects of another
- This means that, given an object of the class, you can link to other objects of the same class.
- Graphically, an association is rendered as a solid line
- there are four adornments that apply to associations.
  1) **Association Name**
  2) **Role Name**
  3) **Multiplicity**



Class Diagram

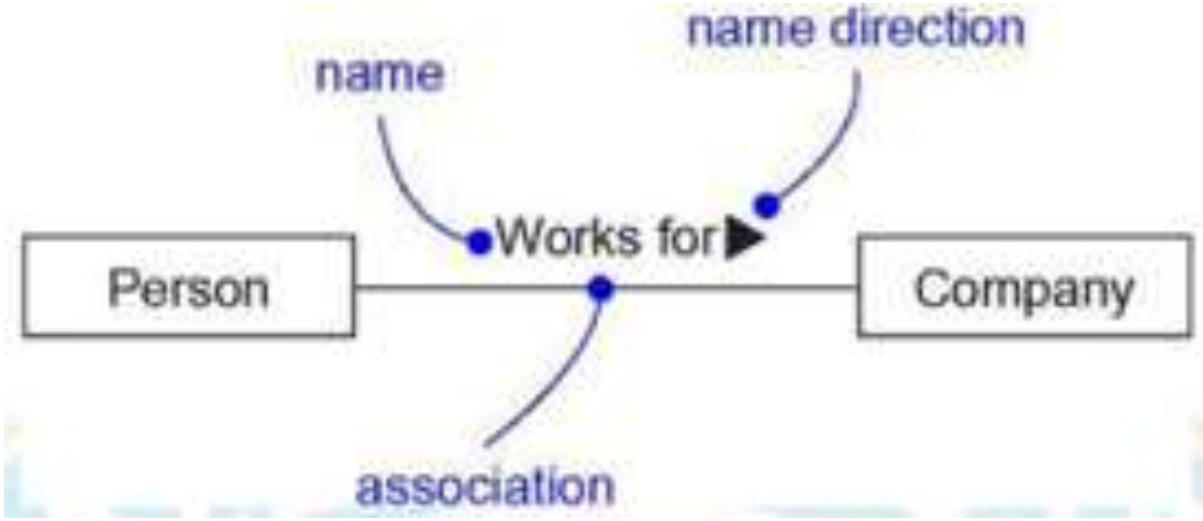


Many-to-many association. An association describes a set of potential links in the same way that a class describes a set of potential objects.

Class Diagram

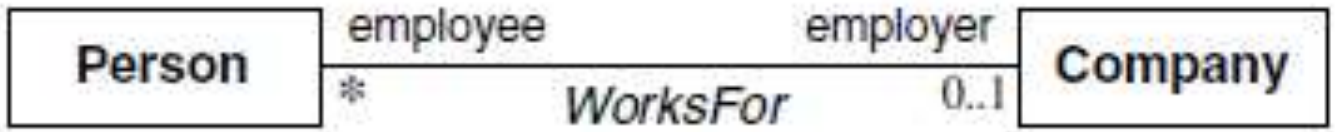
1. Association Name

- An association can have a name,
- you can give a direction to the name by providing a direction triangle that points in the direction you intend to read the name



Role Name(example)

- We can explicitly write the name the role a class plays in both end of an association.
- It is also called association end name



employee	employer
Joe Doe	Simplex
Mary Brown	Simplex
Jean Smith	United Widgets

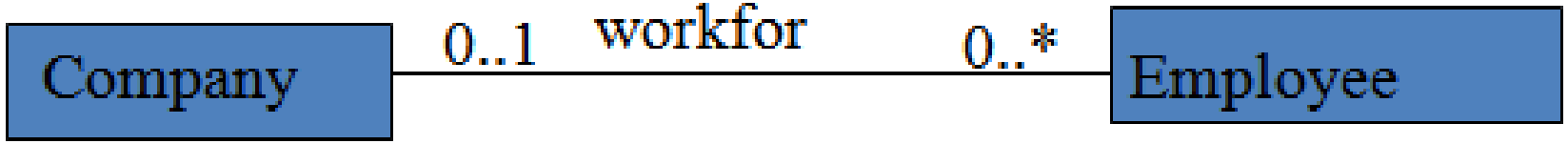
Association end names. Each end of an association can have a name.

Class Diagram

3. Multiplicity

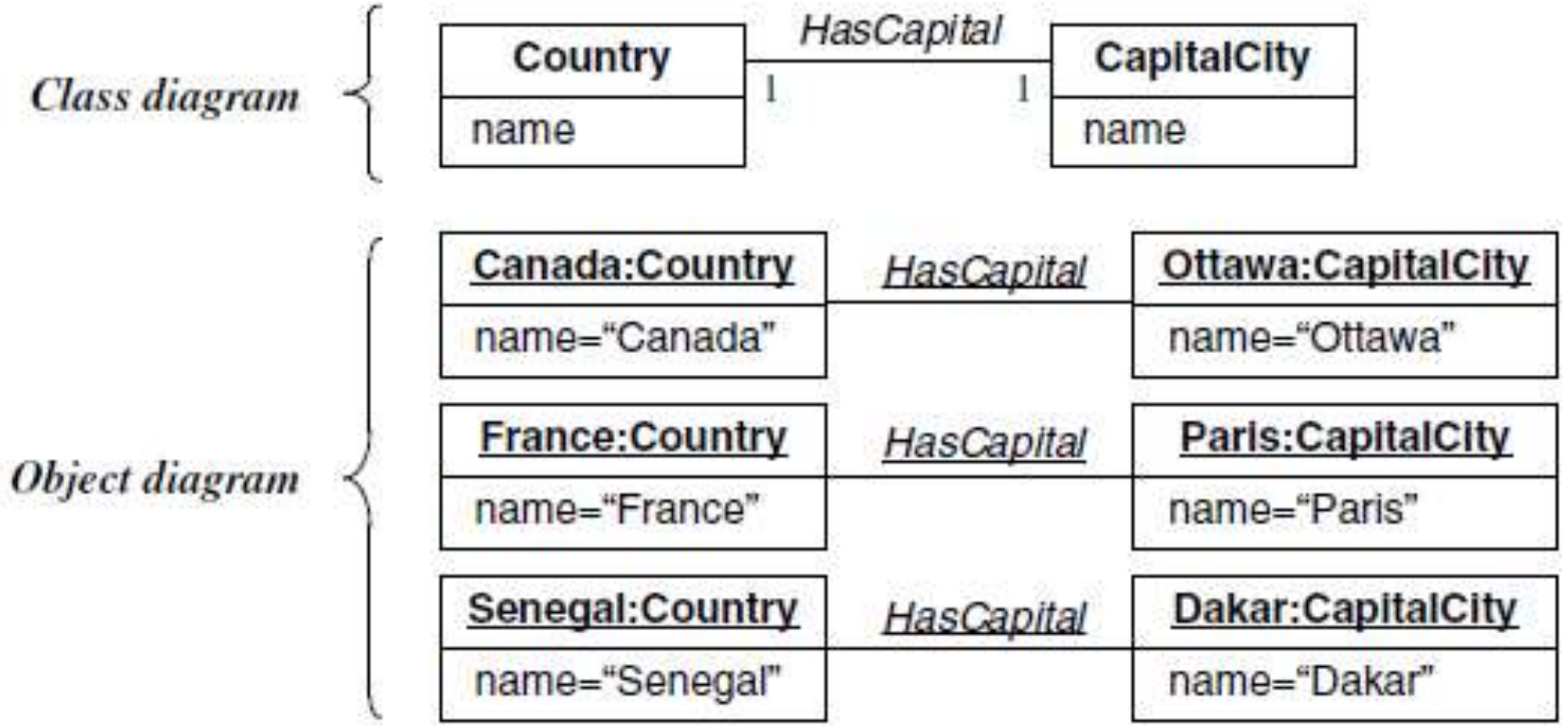
- *Multiplicity specifies the number of instances of one class that may relate to a single instance* of an associated class.
- Multiplicity constrains the number of related objects.

Multiplicity	Option	Cardinality
0..0	0	Collection must be empty
0..1		No instances or one instance
1..1	1	Exactly one instance
0..*	*	Zero or more instances
1..*		At least one instance
5..5	5	Exactly 5 instances
m..n		At least m but no more than n instances



Class Diagram

3. Multiplicity(Example)

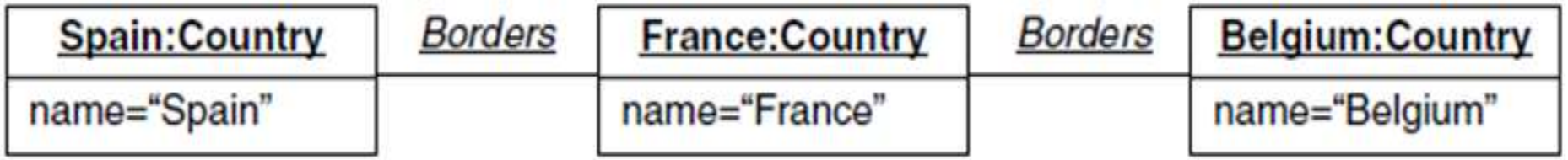


One-to-one association. Multiplicity specifies the number of instances of one class that may relate to a single instance of an associated class.

Class Diagram

Exercise

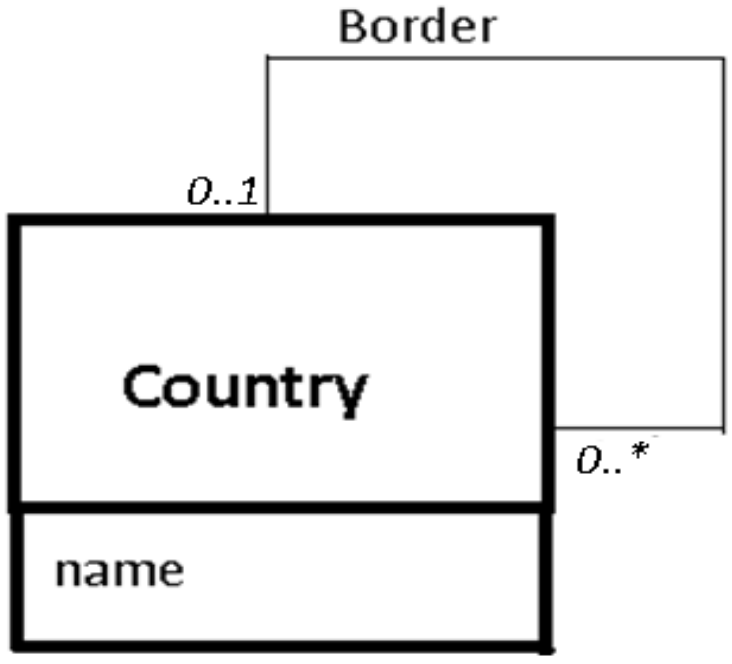
Q1. Prepare a class diagram from the object diagram



Object diagram for a portion of Europe

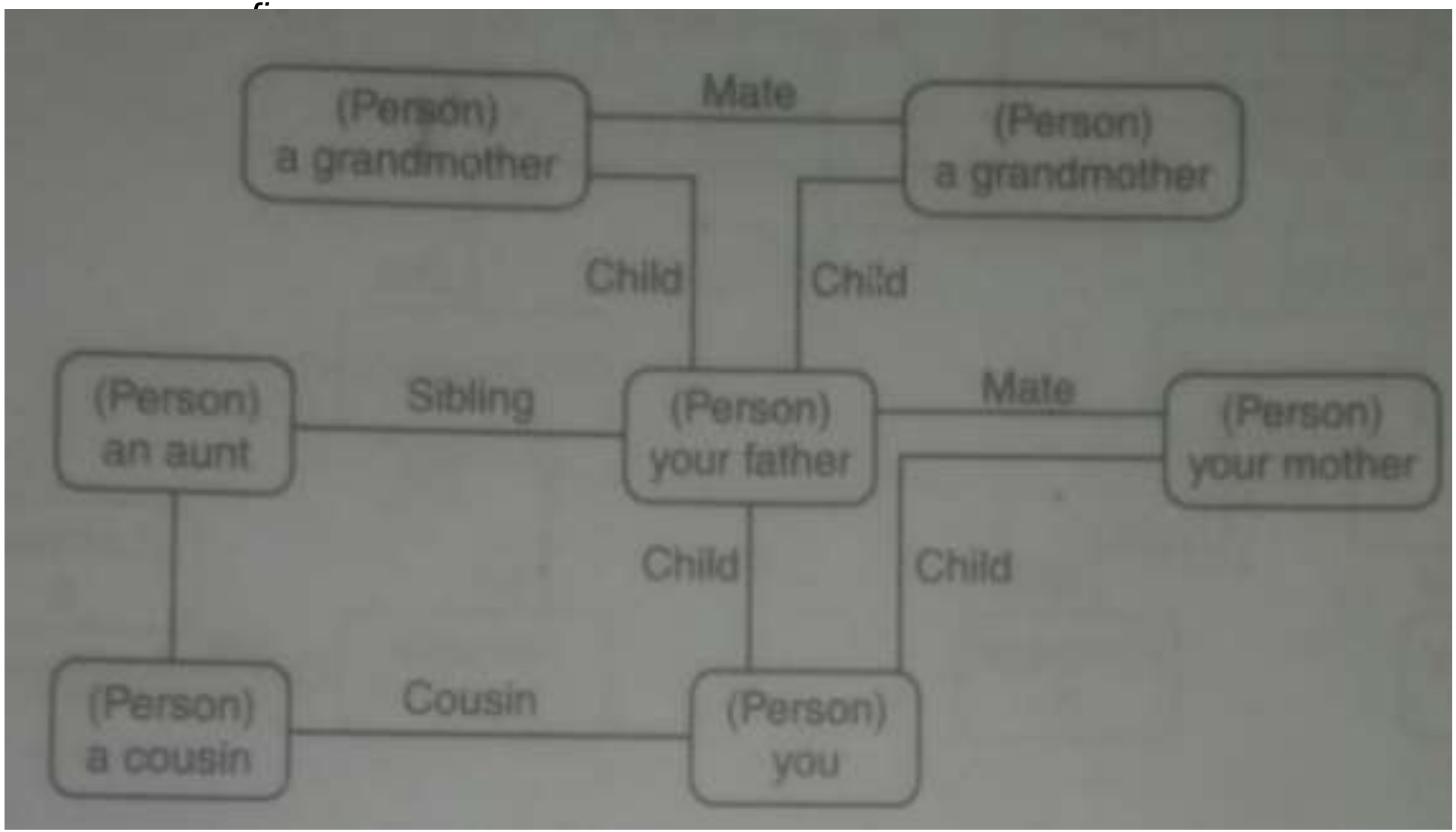
Solution

Answer 1



Class Diagram
)

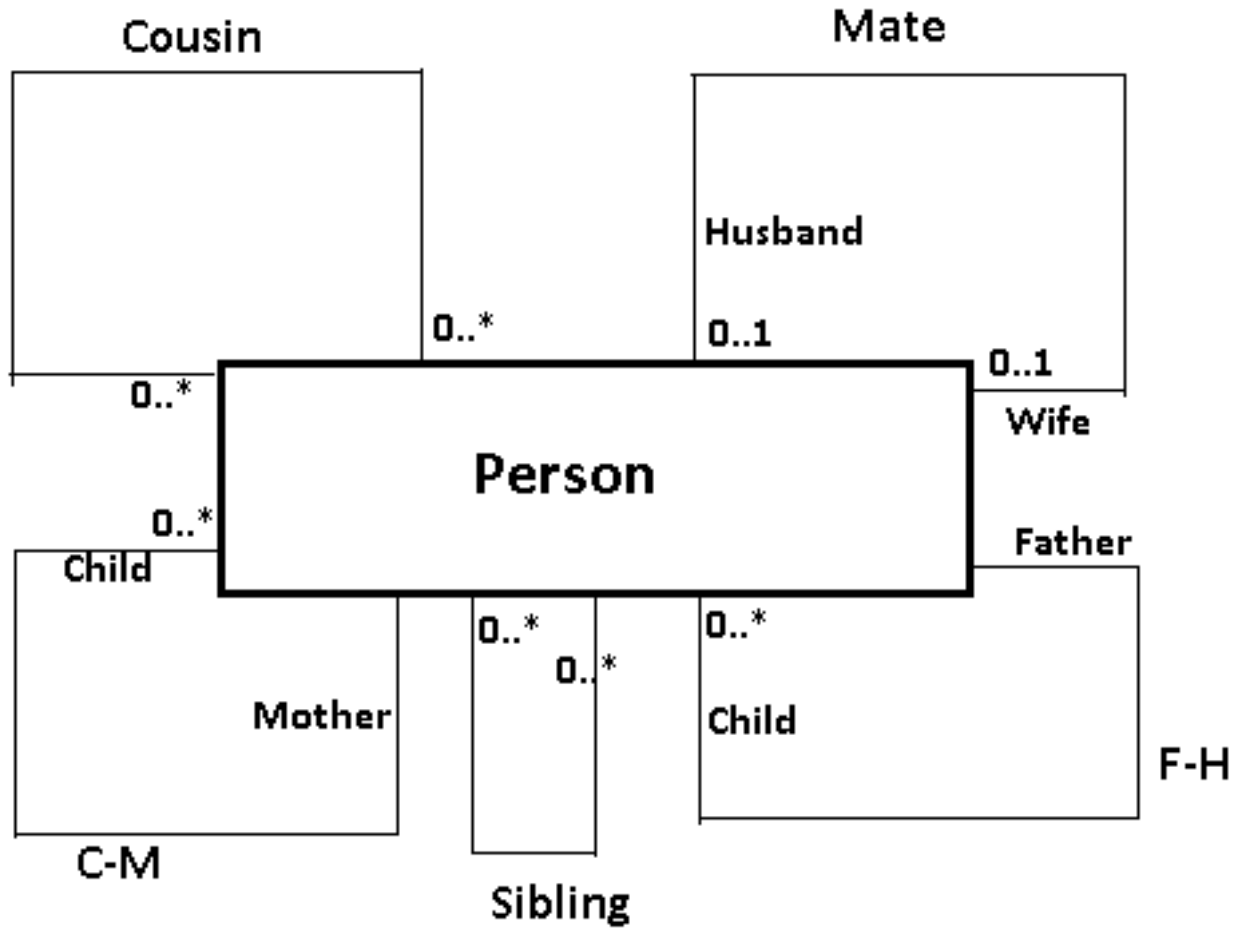
Q3. prepare a class diagram from the instance diagram given in the



Class Diagram

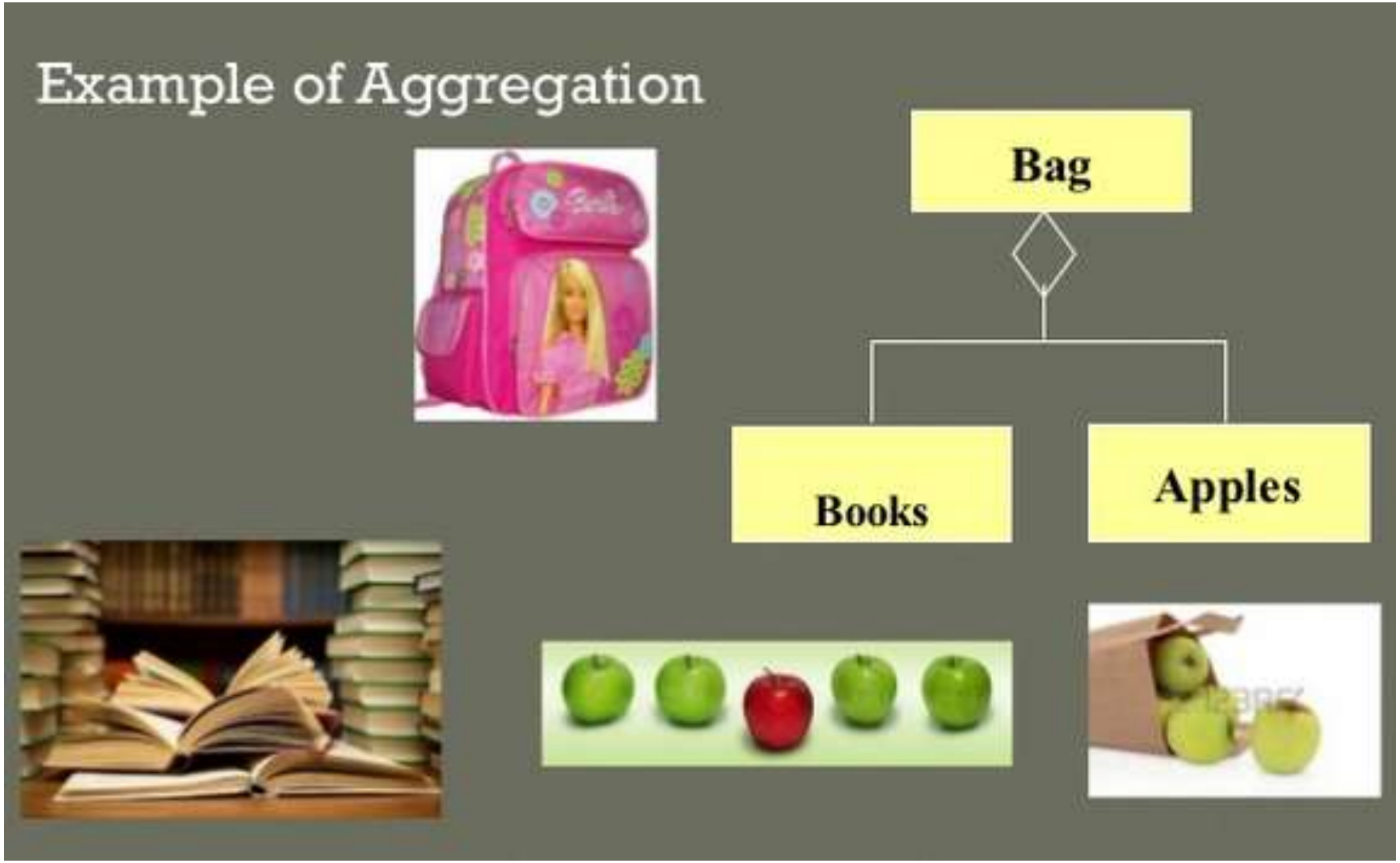
Solution

Answer 3



0..* : o or many
nothing specify then exactly one

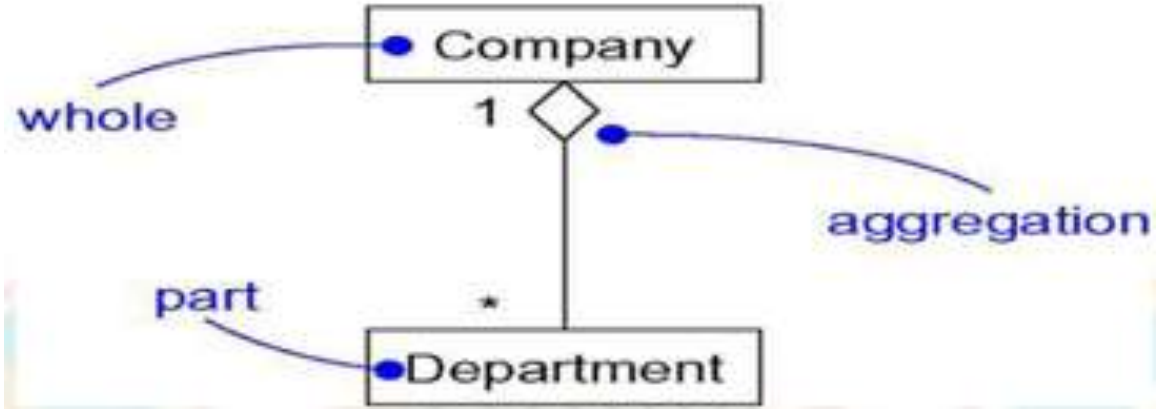
Class Diagram



Class Diagram

4. Aggregation

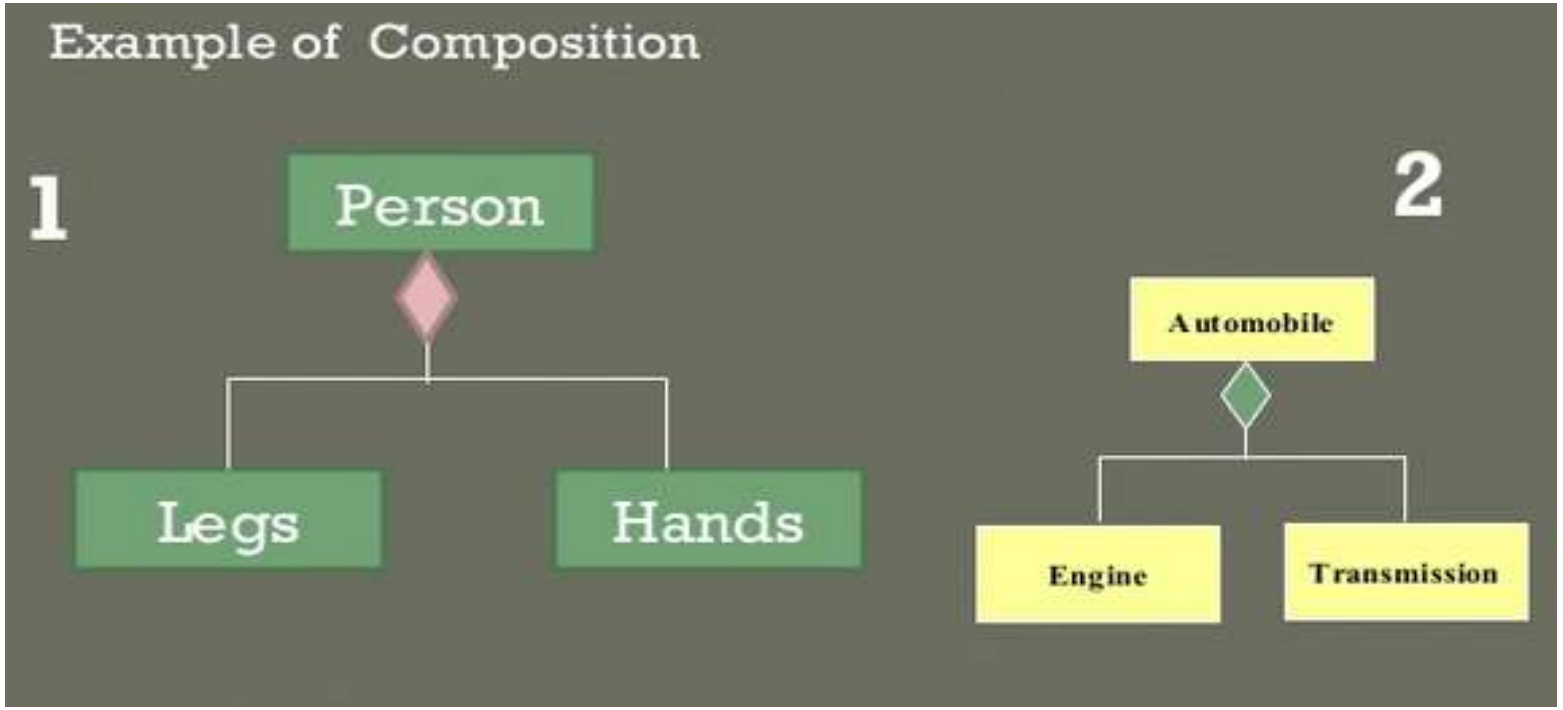
- *Aggregation is a strong form of association in which an aggregate object is made of constituent* parts.
- It is represents a "has-a" relationship,
- It shows the relationship between assembly and its components.
- Meaning that an object of the whole(assembly) has objects of the part(component).
- It is it represented by an open diamond at the whole(assembly) end.



Class Diagram

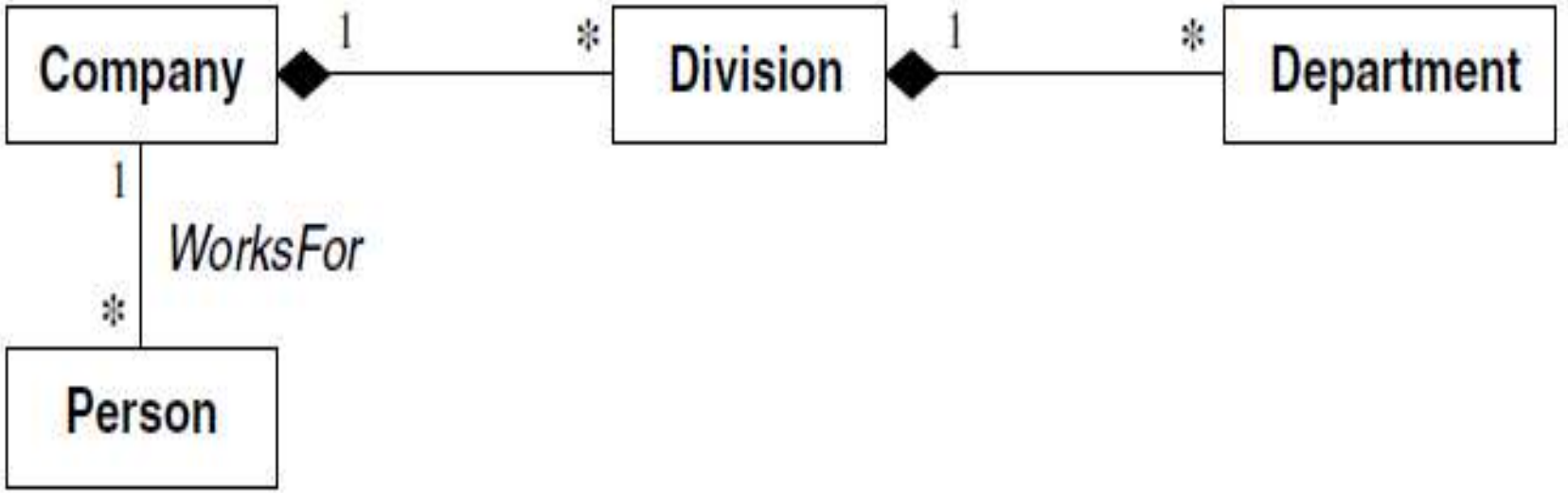
5. Composition

- It is represents a "has-a" relationship,
- It shows the relationship between assembly and its life time associated components.
- meaning that an object of the whole(assembly) has objects of the part(component).
- It is it represented by an solid diamond at the whole(assembly) end,



Class Diagram

Composition(Example 2)

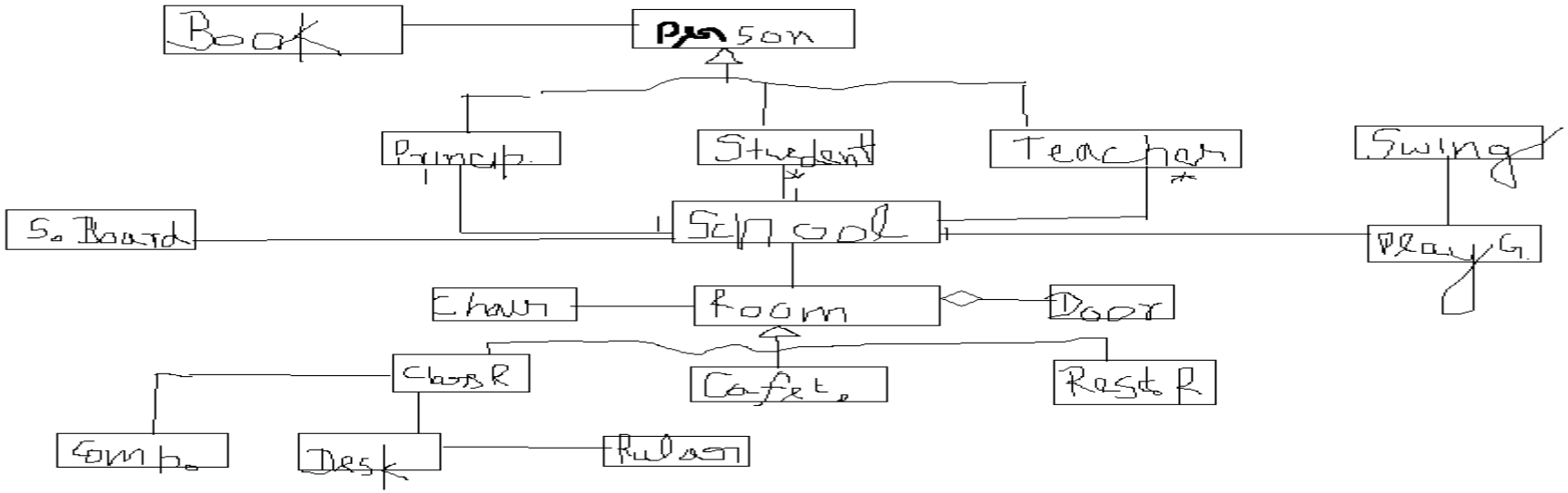


Composition. With composition a constituent part belongs to at most one assembly and has a coincident lifetime with the assembly.

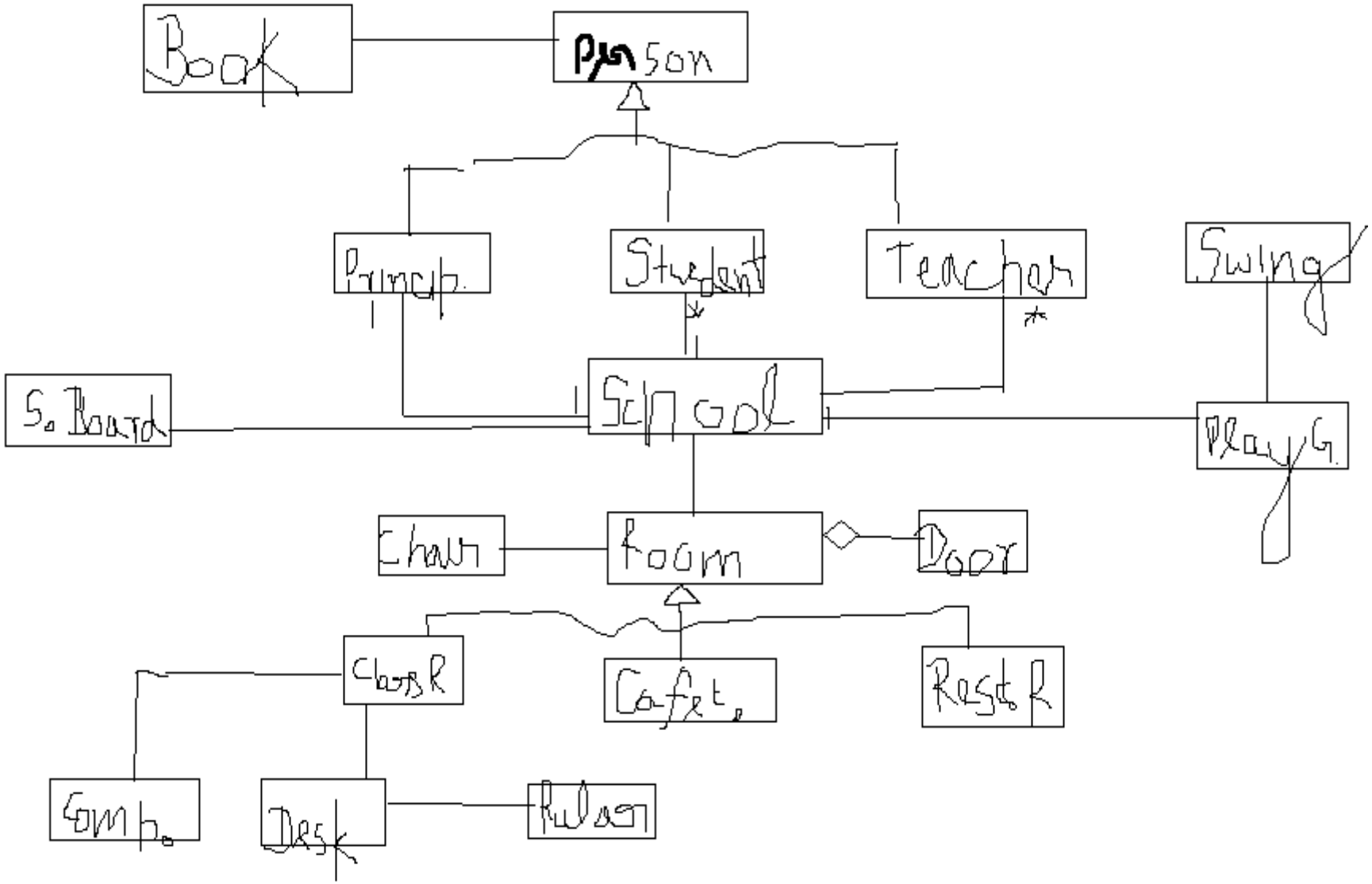
Problem

Q1. Design a class diagram for the following object classes using association name where needed and you may add addition object classes if required.

Object classes: school, play ground, principal, school board, class room, book, student, teacher, cafeteria, restroom, computer, desk, chair, ruler, door, swing.



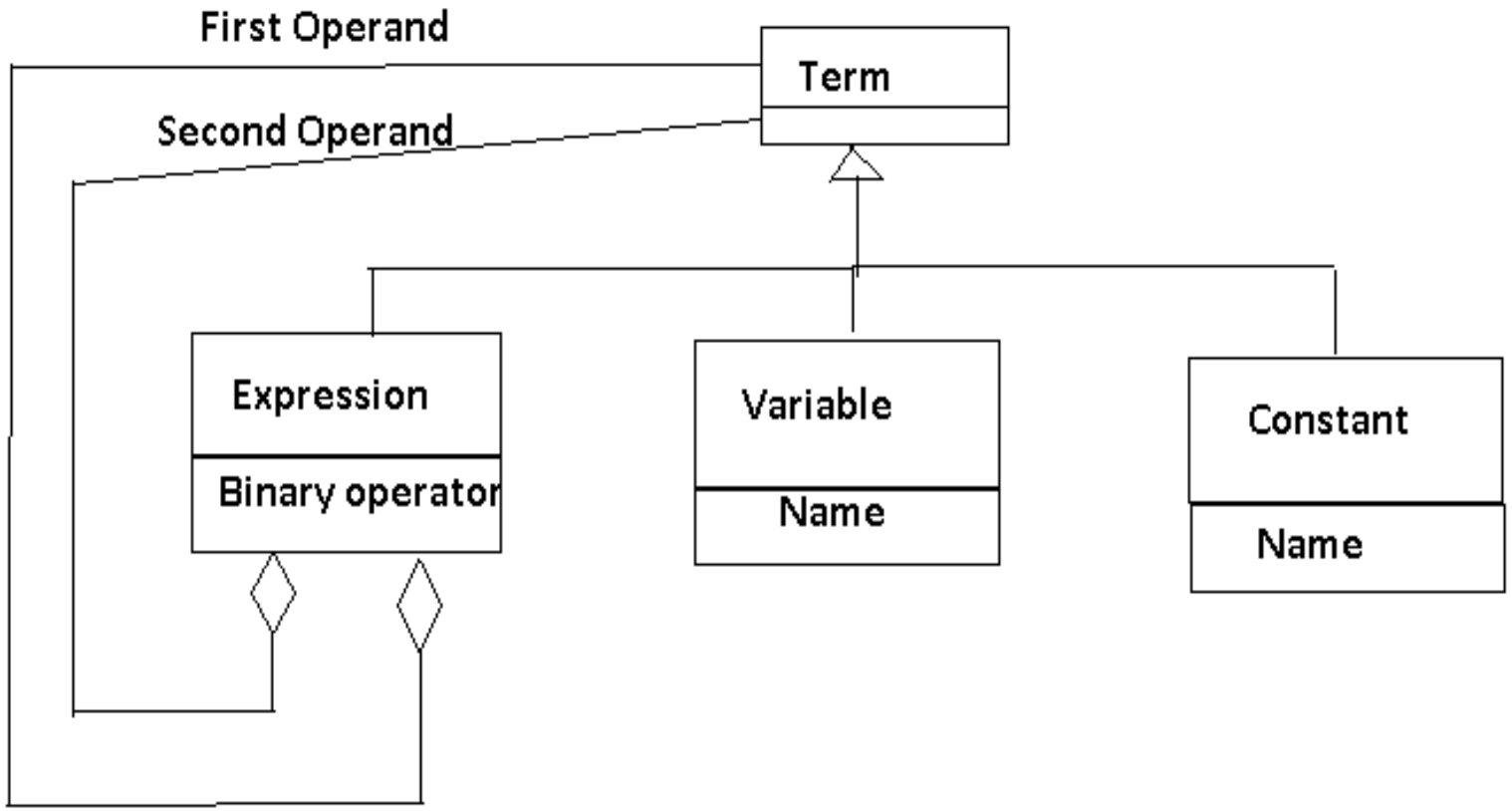
Problem


Book
Person
Princip.
Student
Teacher
Swing
S. Board
School
Play G.
Chair
Room
Door
ClassR
Cafet.
RestR
Gamb.
Desk
Ruler

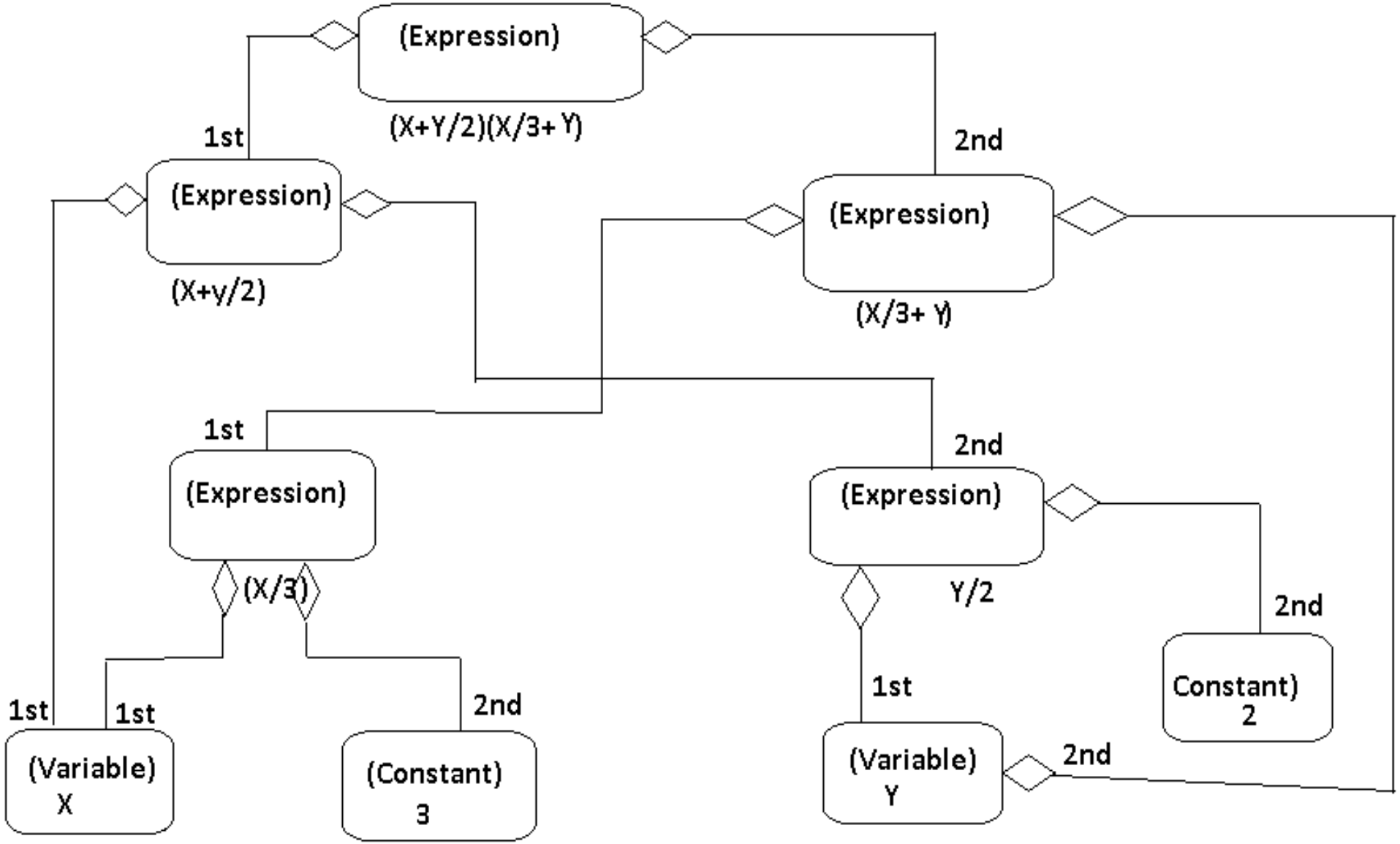
Problem

Q2. Prepare an instance diagram for the given class diagram for the expression

(X +Y /2)/((X/3+Y)



Solution


(Expression)
(X+Y/2)(X/3+ Y)
1st
2nd
(Expression)
(X+y/2)
(Expression)
(X/3+ Y)
1st
2nd
(Expression)
(X/3)
(Expression)
Y/2
1st
2nd
1st
1st
2nd
(Variable)
X
(Constant)
3
(Variable)
Y
Constant)
2
2nd

Lecture 5

- Control Statements
- Decision Making
- Looping

Control Statements

- Java compiler executes the code from top to bottom.
- The statements in the code are executed according to the order in which they appear. However, java provides statements that can be used to control the flow of Java code. Such statements are called control flow statements.
- Java provides three types of control flow statements:

I. Decision Making statements

II. Loop statements

III. Jump statements

Decision Making Statements

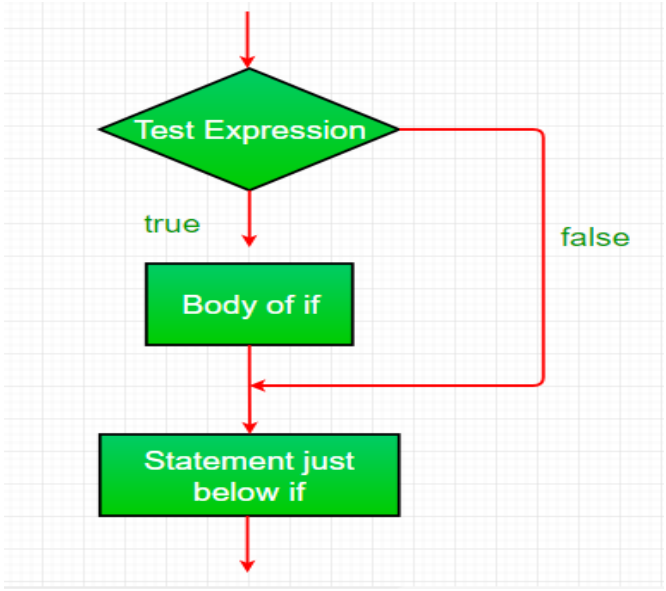
- Decision-making statements decide which statement to execute and when.

- Decision-making statements evaluate the Boolean expression and control the program flow depending upon the result of the condition provided.

Decision Making Statements

1.If Statement:

- In Java, the "if" statement is used to evaluate a condition. The control of the program is diverted depending upon the specific condition.



Decision Making Statements

1.If Statement:

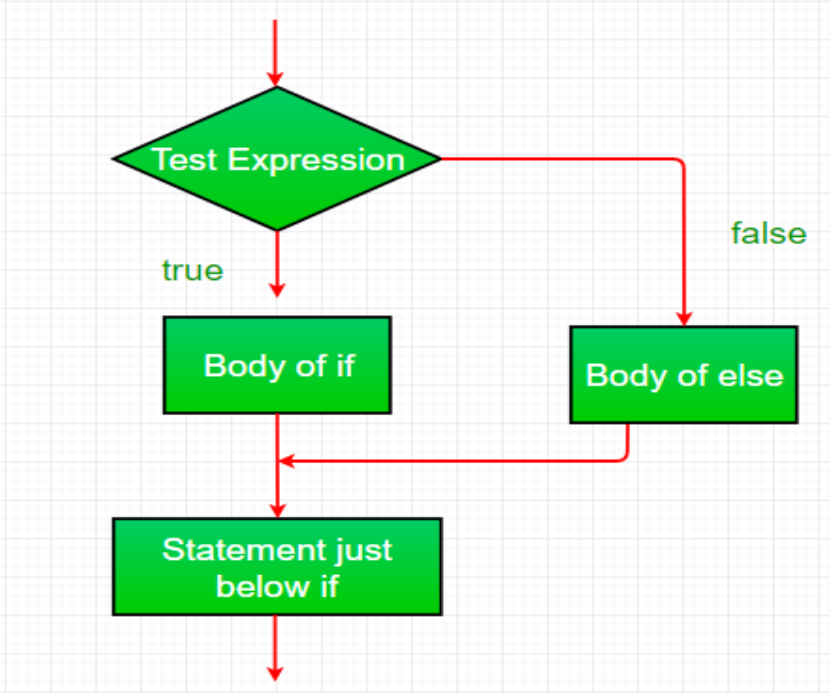
- In Java, the "if" statement is used to evaluate a condition. The control of the program is diverted depending upon the specific condition.

```
1. //Java Program to demonstate the use of if statement.
2. public class IfExample {
3. public static void main(String[] args) {
4.     //defining an 'age' variable
5.     int age=20;
6.     //checking the age
7.     if(age>18){
8.         System.out.print("Age is greater than 18");
9.     }
10.}
11.}
```

Decision Making Statements

If-else:

- We can use the else statement with if statement to execute a block of code when the condition is false.



Decision Making Statements

If-else:

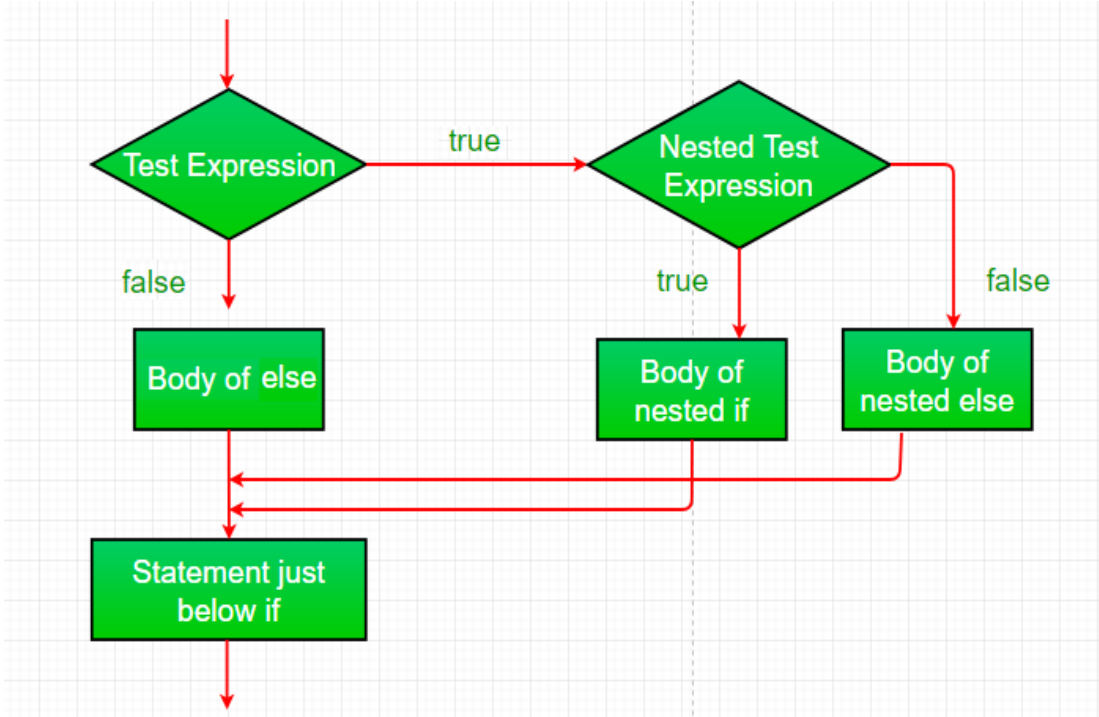
- We can use the else statement with if statement to execute a block of code when the condition is false.

```
1.  public class IfElseExample {
2.  public static void main(String[] args) {
3.      //defining a variable
4.      int number=13;
5.      //Check if the number is divisible by 2 or not
6.      if(number%2==0){
7.          System.out.println("even number");
8.      }else{
9.          System.out.println("odd number");
10.     }
11. }
12. }
```

Decision Making Statements

Nested-if:

A nested if is an if statement that is the target of another if or else. Nested if statements means an if statement inside an if statement.



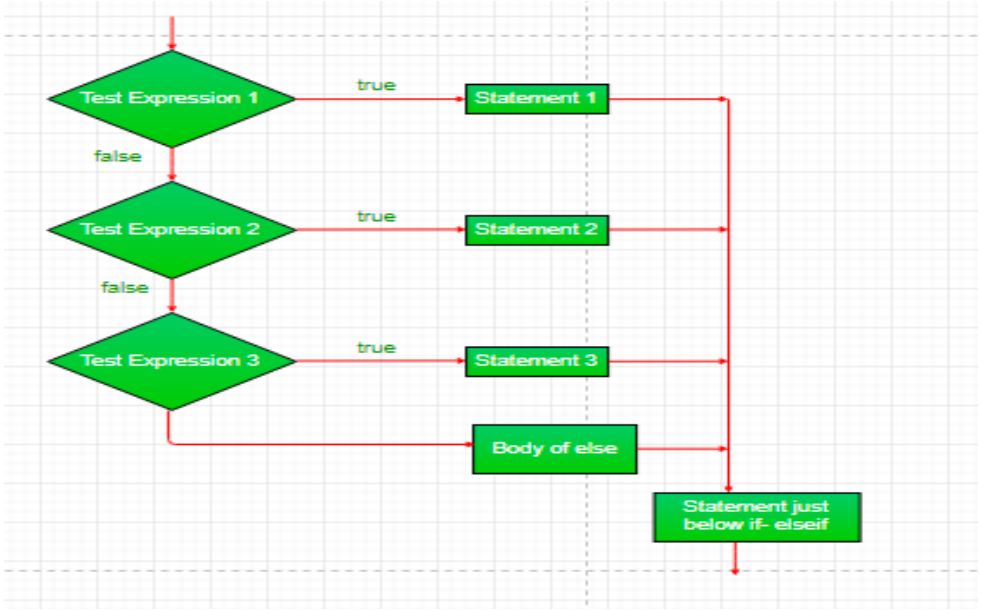
Decision Making Statements

```
1.//Java Program to demonstrate the use of Nested If Statement.
2.public class JavaNestedIfExample {
3.public static void main(String[] args) {
4.    //Creating two variables for age and weight
5.    int age=20;
6.    int weight=80;
7.    //applying condition on age and weight
8.    if(age>=18){
9.        if(weight>50){
10.            System.out.println("You are eligible to donate blood");
11.        }
12.    }
13.}}
```

Decision Making Statements

if-else-if-ladder:

- if statements are executed from the top down.
- As soon as one of the conditions controlling the if is true, the statement associated with that if is executed, and the rest of the ladder is bypassed.



Decision Making Statements

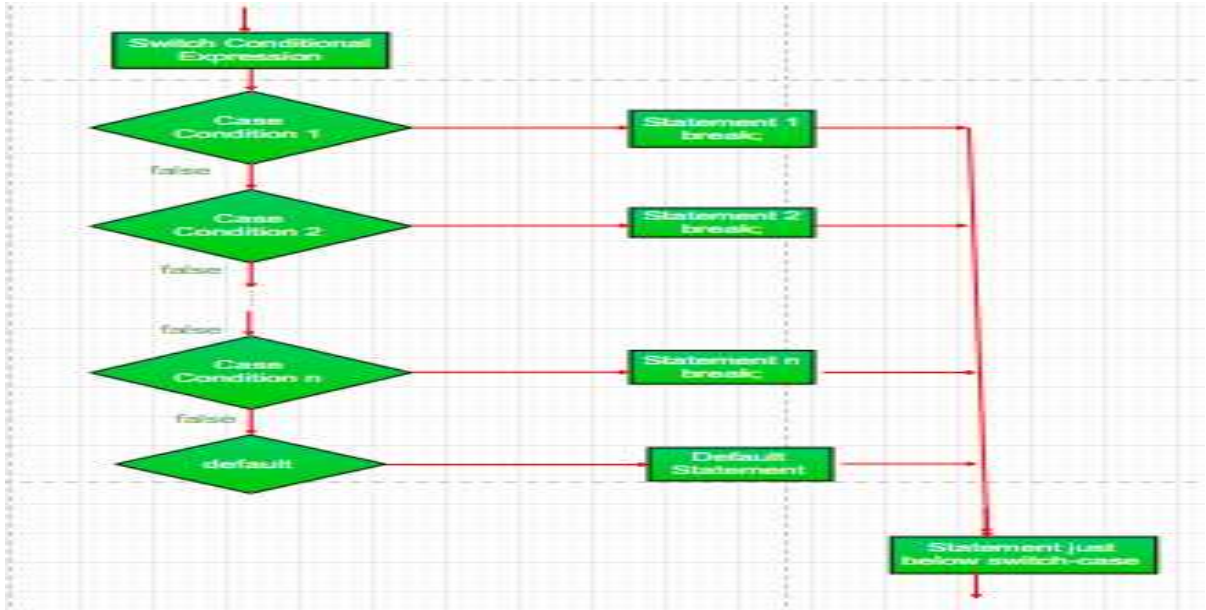
```
1.public class PositiveNegativeExample {
2.public static void main(String[] args) {
3.    int number=-13;
4.    if(number>0){
5.    System.out.println("POSITIVE");
6.    }else if(number<0){
7.    System.out.println("NEGATIVE");
8.    }else{
9.    System.out.println("ZERO");
10.    }
11.}
12.}
```

Decision Making Statements

switch statement

The switch statement is a multiway branch statement. It provides an easy way to dispatch execution to different parts of code based on the value of the expression.

Syntax:



Decision Making Statements

```
1.public class SwitchExample {
2.public static void main(String[] args) {
3.    //Declaring a variable for switch expression
4.    int number=20;
5.    //Switch expression
6.    switch(number){
7.    //Case statements
8.    case 10: System.out.println("10");
9.    break;
10.     case 20: System.out.println("20");
11.     break;
12.     case 30: System.out.println("30");
13.     break;
14.     //Default case statement
15.     default:System.out.println("Not in 10, 20 or 30");
16.     }
17.}
18.}
```

Decision Making Statements

Example:

1. Finding Month

2. Program to check Vowel or Consonant:

Decision Making Statements

Java Switch Statement is fall-through

The Java switch statement is fall-through. It means it executes all statements after the first match if a break statement is not present.

Java Switch Statement with String

Java allows us to use strings in switch expression since Java SE 7. The case statement should be string literal.

Java Nested Switch Statement

We can use switch statement inside other switch statement in Java. It is known as nested switch statement.

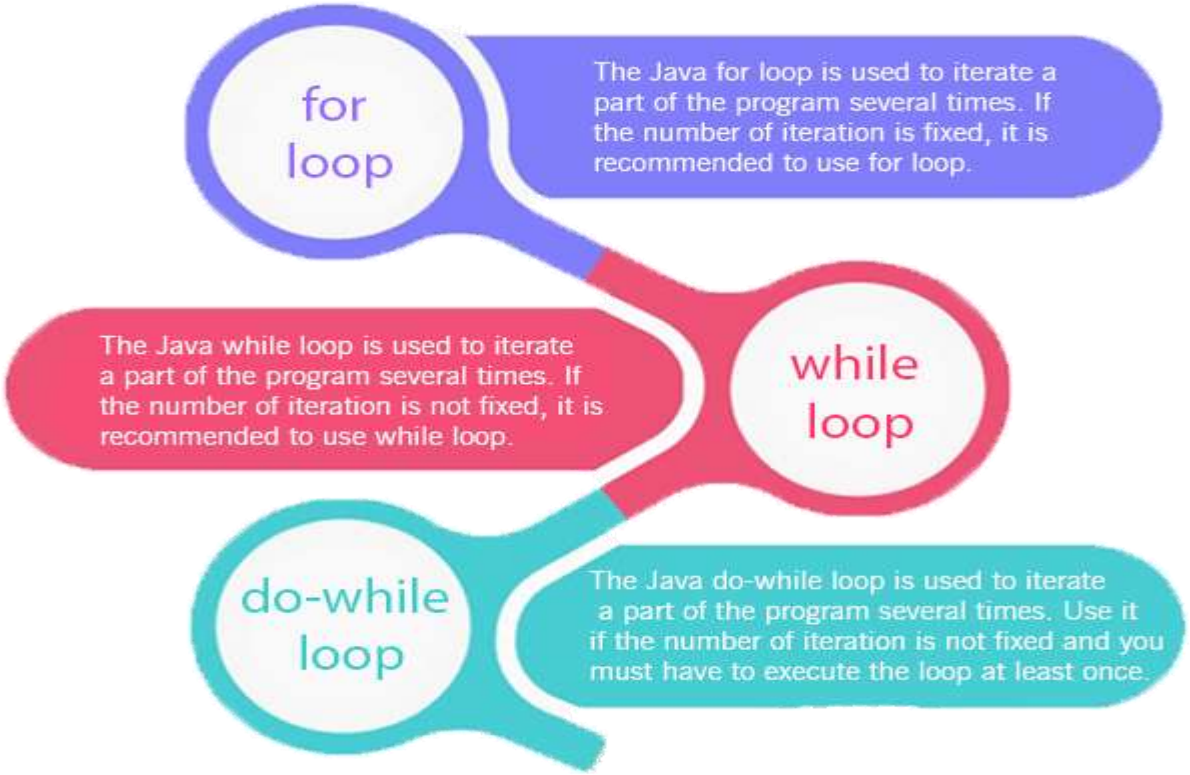
Java Enum in Switch Statement

Java allows us to use enum in switch statement. Java enum is a class that represent the group of constants. (immutable such as final variables). We use the keyword enum and put the constants in curly braces separated by comma

Looping

Loops in Java

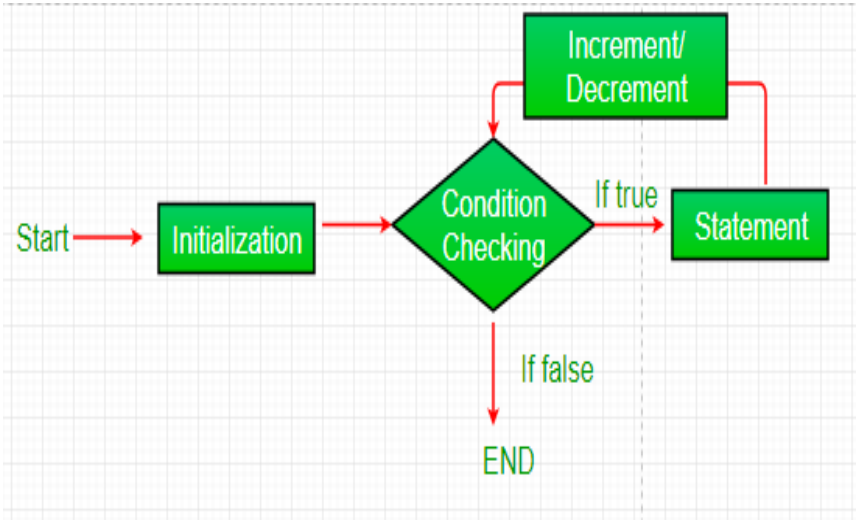
The Java *for loop* is used to iterate a part of the program several times. If the number of iteration is **fixed**, it is recommended to use for loop.



Looping

for loop:

- for loop provides a concise way of writing the loop structure.
- Unlike a while loop, a for statement consumes the initialization, condition and increment/decrement in one line thereby providing a shorter, easy to debug structure of looping.



Looping

- **Initialization condition:** Here, we initialize the variable in use. It marks the start of a for loop. An already declared variable can be used or a variable can be declared, local to loop only.
- **Testing Condition:** It is used for testing the exit condition for a loop. It must return a boolean value. It is also an **Entry Control Loop** as the condition is checked prior to the execution of the loop statements.
- **Statement execution:** Once the condition is evaluated to true, the statements in the loop body are executed.
- **Increment/ Decrement:** It is used for updating the variable for next iteration.
- **Loop termination:**When the condition becomes false, the loop terminates marking the end of its life cycle.

Looping

```
1.//Java Program to demonstrate the example of for loop
2.//which prints table of 1
3.public class ForExample {
4.public static void main(String[] args) {
5.    //Code of Java for loop
6.    for(int i=1;i<=10;i++){
7.        System.out.println(i);
8.    }
9.}
10.}
```

Looping

Java Nested for Loop

If we have a for loop inside the another loop, it is known as nested for loop. The inner loop executes completely whenever outer loop executes.

```
1. public class NestedForExample {
2. public static void main(String[] args) {
3. //loop of i
4. for(int i=1;i<=3;i++){
5. //loop of j
6. for(int j=1;j<=3;j++){
7.         System.out.println(i+" "+j);
8. }//end of i
9. }//end of j
10. }
11. }
```

Looping

Java for-each Loop

- The for-each loop is used to traverse array or collection in Java. It is easier to use than simple for loop because we don't need to increment value and use subscript notation.
- It works on the basis of elements and not the index. It returns element one by one in the defined variable.

Syntax:

```
for(data_type variable : array_name){
//code to be executed
}
```

```
1.  public class ForEachExample {
2.  public static void main(String[] args) {
3.      //Declaring an array
4.      int arr[]={12,23,44,56,78};
5.      //Printing array using for-each loop
6.      for(int i:arr){
7.          System.out.println(i);
8.      }
9.  }
10. }
```

Looping

Java Labeled For Loop

- We can have a name of each Java for loop. To do so, we use label before the for loop. It is useful while using the nested for loop as we can break/continue specific for loop.

Syntax:

```
labelname:
for(initialization; condition; increment/decrement){
}
```

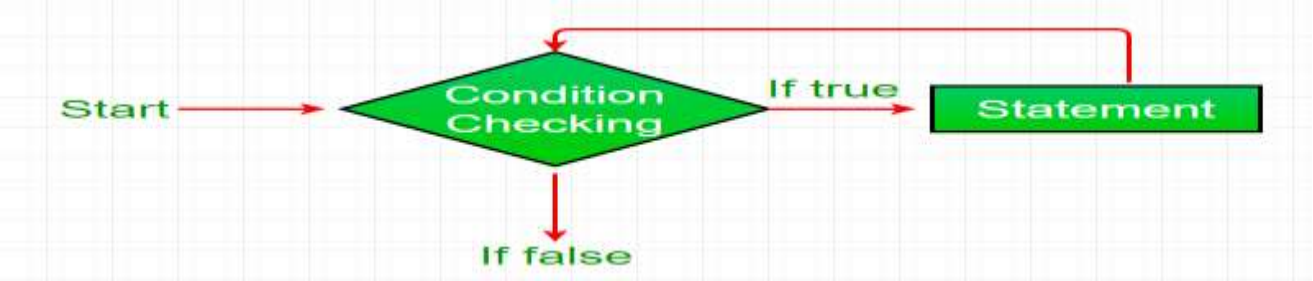
Looping

```
1. public class LabeledForExample {
2. public static void main(String[] args) {
3.     //Using Label for outer and for loop
4.     aa:
5.         for(int i=1;i<=3;i++){
6.             bb:
7.                 for(int j=1;j<=3;j++){
8.                     if(i==2&&j==2){
9.                         break aa;
10.                    }
11.                    System.out.println(i+" "+j);
12.                }
13.        }
14.}
15.}
```

Looping

While loop:

- A while loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. The while loop can be thought of as a repeating if statement.
- While loop starts with the checking of condition. If it evaluated to true, then the loop body statements are executed otherwise first statement following the loop is executed. For this reason it is also called **Entry control loop**



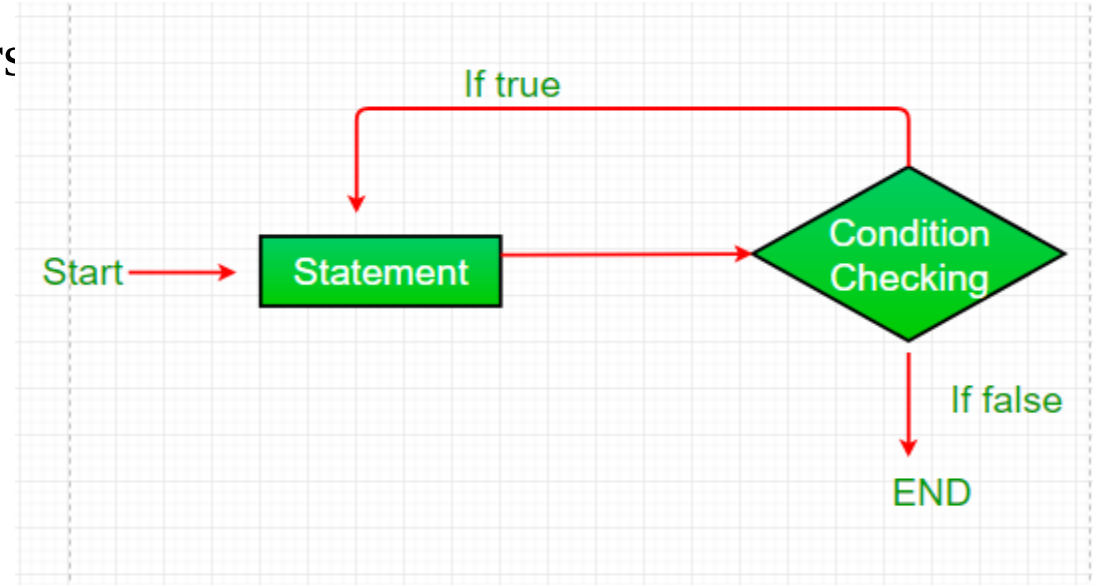
Looping

```
1.public class WhileExample {
2.public static void main(String[] args) {
3.    int i=1;
4.    while(i<=10){
5.        System.out.println(i);
6.    i++;
7.    }
8.}
9.}
```

Looping

Do-while:

- do while loop is similar to while loop with only difference that it checks for condition after executing the statements, and therefore is an example of **Exit Control Loop.**
- do while loop starts with the execution of the statement(s). There is no checking of any condition for the firs



Looping

```
1.public class DoWhileExample {
2.public static void main(String[] args) {
3.    int i=1;
4.    do{
5.        System.out.println(i);
6.    i++;
7.    }while(i<=10);
8.}
9.}
```

Looping Comparison

Comparison	for loop	while loop	do-while loop
Introduction	The Java for loop is a control flow statement that iterates a part of the programs multiple times.	The Java while loop is a control flow statement that executes a part of the programs repeatedly on the basis of given boolean condition.	The Java do while loop is a control flow statement that executes a part of the programs at least once and the further execution depends upon the given boolean condition.
When to use	If the number of iteration is fixed, it is recommended to use for loop.	If the number of iteration is not fixed, it is recommended to use while loop.	If the number of iteration is not fixed and you must have to execute the loop at least once, it is recommended to use the do-while loop.

Syntax	for(init;condition;incr/decr){
	// code to be executed
	}
		while(condition){
		//code to be executed
		}
			do{
			//code to be executed
			}while(condition);

Example	//for loop
	for(int i=1;i<=10;i++){
	System.out.println(i);
	}
		//while loop
		int i=1;
		while(i<=10){
		System.out.println(i);
		i++;
		}
			//do-while loop
			int i=1;
			do{
			System.out.println(i);
			i++;
			}while(i<=10);

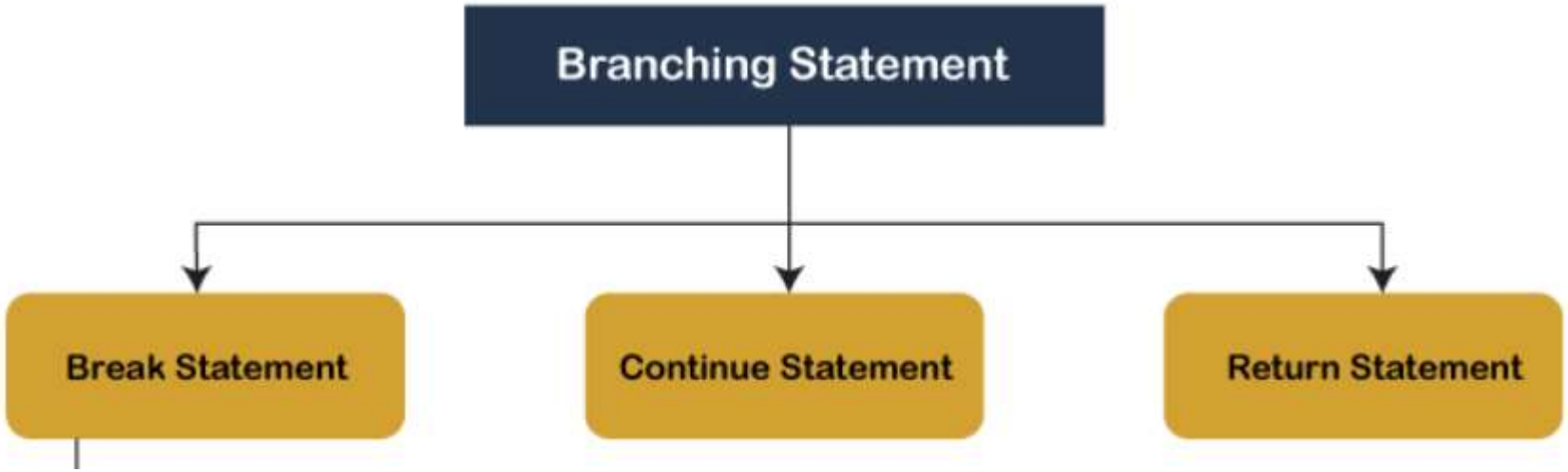
Syntax for infinitive loop	for(;;){
	//code to be executed
	}
		while(true){
		//code to be executed
		}
			do{
			//code to be executed
			}while(true);

Lecture 6

- Control Statements
- Decision Making
- Looping
- Branching
- Argument Passing Mechanism
- Command Line Argument.

Branching Statements

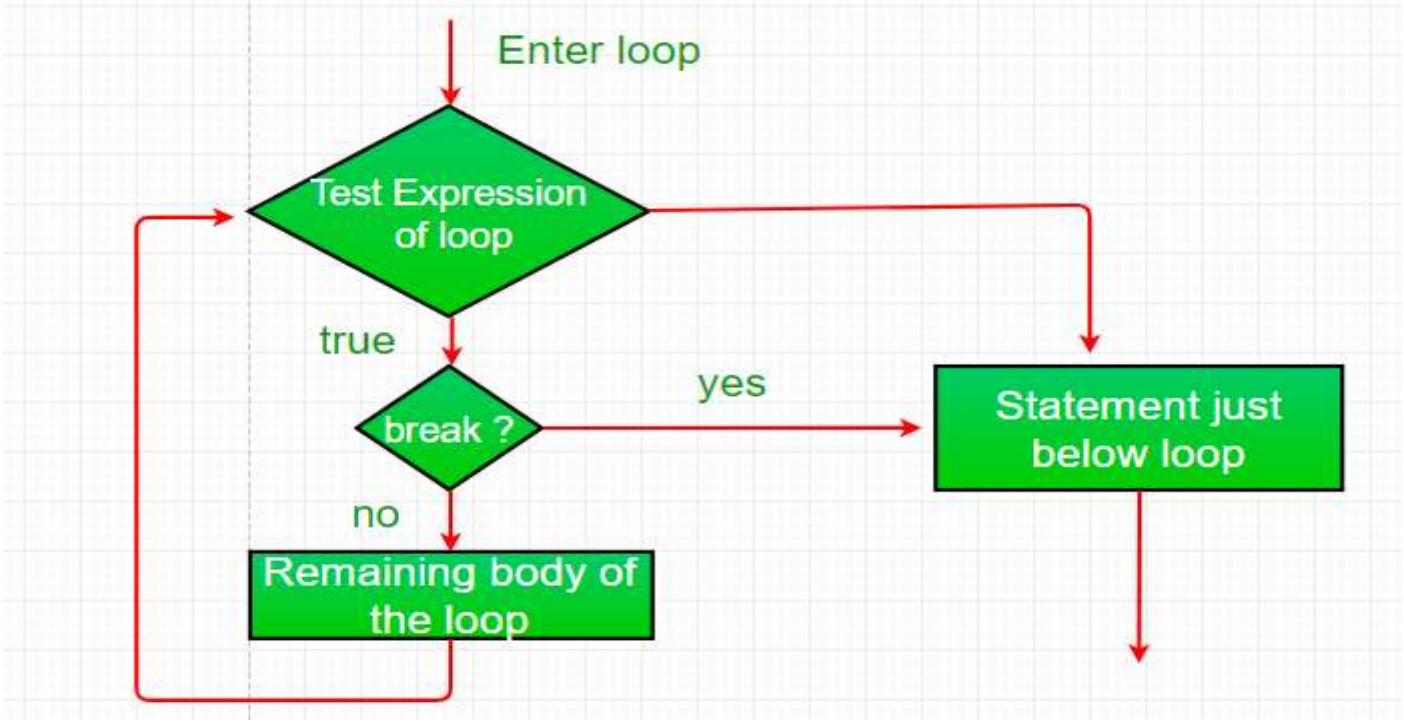
- **Branching statements** are the statements used to jump the flow of execution from one part of a program to another.
- The **branching statements** are mostly used inside the control statements.



Branching Statements

The break Statement

- Using break, we can force immediate termination of a loop, bypassing the conditional expression and any remaining code in the body of the loop.



Branching Statements

Java Break Statement

When a break statement is encountered inside a loop, the loop is immediately terminated and the program control resumes at the next statement following the loop.

The Java *break* statement is used to break loop or switch statement. It breaks the current flow of the program at specified condition. In case of inner loop, it breaks only inner loop.

We can use Java break statement in all types of loops such as for loop, while loop and do-while loop.

Syntax:

1.jump-statement;
2.break;

Branching Statements

```
1.//Java Program to demonstrate the use of break statement
2.//inside the for loop.
3.public class BreakExample {
4.public static void main(String[] args) {
5.    //using for loop
6.    for(int i=1;i<=10;i++){
7.        if(i==5){
8.            //breaking the loop
9.            break;
10.        }
11.        System.out.println(i);
12.    }
13.}
14.}
```

Branching Statements

Java Break Statement with Labeled For Loop

We can use break statement with a label. The feature is introduced since JDK 1.5. So, we can break any loop in Java now whether it is outer or inner loop.

```
1.public class BreakExample3 {
2.public static void main(String[] args) {
3.          aa:
4.          for(int i=1;i<=3;i++){
5.                  bb:
6.                  for(int j=1;j<=3;j++){
7.                      if(i==2&&j==2){
8.                          break aa;
9.                      }
10.                       System.out.println(i+" "+j);
11.                   }
12.           }
13.}  }
```

Branching Statements

Java Break Statement in while loop

```
1.public class BreakWhileExample {
2.public static void main(String[] args) {
3.    //while loop
4.    int i=1;
5.    while(i<=10){
6.        if(i==5){
7.            //using break statement
8.            i++;
9.            break;//it will break the loop
10.       }
11.       System.out.println(i);
12.       i++;
13.   }
14.}
15.}
```

Branching Statements

Java Break Statement in do-while loop

```
1.public class BreakDoWhileExample {
2.public static void main(String[] args) {
3.    int i=1;
4.    do{
5.        if(i==5){
6.            //using break statement
7.            i++;
8.            break;//it will break the loop
9.        }
10.         System.out.println(i);
11.         i++;
12.    }while(i<=10);
13.}
14.}
```

Conti..

Java Break Statement with Switch

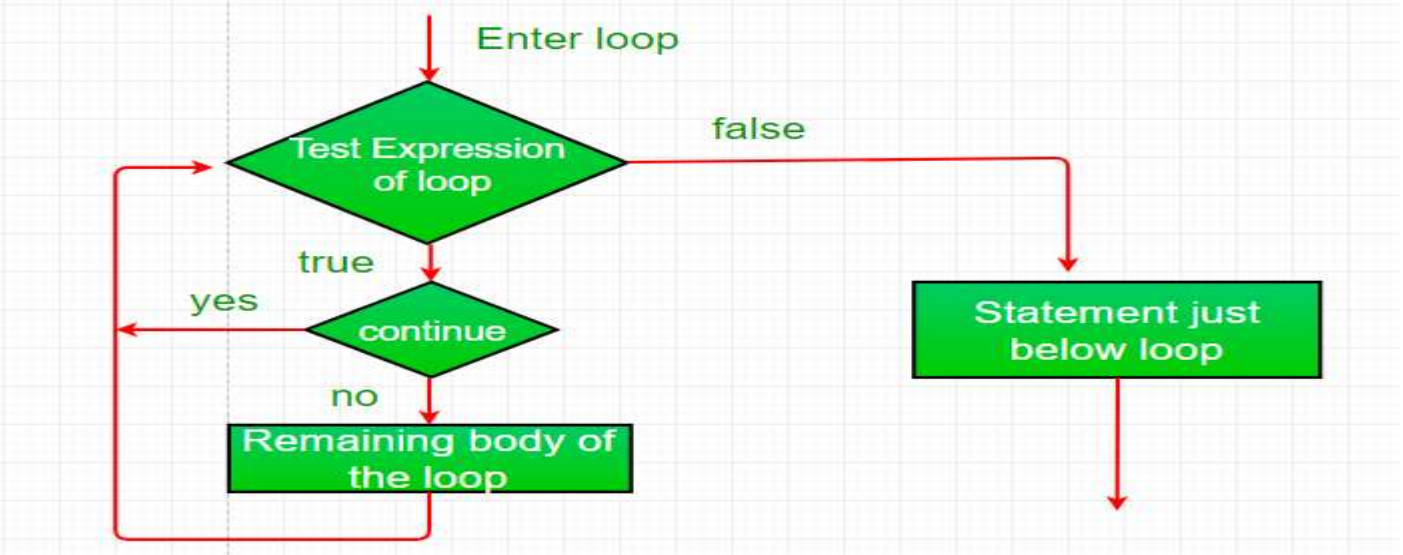
To understand the example of break with switch statement, please visit Switch slide.

1.switch(expression){
2.case value1:
3. //code to be executed;
4. **break**; //optional
5.case value2:
6. //code to be executed;
7. **break**; //optional
8.....
9.
10.default:
11. code to be executed **if** all cases are not matched;
12.}

Branching Statements

Continue:

- Sometimes it is useful to force an early iteration of a loop. That is, you might want to continue running the loop but stop processing the remainder of the code in its body for this particular iteration.



Branching Statements

Java Continue Statement

The continue statement is used in loop control structure when you need to jump to the next iteration of the loop immediately. It can be used with for loop or while loop.
The Java *continue statement* is used to continue the loop. It continues the current flow of the program and skips the remaining code at the specified condition. In case of an inner loop, it continues the inner loop only.
We can use Java continue statement in all types of loops such as for loop, while loop and do-while loop.

Syntax:

```
1.jump-statement;
2.continue;
```

Branching Statements

```
1.//Java Program to demonstrate the use of continue statement
2.//inside the for loop.
3.public class ContinueExample {
4.public static void main(String[] args) {
5.    //for loop
6.    for(int i=1;i<=10;i++){
7.        if(i==5){
8.            //using continue statement
9.            continue;//it will skip the rest statement
10.        }
11.        System.out.println(i);
12.    }
13.}
14.}
```

Branching Statements

Java Continue Statement with Inner Loop

It continues inner loop only if you use the continue statement inside the inner loop.

```
1. public class ContinueExample2 {
2.public static void main(String[] args) {
3.          //outer loop
4.          for(int i=1;i<=3;i++){
5.                  //inner loop
6.                  for(int j=1;j<=3;j++){
7.                      if(i==2&&j==2){
8.                          continue;
9.                      }
10.                       System.out.println(i+" "+j);
11.                   }
12.           }
13.}
14.}
```

Conti…….

Java Continue Statement with Labelled For Loop

We can use continue statement with a label. This feature is introduced since JDK 1.5. So, we can continue any loop in Java now whether it is outer loop or inner.

```
1.public class ContinueExample3 {
2.public static void main(String[] args) {
3.          aa:
4.          for(int i=1;i<=3;i++){
5.                  bb:
6.                  for(int j=1;j<=3;j++){
7.                      if(i==2&&j==2){
8.                          continue aa;
9.                      }
10.                       System.out.println(i+" "+j);
11.                   }
12.         }
13.}
14.}
```

Branching Statements

Java Continue Statement in while loop

```
1.public class ContinueWhileExample {
2.public static void main(String[] args) {
3.    int i=1;
4.    while(i<=10){
5.        if(i==5){
6.            i++;
7.            continue;//it will skip the rest statement
8.        }
9.        System.out.println(i);
10.         i++;
11.    }
12.}
13.}
```

Branching Statements

Java Continue Statement in do-while Loop

```
1.public class ContinueDoWhileExample {
2.public static void main(String[] args) {
3.    //declaring variable
4.    int i=1;
5.    do{
6.        if(i==5){
7.                i++;
8.            continue;//it will skip the rest statement
9.        }
10.        System.out.println(i);
11.        i++;
12.    }while(i<=10);
13.}
14.}
```

Branching Statements

Java Continue Statement in do-while Loop

```
1.public class ContinueDoWhileExample {
2.public static void main(String[] args) {
3.    //declaring variable
4.    int i=1;
5.    do{
6.        if(i==5){
7.                i++;
8.            continue;//it will skip the rest statement
9.        }
10.        System.out.println(i);
11.        i++;
12.    }while(i<=10);
13.}
14.}
```

Break vs Conti…..

BASIS FOR COMPARISON	BREAK	CONTINUE
Task	It terminates the execution of remaining iteration of the loop.	It terminates only the current iteration of the loop.
Control after break/continue	'break' resumes the control of the program to the end of loop enclosing that 'break'.	'continue' resumes the control of the program to the next iteration of that loop enclosing 'continue'.
Causes	It causes early termination of loop.	It causes early execution of the next iteration.
Continuation	'break' stops the continuation of loop.	'continue' do not stops the continuation of loop, it only stops the current iteration.
Other uses	'break' can be used with 'switch', 'label'.	'continue' can not be executed with 'switch' and 'labels'.

Branching Statements

Return:

- The return statement is used to explicitly return from a method. That is, it causes a program control to transfer back to the caller of the method.

Argument Passing Mechanism

- Arguments in Java are always **passed-by-value**.

- During method invocation, a copy of each argument, whether its a value or reference, is created in stack memory which is then passed to the method.

- When we pass an object, the reference in stack memory is copied and the new reference is passed to the method.

Command Line Argument

- Command line argument is a parameter supplied to the program when it is invoked.
- Command line argument is an important concept in programming.
- It is mostly used when you need to control your program from outside.
- Command line arguments are passed to the main() method.

Command Line Argument

To pass command line arguments, we typically define main() with two arguments : first argument is the number of command line arguments and second is list of command-line arguments.

```
int main(int argc, char *argv[]) { /* ... */ }
```

- **argc (ARGument Count)** is int and stores number of command-line arguments passed by the user including the name of the program. So if we pass a value to a program, value of argc would be 2 (one for argument and one for program name).The value of argc should be non negative.
- **argv(ARGument Vector)** is array of character pointers listing all the arguments.
- If argc is greater than zero,the array elements from argv[0] to argv[argc-1] will contain pointers to strings.

Conti….

```
1.class CommandLineExample{
2.public static void main(String args[]){
3.System.out.println("Your first argument is: "+args[0]);
4.}
5.}
```

Daily Quiz

1. Which of the following is not OOPS concept in Java? [CO1]

a) Inheritance
b) Encapsulation
c) Polymorphism
d) Compilation

2. Which of the following is a type of polymorphism in Java?[CO1]

a) Compile time polymorphism
b) Execution time polymorphism
c) Multiple polymorphism
d) Multilevel polymorphism

3. Which concept of Java is a way of converting real world objects in terms of class?[CO1]

a) Polymorphism
b) Encapsulation
c) Abstraction
d) Inheritance

Daily Quiz

4. Which of the following is not an OOPS concept?[CO1]
a. Encapsulation
b.Polymorphism
c.Exception
d.Abstraction
5. Which feature of OOPS described the reusability of code?[CO1]
a. Abstraction
b.Encapsulation
c.Polymorphism
d.Inheritance
6.Which feature of OOPS derives the class from another class?[CO1]
a. Inheritance
b.Data hiding
c.Encapsulation
d.Polymorphism

Daily Quiz

7. An IF-ELSE statement is also called ___.[CO1]

a. Branching statement

b. Control statement

c. Block statements

d. All

8.Which of these jump statements can skip processing the remainder of the code in its body for a particular iteration?[CO1]

a. Break

b.Return

c.Exit

d.continue

Daily Quiz

9. Every loop in Java has a condition that should be ___ in order to proceed for execution. [CO1]

A) FALSE

B) TRUE

10. What is a Loop in Java programming language?[CO1]

A) A Loop is a block of code that is executed repeatedly as long as a condition is satisfied.

B) A Loop is a block of code that is executed only once if the condition is satisfied.

C) A Loop is a block of code that is executed more than 2 times if the condition is satisfied.

D) None

Weekly Assignment

Q1. State some of object oriented technique strategies.[CO1]
Q2.Draw the class diagram of Online Shopping System.[CO1]
Q3.Discuss Inheritance and polymorphism with Example.[CO1]
Q4.Differentiate between Break and Continue statement.[CO1]
Q5.Describe Object Diagram with suitable example.[CO1]
Q6.Define if-else-if ladder with short program.[CO1]
Q7.Describe the control flow statement and types of control flow statements.[CO1]
Q8.Explain branching statements in Java with suitable examples.[CO1]
Q9. Explain Encapsulation with example.[CO1]
Q10. Draw class diagram of online library management system.[CO1]

Topic Links

- https://www.youtube.com/watch?v=r59xYe3Vyks&list=PLS1QulWo1RIbfTjQvTdj8Y6yyq4R7g-Al

- https://www.digimat.in/nptel/courses/video/106105191/L01.html

MCQs

1. Which of the following is not an OOP concept?[CO1]
a) Encapsulation
b) Polymorphism
c) Exception
d) Abstraction

Ans C

2. A single program of OOPS contains _____ classes?[CO1]
a) Only 1
b) Only 999
c) Only 100
d) Any number

Ans D

3. The while loop repeats a set of code while the condition is not met?[CO1]
a)True
b) False

Ans B

4. Which of the following is used with the switch statement?[CO1]
a) Continue
b) Exit
c) break
d) do

Ans C

5. Which of the following is not a valid flow control statement?[CO1]
a) exit()
b) break
c) continue
d) return

Ans A

6. Can command line arguments be converted into int automatically if required?[CO1]
a) Yes
b) No

Ans B

MCQs

Q7. Which of these are selection statements in Java?[CO1]

a) **if()**
b) for()
c) continue
d) break

Q8. Which of this statement is incorrect?[CO1]

a) switch statement is more efficient than a set of nested ifs
b) two case constants in the same switch can have identical values
c) switch statement can only test for equality, whereas if statement can evaluate any type of Boolean expression
d) it is possible to create a nested switch statements

Q9. Which of these selection statements test only for equality?[CO1]

a) if
b) switch
c) if & switch
d) none of the mentioned

MCQs

Q10. switch(x){ default: System.out.println(“Hello”); }

Which of the following is acceptable types for x? [CO1]

(i) byte and char

(ii) char and float

(iii) float and double

(iv) double and int

Q11.An IF statement in Java is also a ___ statement.[CO1]

A) Boolean

B) conditional

C) iterative

D) Optional

Q12.What is the output of the Java program with IF-ELSE statements?[CO1]

if(TRUE) System.out.println("GO"); else System.out.println("STOP");

A) GO

B) STOP

C) Compiler error

D) None

MCQs

Q13.What is the output of the Java program?[CO1]

```
int a=10; if(a==9)
System.out.println("OK ");
System.out.println("MASTER"); else System.out.println("BYE");
```

A) OK MASTER

B) BYE

C) Compiler error

D) None

Q14.What is the output of the Java program with IF-ELSE-IF statements?[CO1]

```
int marks=55; if(marks >= 80)
 System.out.println("DISTINCTION");
else if(marks >=35)
System.out.println("PASS");
else
System.out.println("FAIL");
```

MCQs

A) DISTINCTION

B) PASS

C) FAIL

D) Compiler error

Q15.What is the output of Java program below?[CO1]

```
float temp = 98.4f;
if(temp > 98.4)
 {
 System.out.println("SUMMER");
 }
Else
 { System.out.println("UNKNOWN");
}
```

MCQs

A) SUMMER

B) UNKNOWN

C) Compiler error

D) None

Q16. What is the output of the Java program below?[CO1]

if(3>1) { 4; }

A) 0

B) 4

C) Compiler error

D) None

Glossary Questions

1. **Attempt all the parts. Pick the correct option from glossary.** [CO1]

i) Abstraction ii) Encapsulation iii) Inheritance iv) Polymorphism

a) _____ means one entity have different behaviours in different situations.

b) _____ is used to bind the code and data.

c) _____ is used to hide the necessary details from outside world.

d) _____ is used to acquire properties of one class into other.

2. **Attempt all the parts. Pick the correct option from glossary.** [CO1]

i) Super class ii) Sub class iii) Reusability iv) Redundant code

a) Inheritance can eliminate _____.

b) A parent class is also known as _____ .

c) Using an existing code is known as _____.

d) A child class is also known as _____ .

Sessional Paper-1

Printed page: 2 **Subject Code: ACSE0302**

Roll No: | | | | | | | | | | | | |

NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY ,GREATER NOIDA

(An Autonomous Institute Affiliated to AKTU, Lucknow)

B. Tech (AI / AIML / IOT / DS)

(SEM:III SESSIONAL EXAMINATION – I)(2021-2022)

Subject Name: OBJECT ORIENTED TECHNIQUES USING JAVA

Time: 1.15 Hours **Max. Marks:30**

General Instructions:

- All questions are compulsory. Answers should be brief and to the point.
- This Question paper consists of 2 pages & 5 questions.
- It comprises of three Sections, A, B, and C. You are to attempt all the sections.
- Section A -Question No- 1 is objective type questions carrying 1 mark each, Question No- 2 is very short answer type carrying 2 mark each. You are expected to answer them as directed.
- Section B - Question No-3 is short answer type questions carrying 5 marks each. You need to attempt any two out of three questions given.
- Section C - Question No. 4 &5are Long answer type (within unit choice) questions carrying 6marks each. You need to attempt any one part a or b.
- Students are instructed to cross the blank sheets before handing over the answer sheet to the invigilator.
- No sheet should be left blank. Any written material after a blank sheet will not be evaluated/checked.

		SECTION – A	[8]	
1.	Attempt all parts.		(4×1=4)	CO
	a.	The type of Arguments the main method accepts is ____. a) integer[ ] b) String c) float[ ] d) String[ ]	(1)	CO2
	b.	The default value for data field of a boolean type, numeric type is _____ , _____ respectively. a) false, 1 b) true, 0 c) false, 0 d) true, 1	(1)	CO1
	c.	Using _____, we can force immediate termination of a loop. a) break b) continue c) return d) goto	(1)	CO1
	d.	The _____ statement is used to explicitly return from a method. a) break b) continue c) return d) goto	(1)	CO2
2.	Attempt all parts.		(2×2=4)	CO
	a.	Write a JAVA program to check whether the number entered by user is even or odd by using if-else. (Use the Scanner class to enter the integer	(2)	CO1

Conti….

		number)		

	b.	What is the output of the following Java code snippet?
		int i=0;
		for(i=1; i <= 6; i++)
		{
		if (i % 3 = = 0)
		continue;
		System.out.print (i+",");
		}
			(2)	CO1

		SECTION – B		
3.	Answer any two of the following-		[2×5=10]	CO
	a.	Draw a class diagram of Student class and explain how the access modifiers are represented in the class diagrams?	(5)	CO1
	b.	Discuss different levels of access specifiers available in JAVA.	(5)	CO1
	c.	What is the purpose of constructors? Explain all the types of constructors in JAVA with the help of example of your choice.	(5)	CO2
		SECTION – C		
4	Answer any one of the following-(Any one can be applicative if applicable)		[2×6=12]	CO
	a.	Explain the four pillars of Object-Oriented Programming with the help of examples.	(6)	CO1
	b.	Write a JAVA program to display all even numbers from 100 to 50 using all the loops statements you have studied.	(6)	CO1
5.	Answer any one of the following-			
	a.	Write a program in JAVA that takes arguments name, department and marks of 4 subjects from the user and then print total and average marks obtained. (Use command line arguments for giving input).	(6)	CO1
	b.	Write a JAVA program to display the reverse of an input number. (Use Scanner class to input the positive integer)	(6)	CO1

Sessional Paper-2

Printed page: 2 **Subject Code:ACSE0302**

Roll No: | | | | | | | | | | | | | |

NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY, GREATER NOIDA

(An Autonomous Institute)

Affiliated to Dr. A.P. J. Abdul Kalam Technical University, Uttar Pradesh, Lucknow

Course ...B.Tech.....Branch...IT....

Semester.....III...Sessional Examination.....II.....Year- (2021 - 2022)

Subject Name: OBJECTS ORIENTED TECHNIQUES USING JAVA

Time: 1.15Hours **[SET- A]** **Max. Marks:30**

General Instructions:

- This Question paper consists of 2 pages & 5 questions.It comprises of three Sections, A, B, and C
- Section A -Question No- 1 is objective type questions carrying 1 mark each, Question No- 2 is very short answer type carrying 2 mark each. You are expected to answer them as directed.
- Section B - Question No-3 is Short answer type questions carrying 5 marks each. Attempt any two out of three questions given.
- Section C -Question No. 4 &5are Long answer type (within unit choice) questions carrying 6marks each. Attempt any one part *a* or *b*.

Conti….

		SECTION – A	[08Marks]	
1.	All questions are compulsory		(4×1=4)	
	a.	Identify the correct way to call the constructor of class "Super" that is inherited by the class "Child". (i) class Child extends Super{ Child(){ Super();}} (ii) class Child extends Super{ Child(){ super();}} (iii) class Child extends Super{ Child() { super.Super();}} (iv) All the ways are correct.	(1)	CO2
	b.	Total abstraction can be achieved using? (i) abstract class (ii) interface (iii) both (iv) total abstraction cannot be achieved	(1)	CO2
	c.	The …….. class inherits all the properties of the …… class? (i) base, derived (ii) derived base (iii) base, initial (iv) base, final	(1)	CO2
	d.	Runtime polymorphism is also known as: (i) Dynamic binding (ii) Static binding (iii) Early binding (iv) None of the above	(1)	CO2
2.	All questions are compulsory		(2×2=4)	
	a.	Explain a simple program showing garbage collection.	(2)	CO2
	b.	Explain the concept of interface using suitable example.	(2)	CO2

Conti..

SECTION – B			**[10Marks]**	
3.	**Answer any two of the following-**		**(2×5=10)**	
	a.	**Explain the types of polymorphism in java using suitable examples for each type.**	**(5)**	**CO2**
	b.	**Explain the difference between interface and abstract class in java using suitable example.**	**(5)**	**CO2**
	c.	**Explain inheritance in java. Explain all types of inheritance supported by java using suitable examples.**	**(5)**	**CO2**
SECTION – C			**[12Marks]**	
4	**Answer any one of the following-**		**(1×6=6)**	
	a.	**Explain the working of "this" and "super" keyword in java. Illustrate each using suitable example.**	**(6)**	**CO2**
	b.	**Explain abstract class. State the use of abstract class with suitable example. Also write the names of OOPs concepts that is used to implement abstract class and that is implemented using abstract class.**	**(6)**	**CO2**
5.	**Answer any one of the following-**		**(1×6=6)**	
	a.	**Explain the concept of access modifiers (public, private, protected) using packages.**	**(6)**	**CO3**
	b.	**Explain overloading and overriding of methods? Illustrate overloading and overriding of methods in Java with suitable examples.**	**(6)**	**CO2**

NOTE: No example should be repeated.

Old University Question Paper

Printed Page:-

Subject Code:- ACSE0302

Roll. No:

NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY, GREATER NOIDA

(An Autonomous Institute Affiliated to AKTU, Lucknow)

B.Tech.

SEM: III - THEORY EXAMINATION (2021 - 2022)

Subject: Object Oriented Techniques using Java

Time: 03:00 Hours

Max. Marks: 100

General Instructions:

1. All questions are compulsory. It comprises of three Sections A, B and C.

- Section A - Question No- 1 is objective type question carrying 1 mark each & Question No- 2 is very short type questions carrying 2 marks each.
- Section B - Question No- 3 is Long answer type - I questions carrying 6 marks each.
- Section C - Question No- 4 to 8 are Long answer type - II questions carrying 10 marks each.
- No sheet should be left blank. Any written material after a Blank sheet will not be evaluated/checked

Old University Question Paper

SECTION A 20

1. Attempt all parts:-

1-a. In JAVA main method returns value of type _____ [CO1] 1

1. float
2. int
3. void
4. String

1-b. What is bytecode in Java? [CO1] 1

1. Code generated by a Java compiler
2. Code generated by a Java Virtual Machine
3. Name of Java source code file
4. Block of code written inside a class

1-c. If same message is passed to objects of several different classes and all of those can respond in a different way, what is this feature called? [CO2] 1

1. Inheritance
2. Overloading
3. Polymorphism
4. Overriding

1-d. Java _____ is invoked at the time of object creation. [CO2] 1

1. constructor
2. class
3. method
4. array

1-e. Which of these keywords must be used to monitor for exceptions? [CO3] 1

1. try
2. catch
3. throw
4. finally

1-f. An _____ statement can be used to access the classes and interface of a different package from the current package. [CO3] 1

1. instanceOf
2. import
3. extends
4. implement

Old University Question Paper

1-e. Which of these keywords must be used to monitor for exceptions? [CO3] 1

1. try
2. catch
3. throw
4. finally

1-f. An _____ statement can be used to access the classes and interface of a different package from the current package. [CO3] 1

1. instanceOf
2. import
3. extends
4. implement

1-g. The keyword that is used to protect the methods from simultaneous access in Threads is _____ [CO4] 1

1. save
2. synchronized
3. Both
4. This task is not possible in threads

1-h. Which of these classes are used by Byte streams for input and output operation? [CO4] 1

1. Input Stream
2. InputOutputStream
3. Reader
4. All of the mentioned

1-i. A _____ dictates the style of arranging the components in a container. [CO5] 1

1. border layout
2. grid layout
3. panel
4. layout manager

1-j. _____ interface provides the capability to store objects using a key-value pair. [CO5] 1

1. Java.util.Map
2. Java.util.Set
3. Java.util.List
4. Java.util.Collection

Old University Question Paper

2. Attempt all parts:-

2-a.	What is JVM? [CO1]	2
2-b.	What is the use of final keyword in JAVA? [CO2]	2
2-c.	What is an assertion in Java? How is it different from if - else conditions? [CO3]	2
2-d.	Describe any two Annotations from the Java Standard Library. [CO4]	2
2-e.	What Are Wrapper Classes? Why do we need wrapper classes in JAVA? [CO5]	2

Old University Question Paper

SECTION B 30

3. Answer any five of the following:-

3-a.	What is the difference between object diagrams and class diagrams? Draw a class diagram of order management system. [CO1]	6
3-b.	How to take an input from a user with the help of scanner class in JAVA? Explain using JAVA code. [CO1]	6
3-c.	Explain Abstract class concept with an example program. [CO2]	6
3-d.	Compare overloading and overriding of methods in java using proper examples. [CO2]	6
3-e.	Write a method to check if input string is Palindrome? [CO3]	6
3-f.	Explain the concept of multithreading in java and explain how even and odd numbers can be printed by using multithreading concept. (CO4)	6
3-g.	Examine ArrayList with Example. [CO5]	6

Old University Question Paper

SECTION C 50

4. Answer any one of the following:-

4-a.	What are command line arguments? How are they useful? Write a program to compute the sum of the digits of an input number (Using command line arguments) eg if 4523 is an integer then the sum of digits displayed will be 14. [CO1]	10
4-b.	Write a JAVA program that takes values of name, age, department and marks of 4 subjects from the user. Display the name, total and average of marks computed. [CO1]	10

5. Answer any one of the following:-

5	Explain the following with respect to JAVA: [CO2] a) super keyword b) Garbage collection c) Interface d) Static data members e) final keyword	10
5	What is the lambda expression in Java and what are the features of a lambda expression? Briefly explain its use with the help of suitable example. [CO2]	10

Old University Question Paper

6. Answer any one of the following:-

6	Write the differences between String, StringBuffer and StringBuilder classes. With proper syntax, explain the following methods. [CO3] 1. Method to extract a particular character of a string. 2. Reverse a String.	10
6	What is the difference between an error and exception? Write the following Java program for illustrating the use of throw keyword. Write a class ThrowExample contains a method checkEligibilty(int age, int weight) which throw an ArithmeticException with a message "Student is not eligible for registration" when age < 12 and weight < 40, otherwise it prints "Student Entry is Valid!!". [CO3]	10

7. Answer any one of the following:-

7-a.	What is the difference between thread and a process? Explain the concept of Inter Thread Communication and describe the role of wait(), notify(), and notifyAll() methods in inter thread communication. [CO4]	10

Old University Question Paper

7-b. While reading a file, how would you check whether you have reached to the end of file? Write a JAVA program to copy the content of "file1.txt" to "file2.txt". [CO4] 10

8. Answer any one of the following:-

8-a. Discuss some general rules for using layout managers. Describe the various layout managers available in AWT. [CO5] 10

8-b. Differentiate between List and ArrayList. Create a class TestArrayList having main method. Perform following functionality. [CO5] 10

1. Create an ArrayList having fruits name of type String.
2. Store different fruit names. (Try to add duplicate fruit names).
3. Print all fruit names.
4. Print the first and last fruit names.
5. Print the size of ArrayList.
6. Remove a particular fruit from ArrayList.

Expected Questions

Q1. Make Object diagram of Library management system. [CO1]
Q2.Make the class diagram for e-commerce System. [CO1]
Q3.Elaborate the concept of Inheritance and polymorphism with suitable example. [CO1]
Q4.Give differences between Break and Continue statement with an example. [CO1]
Q5.Give example code illustrating the use of if-else-if . [CO1]
Q6. State the different OOT strategies. [CO1]
Q7 Differentiate between class diagram and object diagram. [CO1]
Q8 Explain command line arguments. [CO1]
Q9 Explain the important points to be remembered while drawing object diagrams. [CO1]
Q10 Explain the working of if-else-if ladder. [CO1]
Q11Explain the importance of using break in switch statement. [CO1]
Q12Explain while loop with its syntax and an example. [CO1]
Q13Explain a real life example of polymorphism.

Old Question Papers

- https://www.iare.ac.in/sites/default/files/IARE_JAVA_MODEL_QP.pdf
- https://www.manaresults.co.in/jntuh/download.php?subcode=133BM

Recap of Unit

- The main aim of OOP is to bind together the data and the functions that operate on them so that no other part of the code can access this data except that function.
- Object-oriented modeling and design is a way of thinking about problems using models organized around real world concepts.
- Decision-making statements evaluate the Boolean expression and control the program flow depending upon the result of the condition provided.

References

Text Books:

(1) Herbert Schildt," Java - The Complete Reference", McGraw Hill Education 12th edition

(2) Herbert Schildt," Java: A Beginner's Guide", McGraw-Hill Education 2nd edition

(3) James Rumbaugh et. al, "Object Oriented Modeling and Design", PHI 2nd Edition

Reference Books:

(4) Cay S. Horstmann, "Core Java Volume I – Fundamentals", Prentice Hall

(5) Joshua Bloch," Effective Java", Addison Wesley

(6) E Balagurusamy, "Programming with Java A Primer", TMH, 4th edition.

Thank You