

# Object Oriented Programming (OOP)



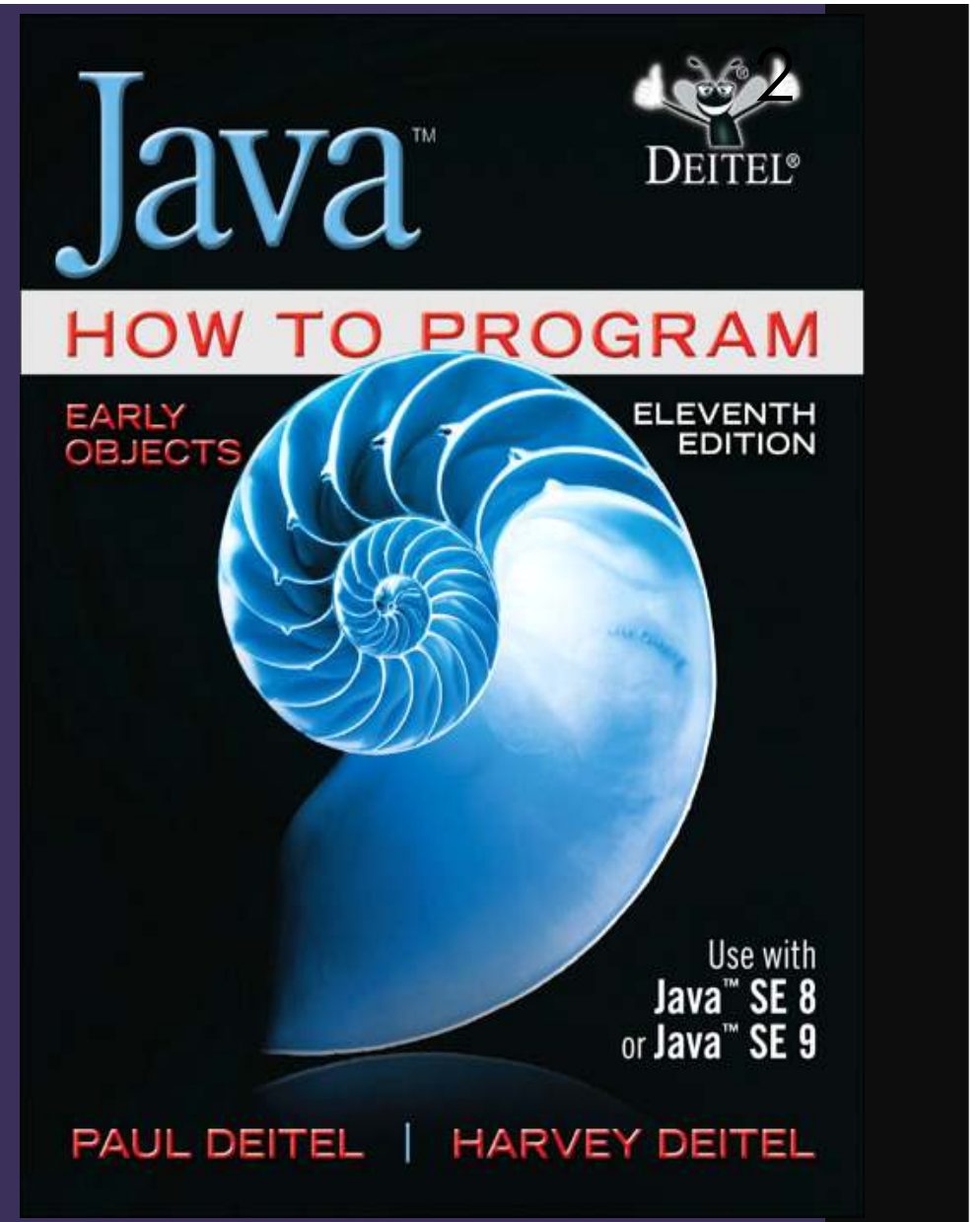
## Lecture1 : Introduction to OOP

Prepared by:  
Mohamed Mabrouk

Those slides are based on slides by:  
Dr. Sherin Mousa and Dr. Sally Saad

# Course Textbook

Java How to Program, Early Objects, 11th Edition, Pearson, 2018.



# Online Resources

- JDK 11 Documentation  
<https://docs.oracle.com/en/java/javase/11>
- Java Tutorial  
<https://www.tutorialspoint.com/java/index.htm>
- Java tutorial: Learn Java Programming with examples  
<https://beginnersbook.com/java-tutorial-for-beginners-with-examples/>

# Grading Policy

<b>Semester Work</b>	<b>15</b>
<b>Midterm Exam</b>	<b>15</b>
<b>Practical Exam</b>	<b>20</b>
<b>Final Exam</b>	<b>50</b>
<b>Total</b>	<b>100</b>

# Course Format

- One lecture/week
- One lab/week
- Programming Language is Java
- Mid term exam
- Project/Practical exam or both
- Final exam

# Course Outline

Week	Topics
1	<ol style="list-style-type: none"><li>1. Introduction to Java Language</li><li>2. Introduction to Object Oriented Programming</li><li>3. First program in Java</li></ol>
2	<ol style="list-style-type: none"><li>1. Control Structures</li><li>2. Arrays</li><li>3. Classes and objects</li><li>4. Introduction to Class Diagrams and UML : Unified Modelling Language</li><li>5. Encapsulation and access modifiers</li></ol>
3	<ol style="list-style-type: none"><li>1. Methods in Java</li><li>2. Strings in Java</li></ol>
4	<ol style="list-style-type: none"><li>1. Inheritance</li><li>2. Polymorphism</li></ol>

# Course Outline

Week	Topics
5	<ol style="list-style-type: none"><li>1. Exception Handling</li><li>2. Collections (Generics)</li></ol>
6	<ol style="list-style-type: none"><li>1. Graphical User Interface (GUI 1)</li></ol>
7	<ol style="list-style-type: none"><li>1. Interfaces</li><li>2. Event Handling (GUI 2)</li></ol>
8	Midterm exam
9	<ol style="list-style-type: none"><li>1. Design Patterns (Part 1)</li></ol>
10	<ol style="list-style-type: none"><li>1. Design Patterns (Part 2)</li></ol>

# Lecture Outline

- Introduction to Java language
- Introduction to object oriented programming
- First program in Java



# Introduction to Java Language

# Java Language History

- 1991 - Green Project for consumer electronics market (Oak language → Java)
- 1995 – Sun announces Java
- 1996 – JDK 1.0
- 1997 – JDK 1.1 RMI, AWT, Servlets
- 1998 – Java 1.2 Reflection, Swing, Collections
- 2004 – J2SE 1.5 (Java 5) Generics, enums
- 2014 – Java 8 Lambdas
- 2018 – Java 11 var keyword, improved garbage collection

# Java Technology

- JVM – Java Virtual Machine
- JRE – Java Runtime Environment
- JDK – Java Development Kit

**JDK**      javac, jar, debugging

**JRE**      Java, Libraries

**JVM**

# Java Virtual Machine (JVM)

- JVM is a virtual machine that runs Java bytecode
- JVM does not understand java → \*.java
- JVM only understands bytecode → \*.class
- Java files should be compiled into bytecode
- Several implementations of the JVM for different systems (Windows, Linux, macOS)

# Java Runtime Environment (JRE)

- According to Oracle:
  - “JRE provides the libraries, the Java Virtual Machine, and other components to run applets and applications written in the Java programming language.”
- JRE does not contain tools and utilities such as compilers or debuggers for developing applets and applications.

# Java Development Kit(JDK)

- According to Oracle:  
“The JDK is a superset of the JRE, and contains everything that is in the JRE, plus tools such as the compilers and debuggers necessary for developing applets and applications.”

# Properties of Java

- Object-oriented
- Interpreted
- Portable
- Secure and robust
- Multi-threaded
- Garbage collected
- No support for multiple inheritance

# Hello World Application

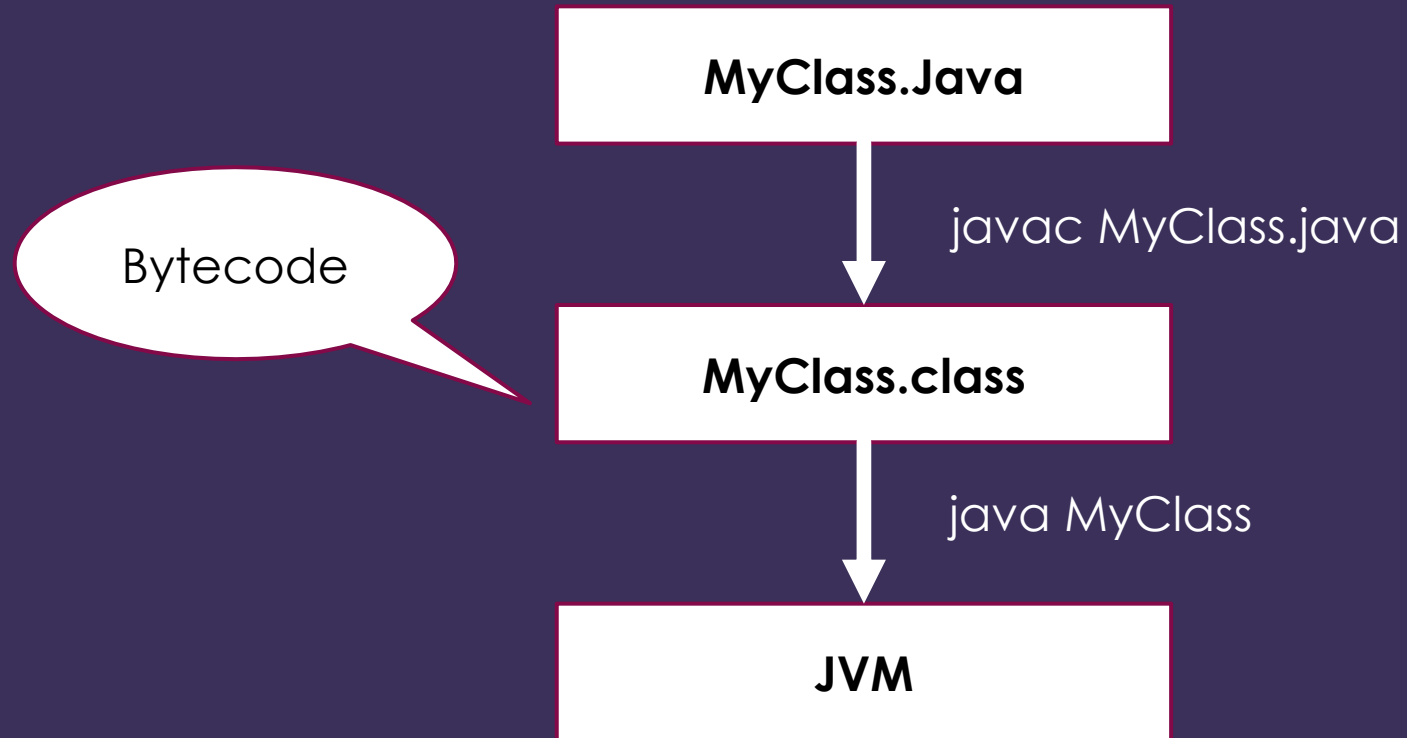
- Write the following source code and name the file MyClass.java

```
public class MyClass {  
    public static void main(String[] args){  
        System.out.println("Hello World")  
    }  
}
```

- Compile it using: javac MyClass.java
- Run it using: java MyClass



# Hello World Application



# Garbage Collection

- Memory is dynamically allocated in Java
- Deallocation is removing the objects that are no longer referenced from memory
- Programmer responsibility → C, C++
  - Might lead to memory leaks
- Garbage collector responsibility → Java, C#
  - Theoretically no memory leaks

# Introduction to Object Oriented Programming

# Object Oriented Thinking

- It is programming paradigm based on the concept of "objects"
- Each object contains data, in the form of fields, often known as attributes; and actions to work on that data, in the form of procedures, often known as methods
- Objects constitute the building blocks of the program

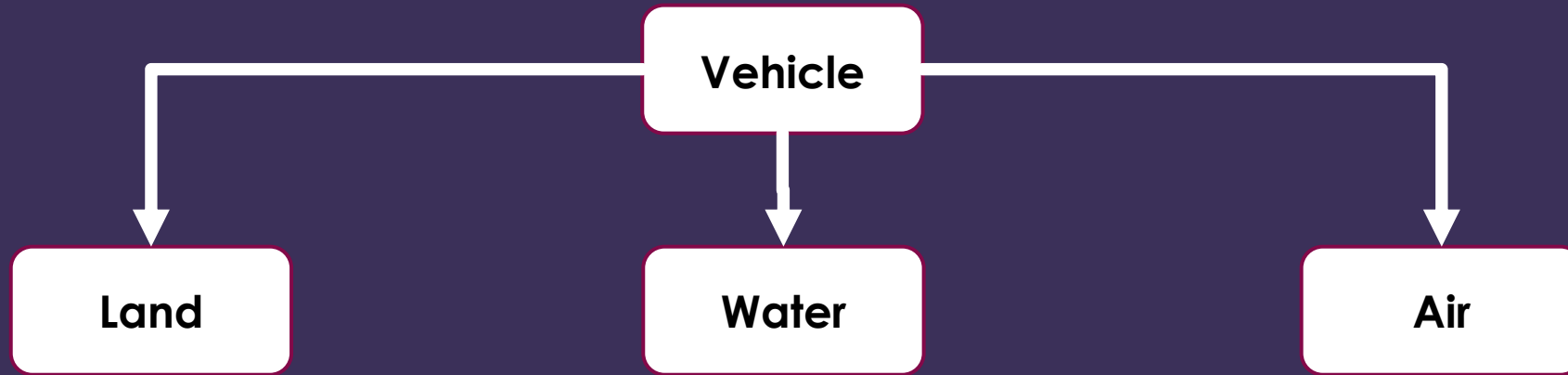
# Object Oriented Thinking

- Objects interact with each other and exchange data
- Objects of the same type constitute "class", e.g. person, and car
- Classes can form a hierarchy

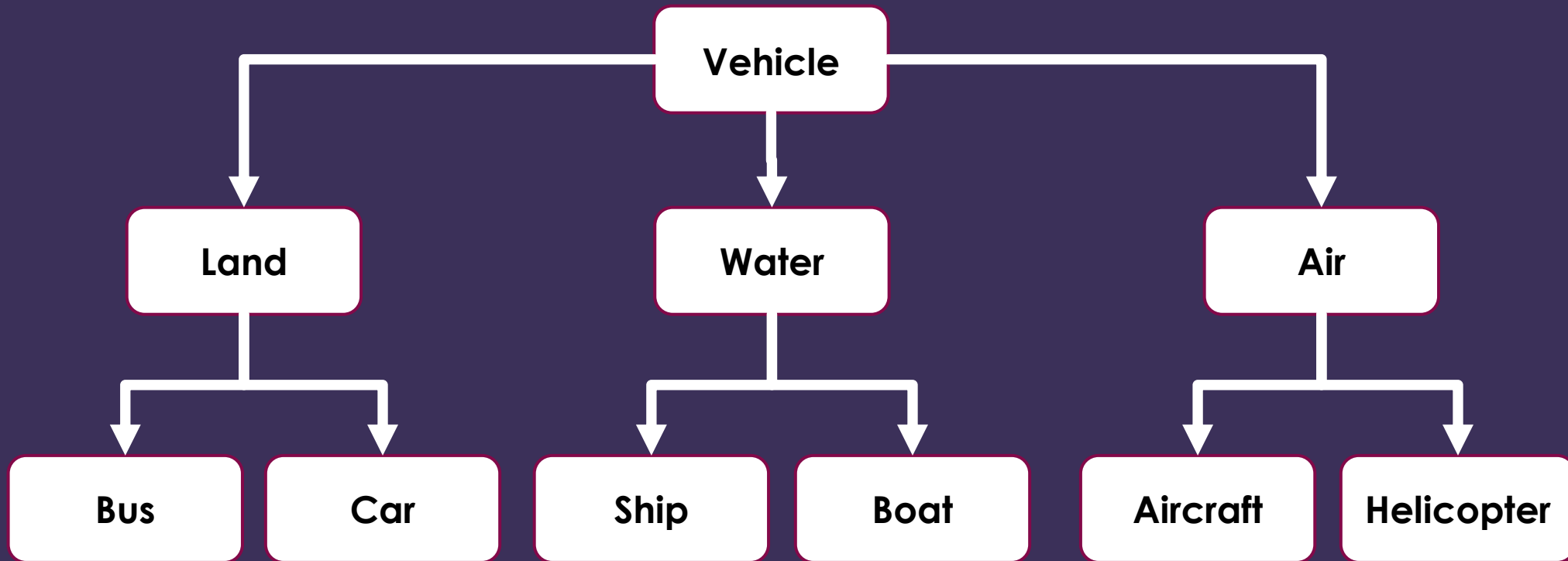
# Class Hierarchy Example

**Vehicle**

# Class Hierarchy Example



# Class Hierarchy Example





# Name Some Classes of Objects

- Think of some classes of objects
- What are the data and methods of each class
- Can you create a simple hierarchy



# Principles of Object Oriented Programming

# Principles of Object Oriented Programming

- Encapsulation

# Principles of Object Oriented Programming

- Encapsulation
- Abstraction

# Principles of Object Oriented Programming

- Encapsulation
- Abstraction
- Inheritance

# Principles of Object Oriented Programming

- Encapsulation
- Abstraction
- Inheritance
- Polymorphism

# Principles of Object Oriented Programming

- Encapsulation
- Abstraction
- Inheritance
- Polymorphism



Never forget

# Encapsulation

- Is the process of combining data and methods into a single unit called a class
- Keeps data both safe from outside interference
- Allows to expose only necessary data
- Can be controlled via access modifiers; `private`, `protected`, and `public`



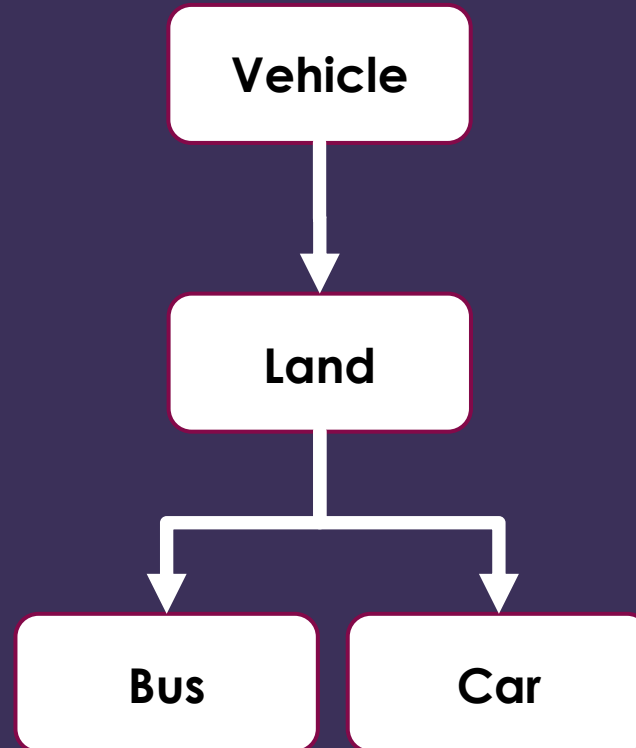
# Abstraction

- Is to hide the complexity of the program by hiding unnecessary detail from the user
- Can be achieved by using classes

# Inheritance

- One of the most important features of OOP
- Allows creating hierarchy of related classes, e.g. Vehicle, car and bus
- Allows code reusability

# Inheritance



# Polymorphism

- Means many forms
- Same concept can have different meanings in different contexts
- Has two forms, overloading and overriding
- Overloading is same method name with different parameters, e.g.

```
int add(int num1, int num2) {}
```

and

```
float add(float num1, float num2) {}
```

# Polymorphism

- Overriding is related to inheritance
- It allows overriding an inherited method by creating your own method
- The method of the subclass **MUST** match the one in super class, i.e. it must have the exact same name, same parameters and same return type

# First Program in Java

# Hello World Application

```
public class MyClass {  
    public static void main(String[] args){  
        System.out.println("Hello World");  
    }  
}
```

- There must be EXACTLY one main method
- It has to be defined inside a class
- It has to have the signature  
`public static void main(String[] args)`

# Hello World Application

- The java filename must match the class name, i.e. `MyClass.java`
- Each statement has to be terminated by a semicolon “;”
- Code can be compiled via command  
`javac MyClass.java`
- Can be run via command  
`java MyClass`



# Hello World Application

- Better to use an Integrated Development Environment (IDE)
- You can use  
Eclipse  
NetBeans  
IntelliJ



# Simple Calculator Application

- Create an application to read two numbers, add them and display the result on the screen
- To display to the console:  
`System.out.println()`
- To read from the console:  
`Scanner in = new Scanner(System.in);`  
`in.nextInt()`

Thank You!