

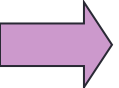
OBJECT-ORIENTED LANGUAGE AND THEORY

# 1. INTRODUCTION TO OBJECT TECHNOLOGY

---

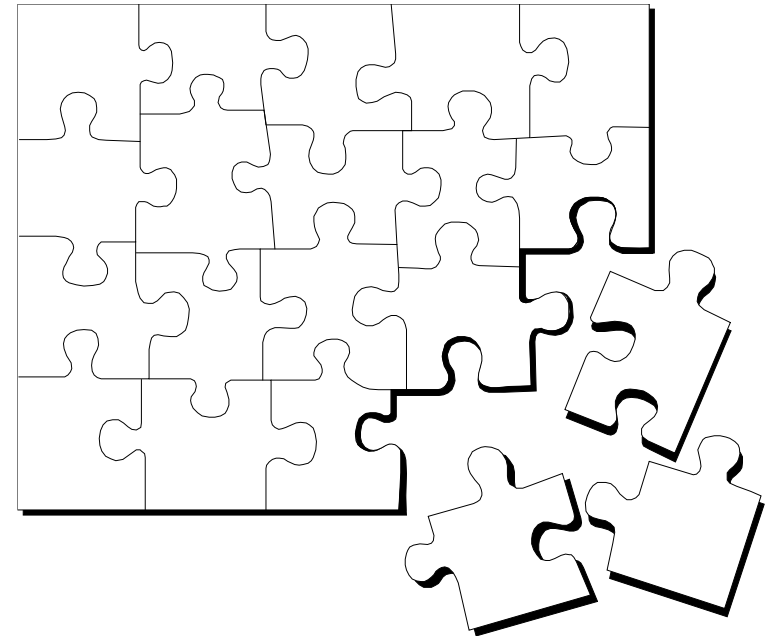


# Outline

- 
1. Object-Oriented Technology
  2. Object and Class
  3. Java programming language
  4. Examples and Exercises

# 1.1 Object Technology

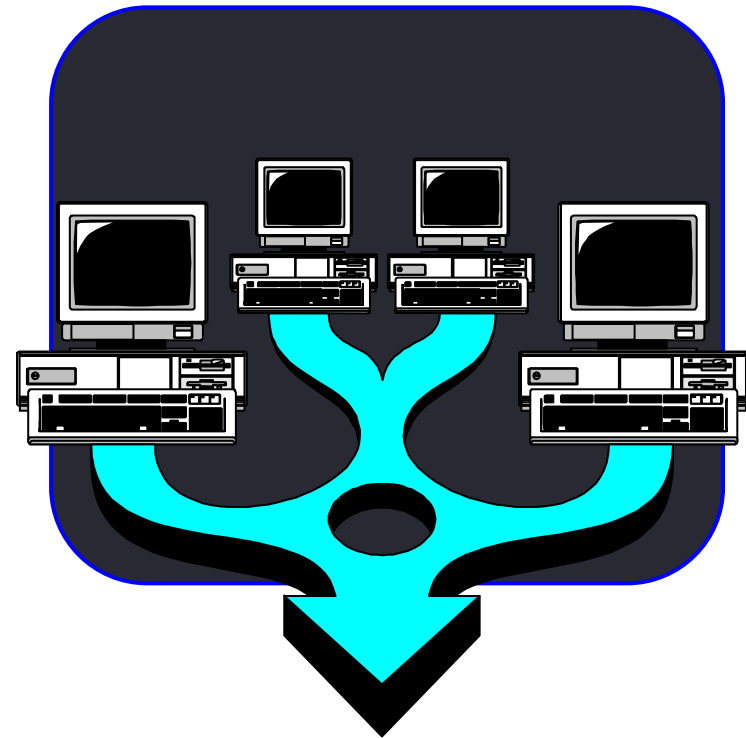
- Object technology is a set of rules (abstraction, encapsulation, polymorphism), instructions to build a software, together with languages, databases and other tools to support these rules.



*(Object Technology - A Manager's Guide, Taylor, 1997)*

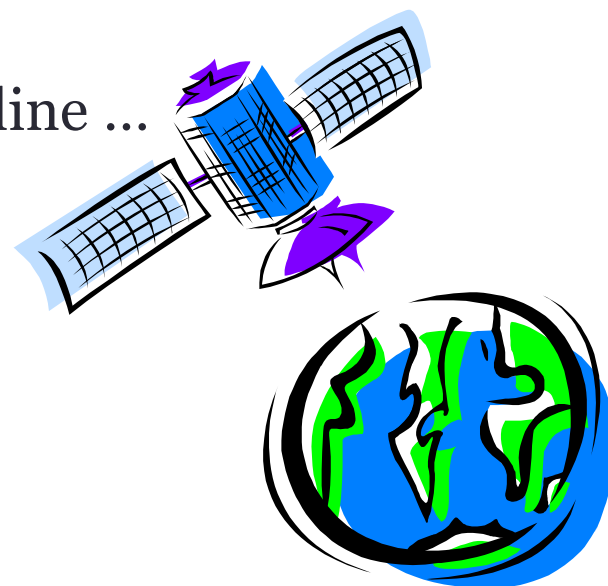
## 1.2 Where is the object technology used?

- Client/Server Systems and Web development
  - Object technology allows companies to encapsulate information in objects and to distribute its computation/processing via Internet or via a network.



## 1.2. Where is the object technology used? (2)

- Mobile development (Android)
- Embedded system
- Real-time systems
  - Object technology allows real-time systems to be developed with higher quality and in a more flexible way
    - Satellite systems
    - Defense systems and space airline ...



# The power of the object technology

- Allow re-using source code and architectures
- Reflecting more closely the real world
- More stable, a system change is done in a small part of the system
- More adaptable with changes

# Milestones of the object technology

Simula



1967

C ++



Late 1980s

The UML



1996

1972



Smalltalk

1991



**Java**

2004



UML 2

## 1.3 Evolution of programming languages

- Assembly language
- Structure/Procedure programming languages
  - Pascal, C
- Object programming languages
  - C++, Java, C#.NET, Python...



## a. Assembly language

**Assembly  
code**

```

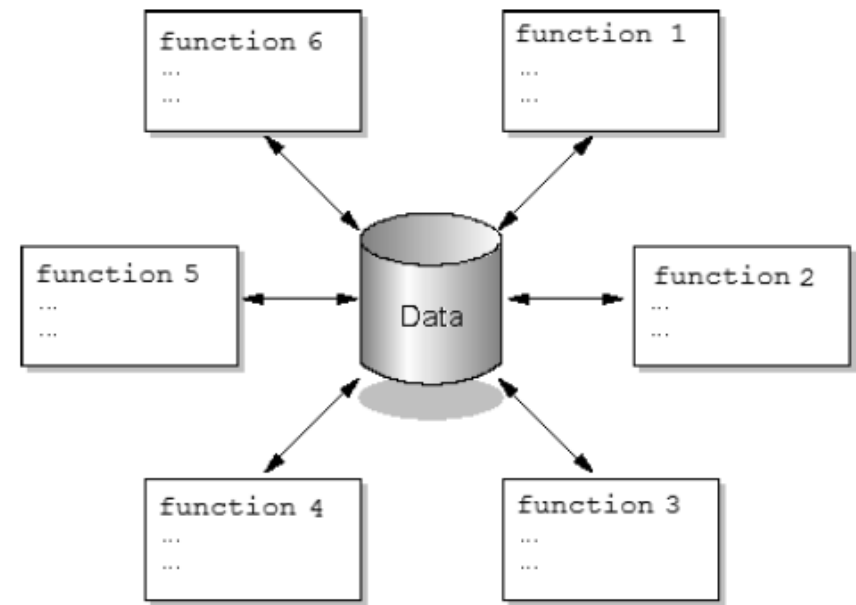
;CLEAR SCREEN USING BIOS
CLR: MOV AX,0600H      ;SCROLL SCREEN
      MOV BH,30        ;COLOUR
      MOV CX,0000      ;FROM
      MOV DX,184FH     ;TO 24,79
      INT 10H          ;CALL BIOS;
;INPUTTING OF A STRING
KEY:  MOV AH,0AH       ;INPUT REQUEST
      LEA DX,BUFFER    ;POINT TO BUFFER WHERE STRING STORED
      INT 21H          ;CALL DOS
      RET              ;RETURN FROM SUBROUTINE TO MAIN PROGRAM;
; DISPLAY STRING TO SCREEN
SCR:  MOV AH,09        ;DISPLAY REQUEST
      LEA DX,STRING    ;POINT TO STRING
      INT 21H          ;CALL DOS
      RET              ;RETURN FROM THIS SUBROUTINE;

```

- Is a sequence programming language, is very close to machine codes of CPU.
- Hard to remember, to write, especially for complex systems.
- Hard to fix, to maintain.

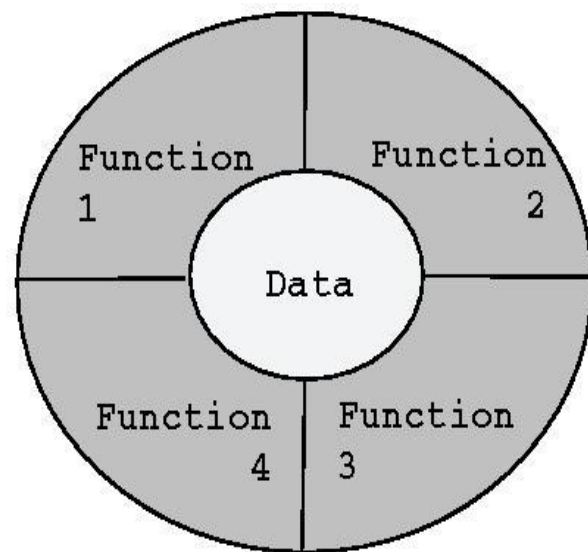
## b. Structure/Procedure programming languages

- Build a program based on functions/procedures/sub-programs
- Data and data processing unit (functions) are separate
- Functions are not forced to follow a common rule for accessing data



## c. Object programming languages

- Characterizing elements of a problem in form of “đối tượng” (object).
- Object-oriented is a technique to model a system by objects.



# Evolution of programming languages

- ***Is the history and evolution of abstraction***

- 1 • Assembly : Abstraction of data type/basic command
- 2 • Structure languages: control abstraction + functional abstraction
- 3 • OO languages: Data abstraction

1. Trừu tượng hóa các lệnh của máy tính. Các ngôn ngữ hợp ngữ trừu tượng hóa các mã nhị phân trên máy tính bằng các

2.

3.

# Reading exercises

- Read and summarize some differences between structure programming and OOP

<http://www.desy.de/gna/html/cc/Tutorial/node3.htm>

# What about other programming paradigms?

- Aspect-Oriented Programming
- Functional Programming

# Outline

1. Object-Oriented Technology
- 2. Object and Class
3. Java programming languages
4. Examples and Exercises

# Alan Kay' concepts

1. All are objects.
2. A software program can be considered as a set of objects interacting with each other
3. An object in a program has its own data and its own memory.
4. An object has all characteristics of its class.
5. All objects of a class have the same behavior



*Alan Kay*



## 2.1 Object

- **Object** is the key to understand the object technology
- In a OO system, all are objects



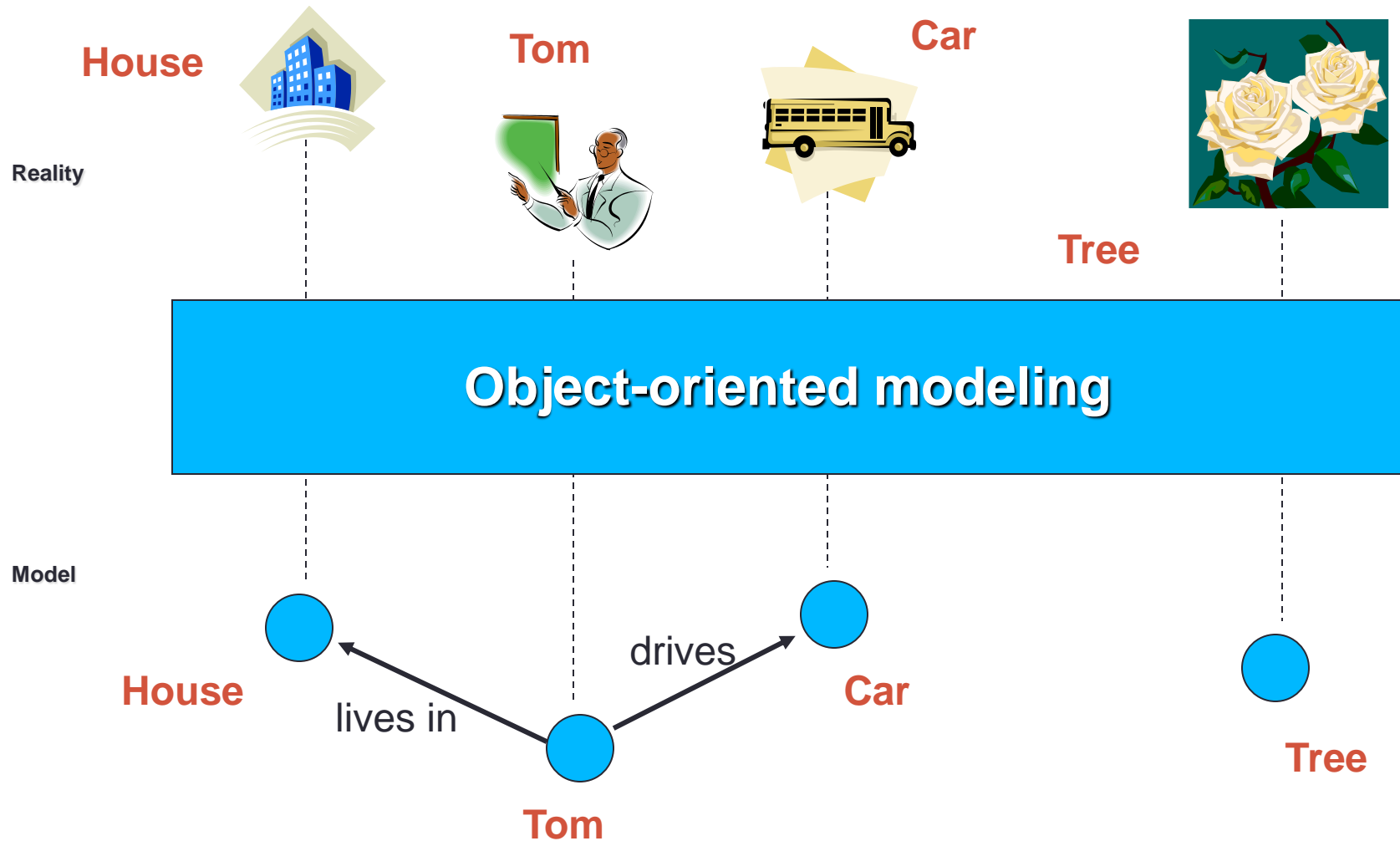
Writing a OO program means to build  
a model of some parts in the real world

## 2.1 What is object?

- Objects in the real world
  - For example, a car
- Related to a car:
  - Car information such as: color, speed,...
  - Car activities: moving forward, reversing, stopping,...

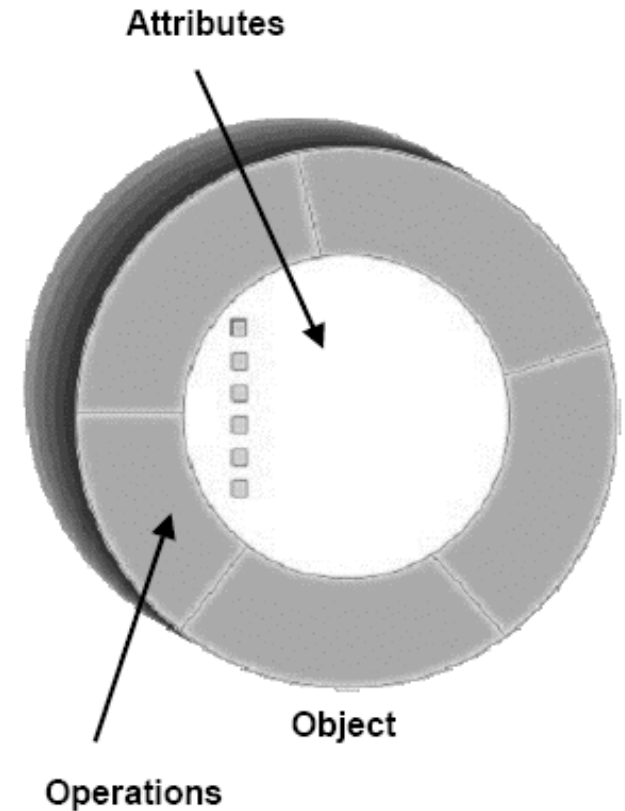


# What is object?



# What is object?

- Is an entity encapsulated in form of state and behavior.
  - **State** is represented by attributes and relationships.
  - **Behaviour** is represented by operations and methods.

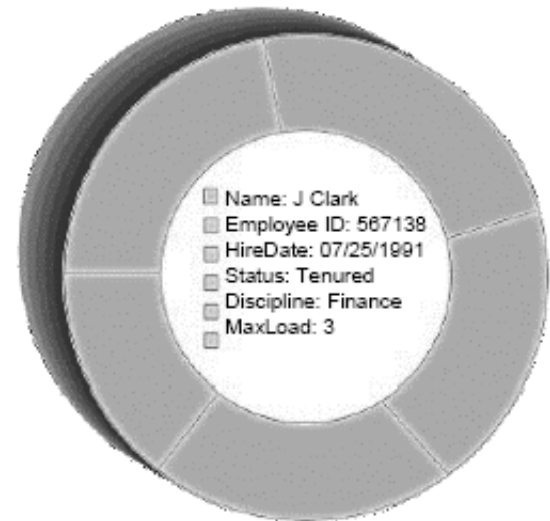


# An object has a state

- The state of an object is one of the possible conditions that the object exists.
- The state of an object can change over time



Name: J Clark  
Employee ID: 567138  
Date Hired: July 25, 1991  
Status: Tenured  
Discipline: Finance  
Maximum Course Load: 3 classes



Professor Clark

# State



Dave  
Age: 32  
Height: 6' 2"



Brett  
Age: 35  
Height: 5' 10"



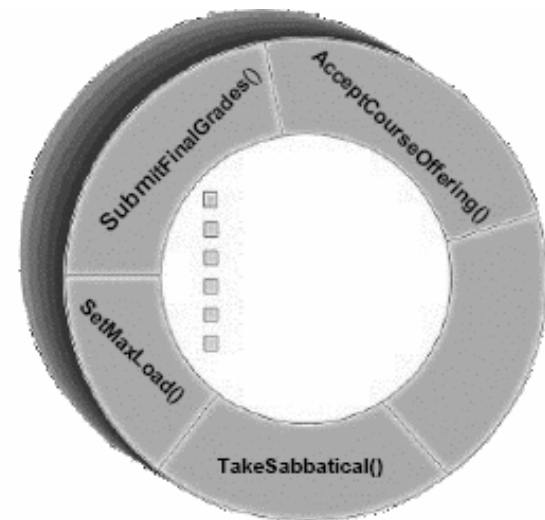
Gary  
Age: 61  
Height: 5' 8"

# An object has its behavior

- Behavior determines how an object acts and reacts to requests from other objects.
- Object behavior is represented by the operations that the object can perform.

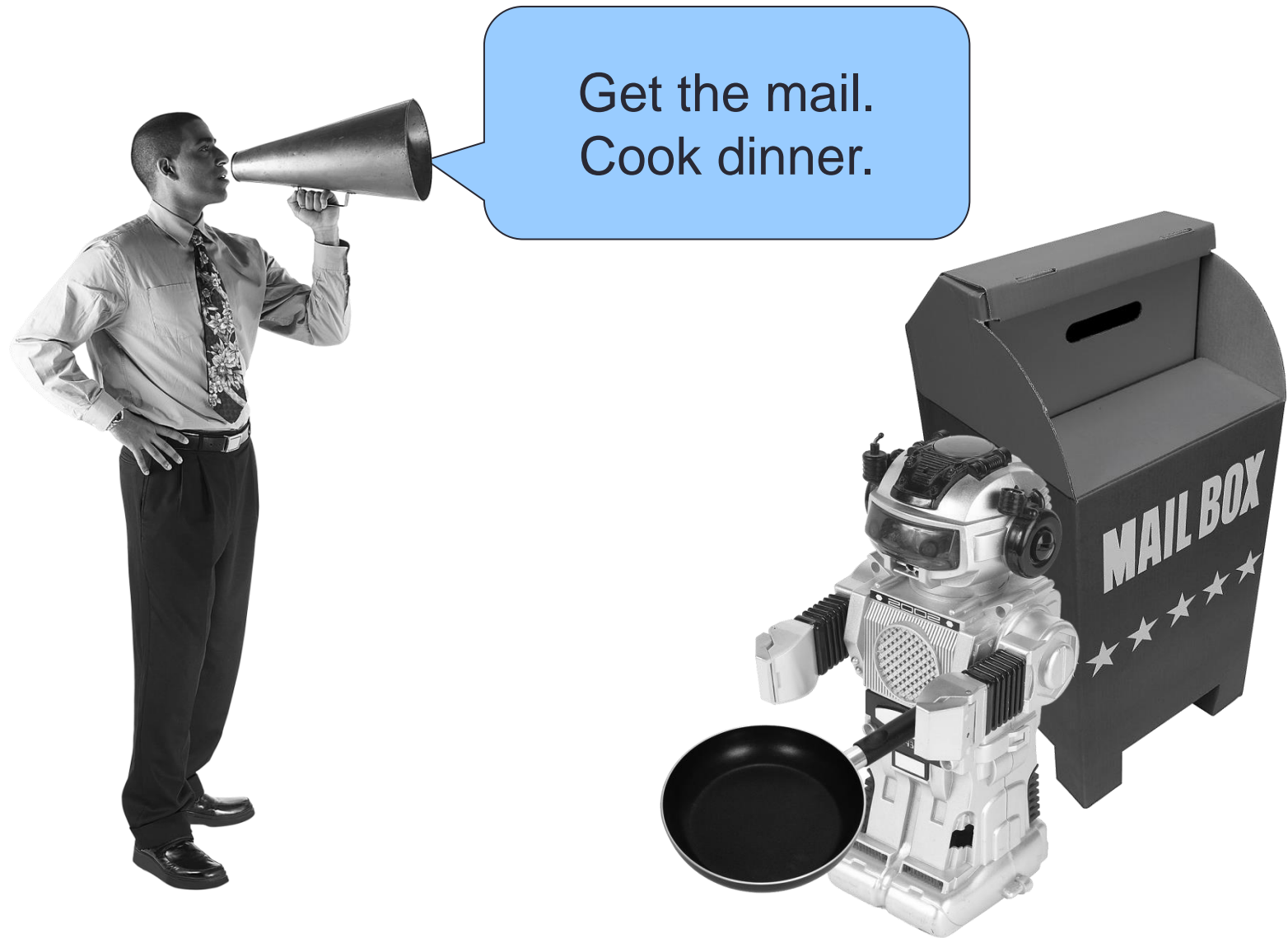


Professor Clark's behavior  
Submit Final Grades  
Accept Course Offering  
Take Sabbatical  
Maximum Course Load: 3 classes



Professor Clark

# Behavior





# An object has an unique identity

- Each object has its own unique identity, although two objects may share the same state (attributes and relationships)




**Professor “J Clark”  
teaches Biology**




**Professor “J Clark”  
teaches Biology**


ID




Okay, which one of you wise guys is the *real* Poppini?




I am the great Poppini!




I'm the great Poppini!



I am the great Poppini.

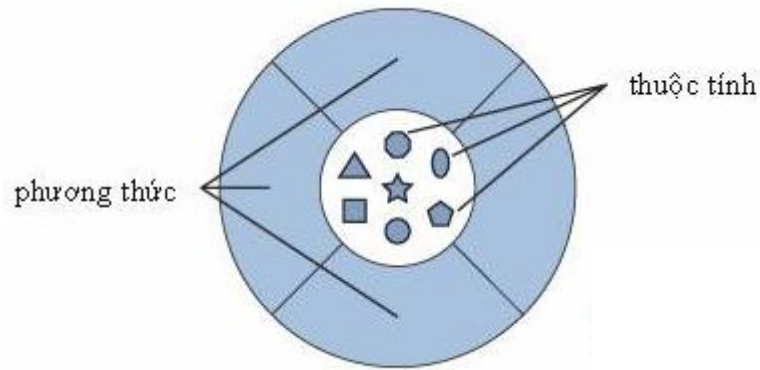


No, I'm the great Poppini.

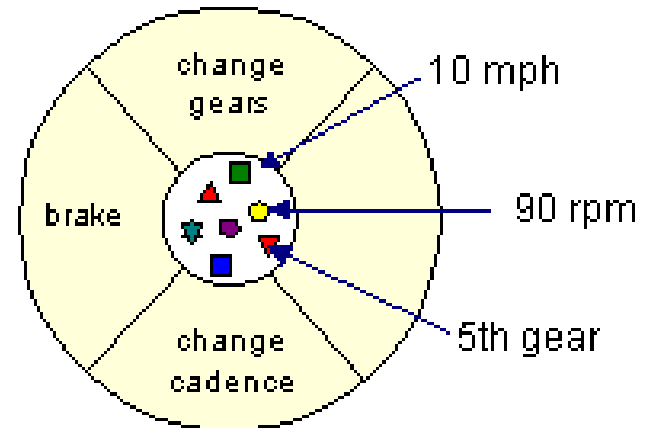


De great Poppini at-a your service.

# Object



Software object



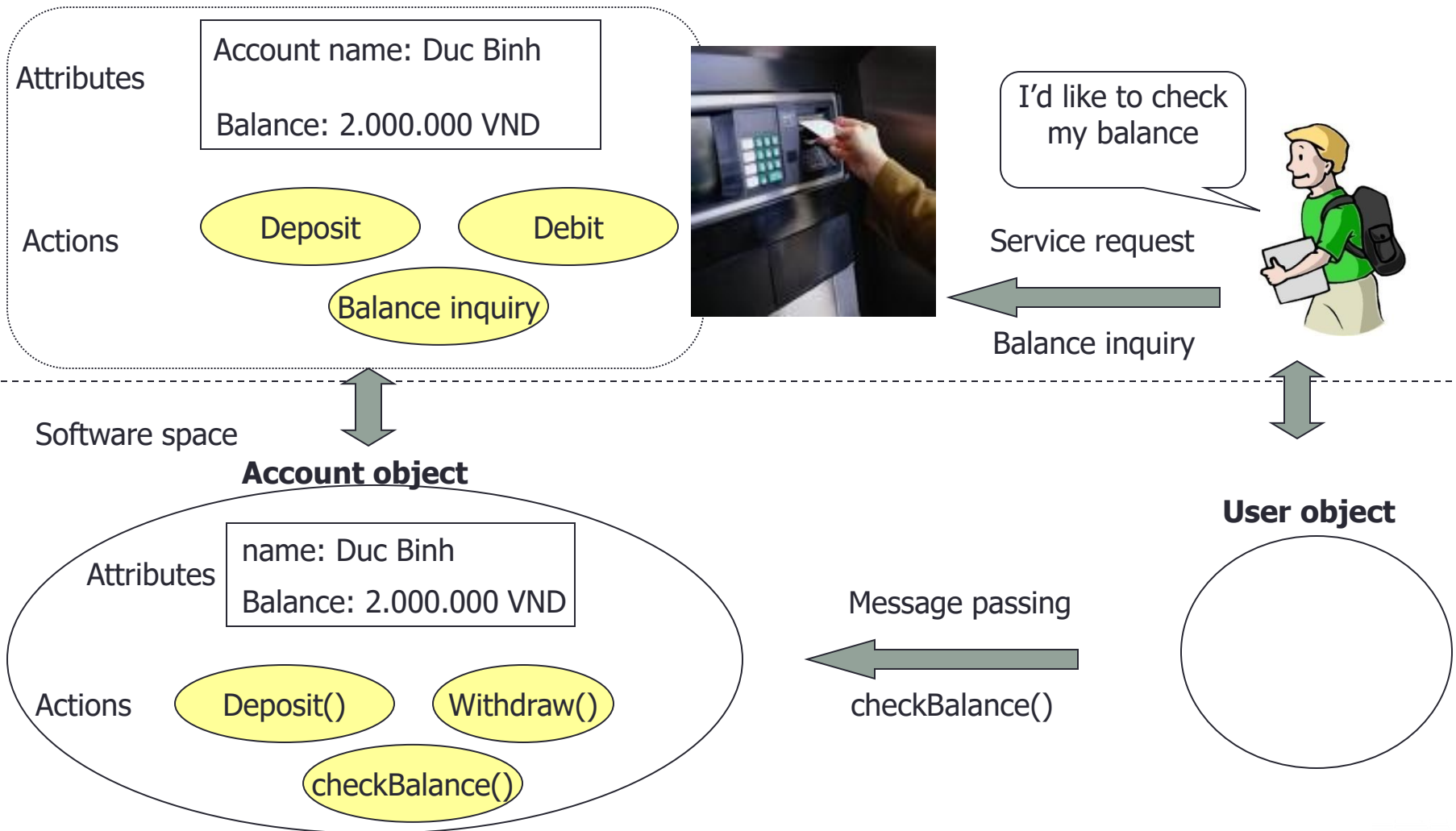
Software object **Xe Đạp**

Object is an entity encapsulating **attributes** và các **methods** involving.

Attributes are defined by a specific value called **representation attributes**. A specific object is called a **representation**.

# Software objects and a real-life problem

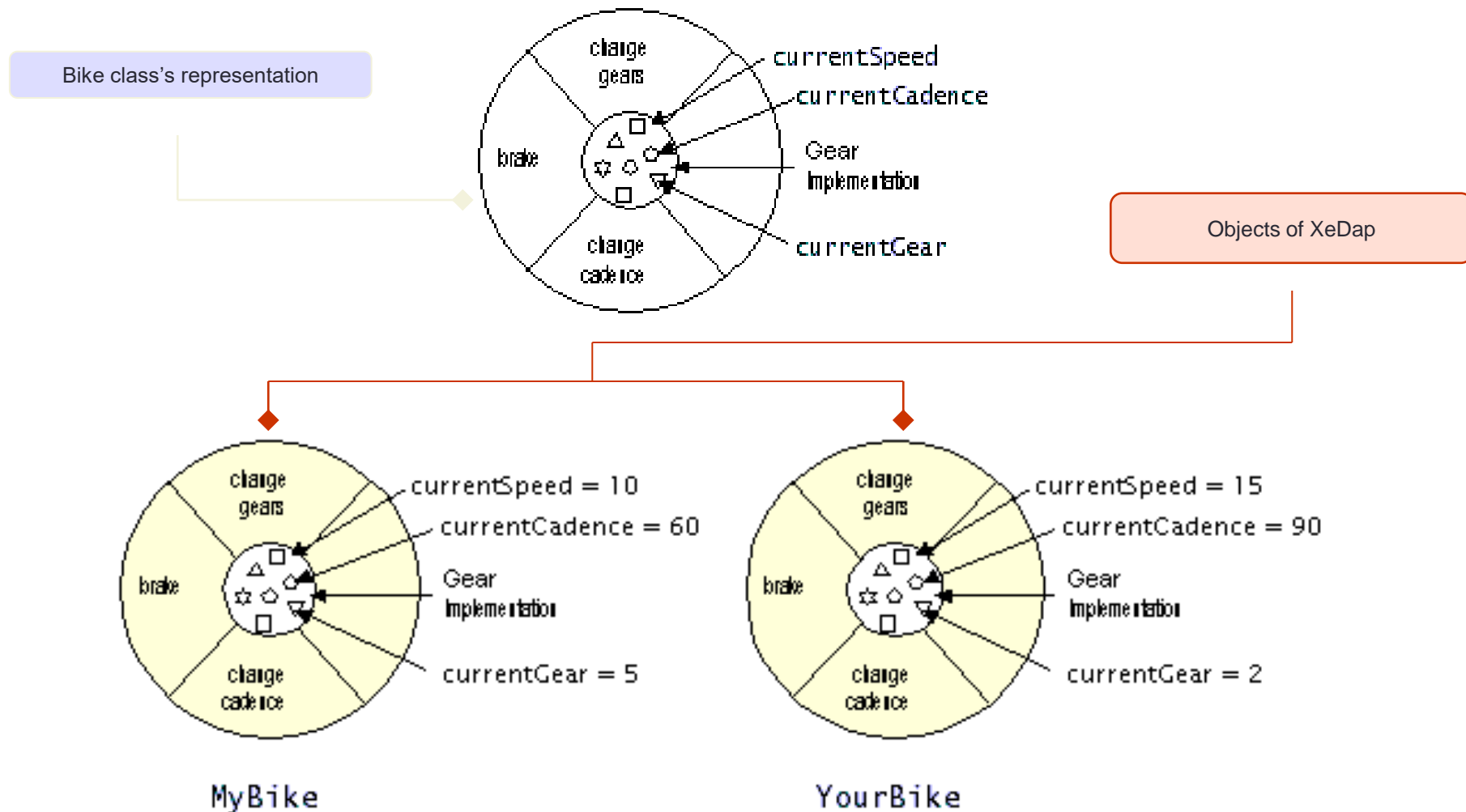
Problem of bank account management – ATM ther – electronic payment



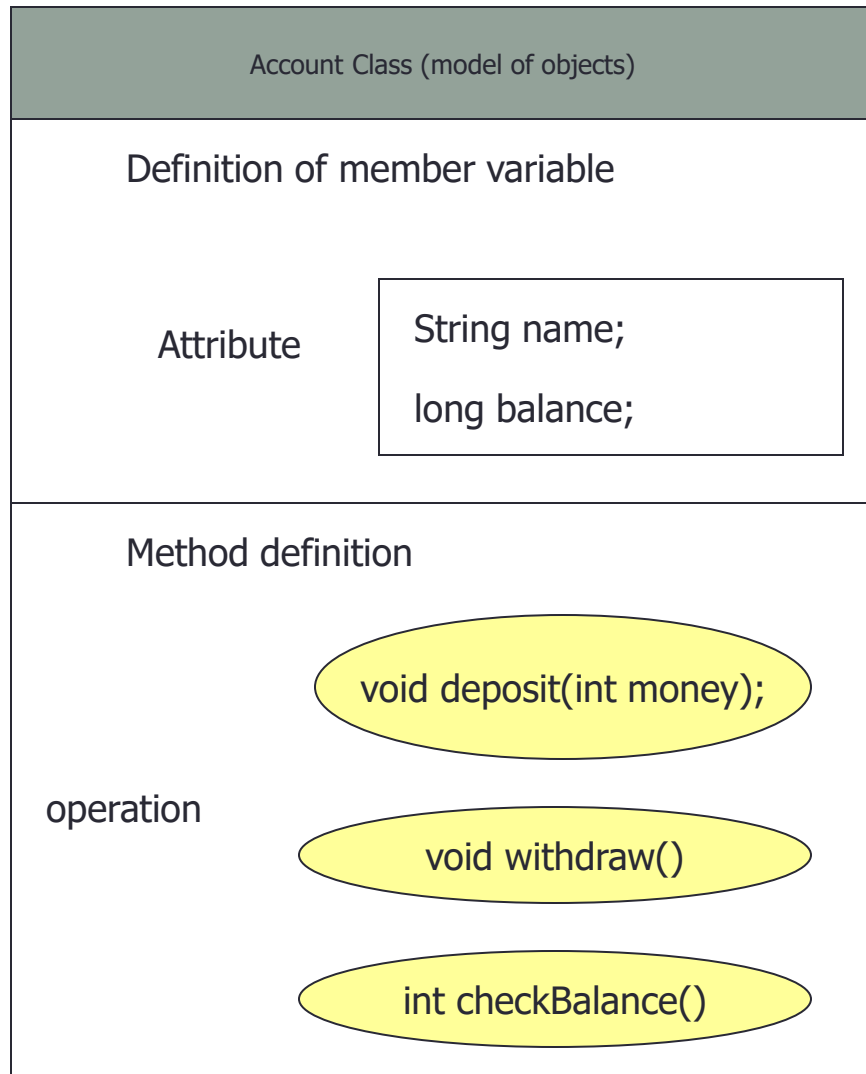
## 2.2 Class

- A class is a blueprint or prototype for all the objects of a same type
  - Example: class Bike is a common blueprint for many bike objects that are created
- A class defines common attributes and methods for all the objects of some type
- An object is a detailed representation of a class.
  - Example: a bike object is a representation of the class Bicycle
- Each object can have different attribute's representation
  - Example: a bike can be at the 5<sup>th</sup> gear while another bike can be at the 3<sup>rd</sup> gear.

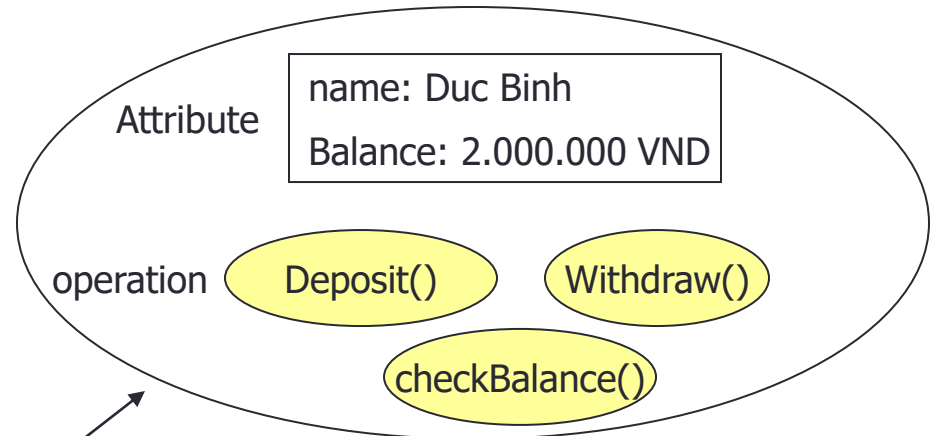
# Example: Bike class



# Class and Object

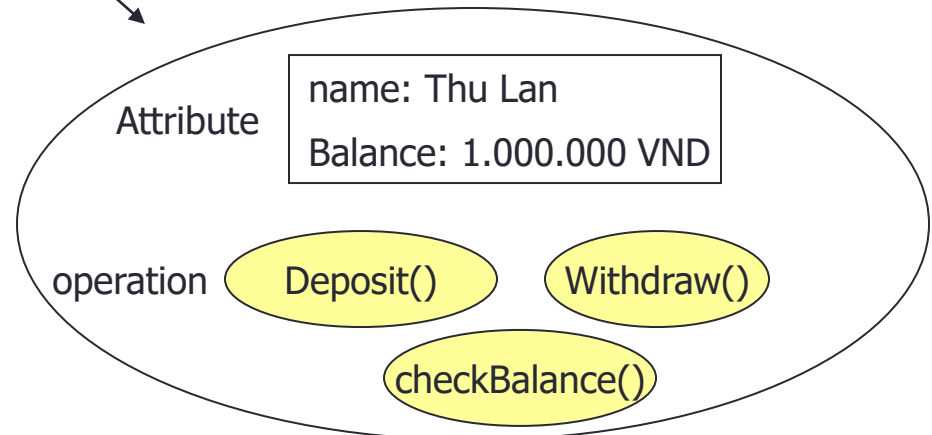


**Account object of Mr Duc Binh**



INSTANTIATE

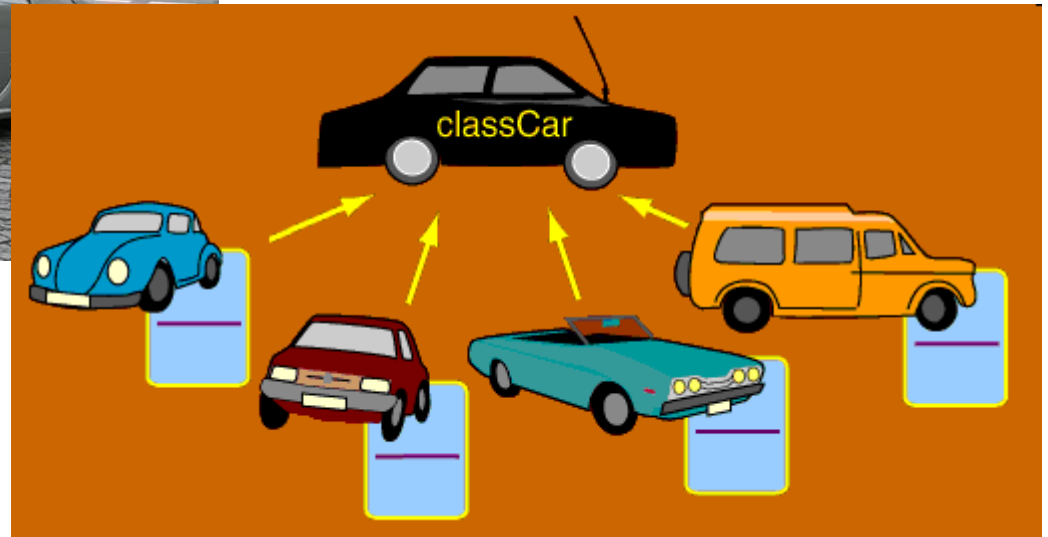
**Account object of Mrs Thu Lan**



# Class and Object



Blueprint/prototype



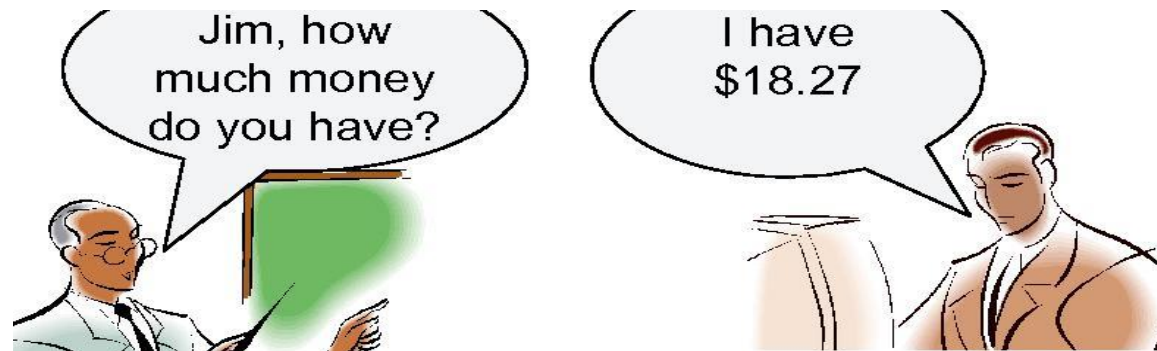


# Quick question

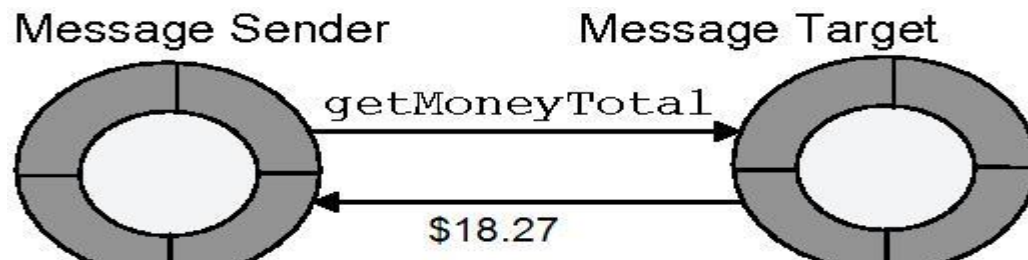
- Given the Amazon online shopping system. Provide some examples about class and object in this system?
- The same question for HUST Student Information System?

## 2.3 Interactions between objects

- Communication between objects in the real world



- Objects and their interactions in programming
  - Objects communicate to each other by message passing



# Message passing

- A program (built via OOP) is a set of objects exchanging messages between them

## Employee Object



### Behaviors

get\_SS#()  
get\_Gender()  
get\_Date\_of\_Birth()

Message - get\_SS#()

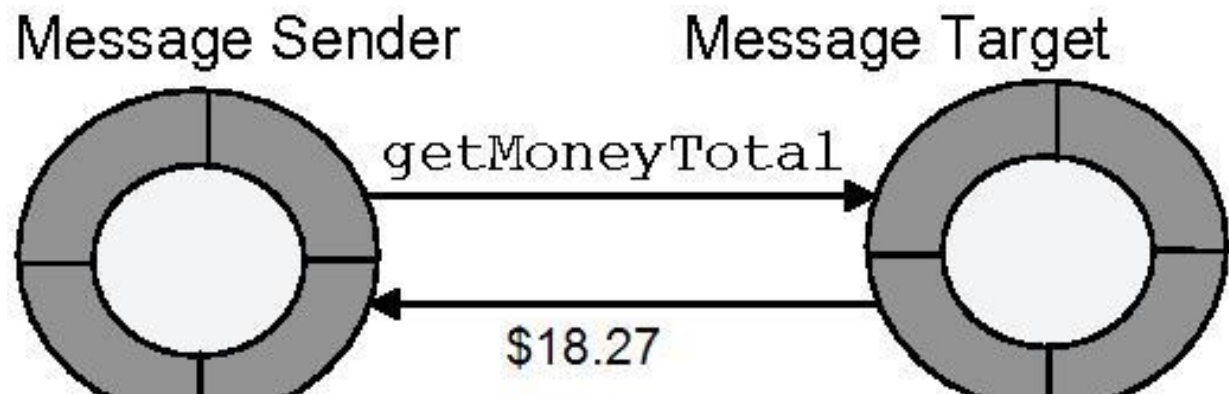
## Payroll Object



return SS#

## 2.4 Structure-Oriented vs. OO?

- Structure-Oriented:
  - data structures + algorithms = Program
- Object-Oriented:
  - objects + messages = Program



# Procedural-oriented vs. Object-oriented

- Procedural Programming:
  - Main components are procedures, functions
  - Data is independent with procedures
- Object-oriented programming
  - Main components are objects
  - Data is associated to function (method) in an object
  - Each data structure has methods executing on it

# Examples of class and object in some OOP languages

*Class declaration:* **each class** is, by default, an **extension of Object** (can be omitted)

*Class constructor:* **initialises** the various **fields**

*Class method:* **retrieves** and/or **modifies** the **state** of the class

```
public class Time extends Object {
    private int hour;
    private int minute;
    private int second;

    public Time () {
        setTime(0, 0, 0);
    }

    public void setTime (int h, int m, int s) {
        hour = ( ( h >= 0 && h < 24 ) ? h : 0 );
        minute = ( ( m >= 0 && m < 60 ) ? m : 0 );
        second = ( ( s >= 0 && s < 60 ) ? s : 0 );
    }
}
```

*Class fields:* **private** means they **can not be accessed** from **outside** the class

# Java: Program and object

```
public class Test {  
  
    public static void main (String args[]) {  
        Time time = new Time();  
  
        time.hour = 7;  
        time.minute = 15;  
        time.second = 30;  
    }  
}
```

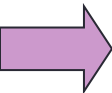
```
Test.java:6: hour has private access in Time  
            time.hour = 7;  
                ^
```

```
Time.java:7: minute has private access in Time  
            time.minute = 15;  
                ^
```

```
Time.java:8: second has private access in Time  
            time.second = 30;  
                ^
```

```
3 errors
```

# Outline

1. Object-Oriented Technology
2. Object and Class
-  3. Java programming languages
4. Examples and Exercises



## 3.1 What is Java?

- Java is a object-oriented programming language developped by Sun Microsystems, and now bought by Oracle
- Java is a popular programming language
  - Initially used for building control processor applications inside the electronics consumer devices such as cell phones, microwaves ...
  - Initially used in 1995



Green Team and James Gosling  
(the leader)



# J2SE (Java 2 Platform Standard Edition)

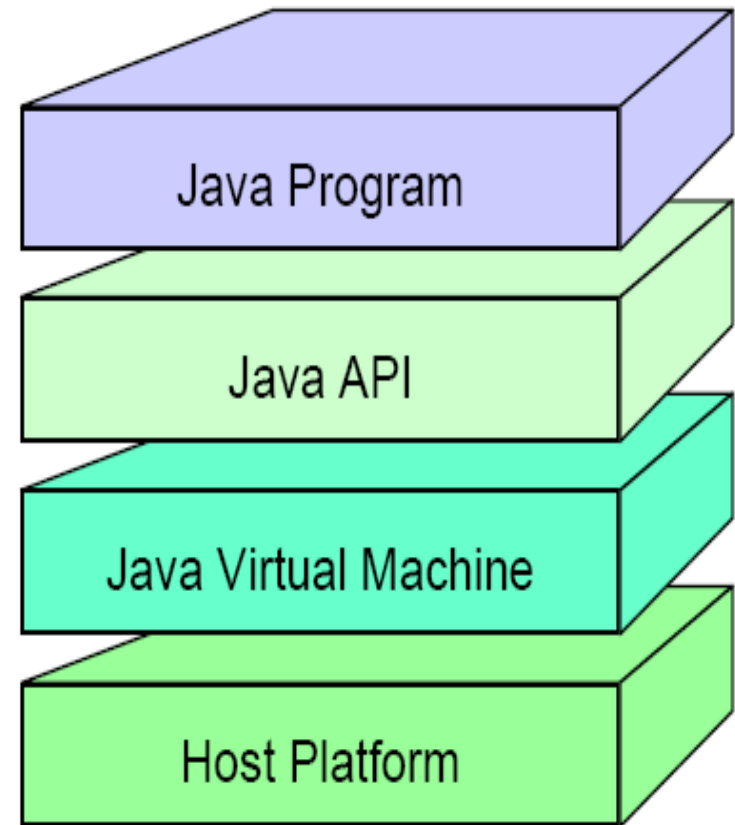
- <http://java.sun.com/j2se>
- Java 2 Runtime Environment, Standard Edition (J2RE):
  - Executable Environment or JRE provides Java APIs, Java Virtual Machine (JVM) and other necessary components to run applets and applications written in Java.
- Java 2 Software Development Kit, Standard Edition (J2SDK)
  - Super set of JRE, and contains everything in the JRE, additional tools such as compilers and the debugger need to develop applets and applications.

# J2EE (Java 2 Platform Enterprise Edition)

- <http://java.sun.com/j2ee>
- Service-Oriented Architecture (SOA) và Web services
- Web Applications
  - Servlet/JSP
  - JSF...
- Enterprise Applications
  - EJB
  - JavaMail...
- ...

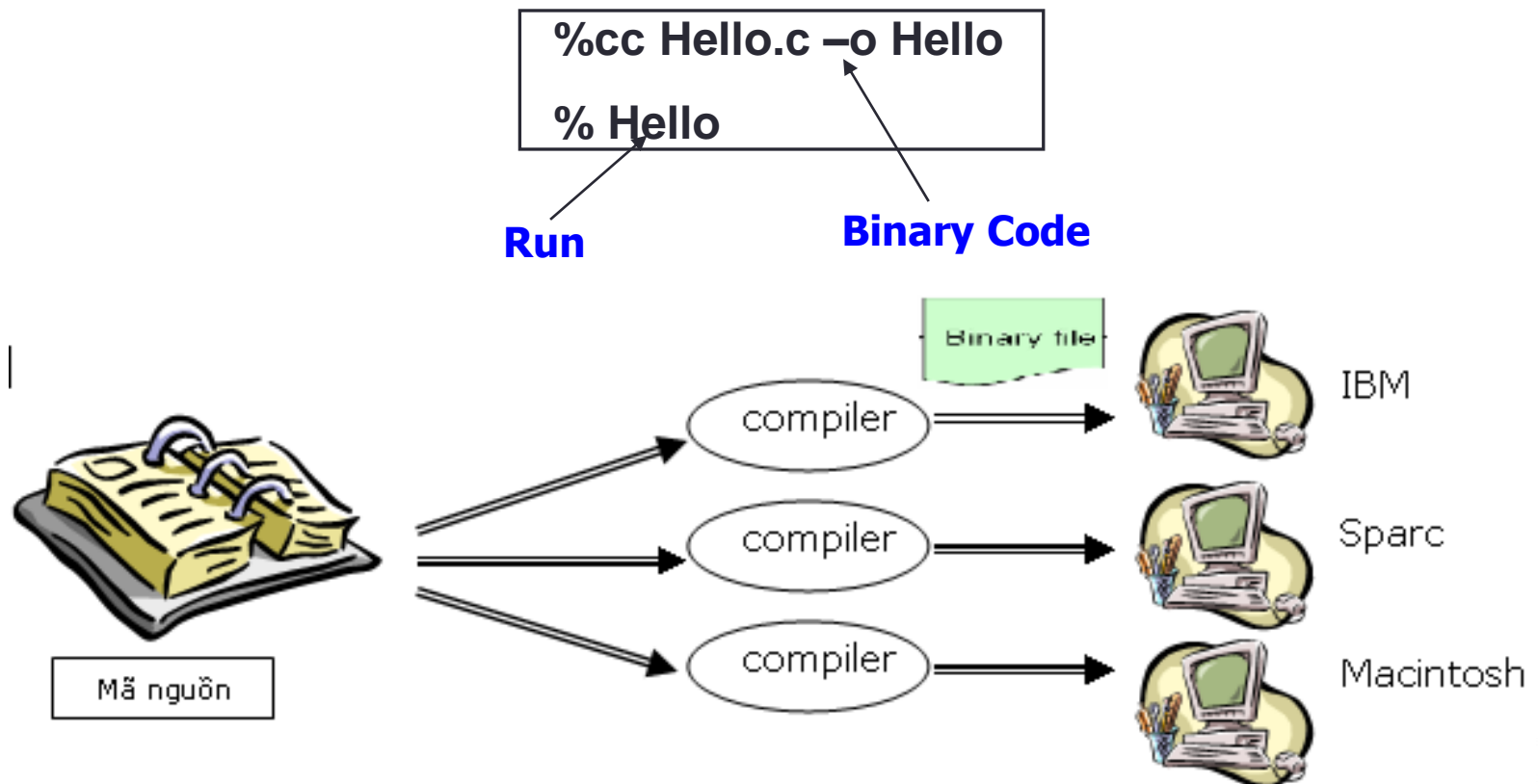
## 3.2 Java platform

- Platform is environment for development of deployment.
- Java platform can be run on all OSs
  - Other platforms depend on hardware
  - Java platform provides:
    - Java Virtual Machine (JVM).
    - Application Programming Interface (API).



## 3.3. Compiling model of Java

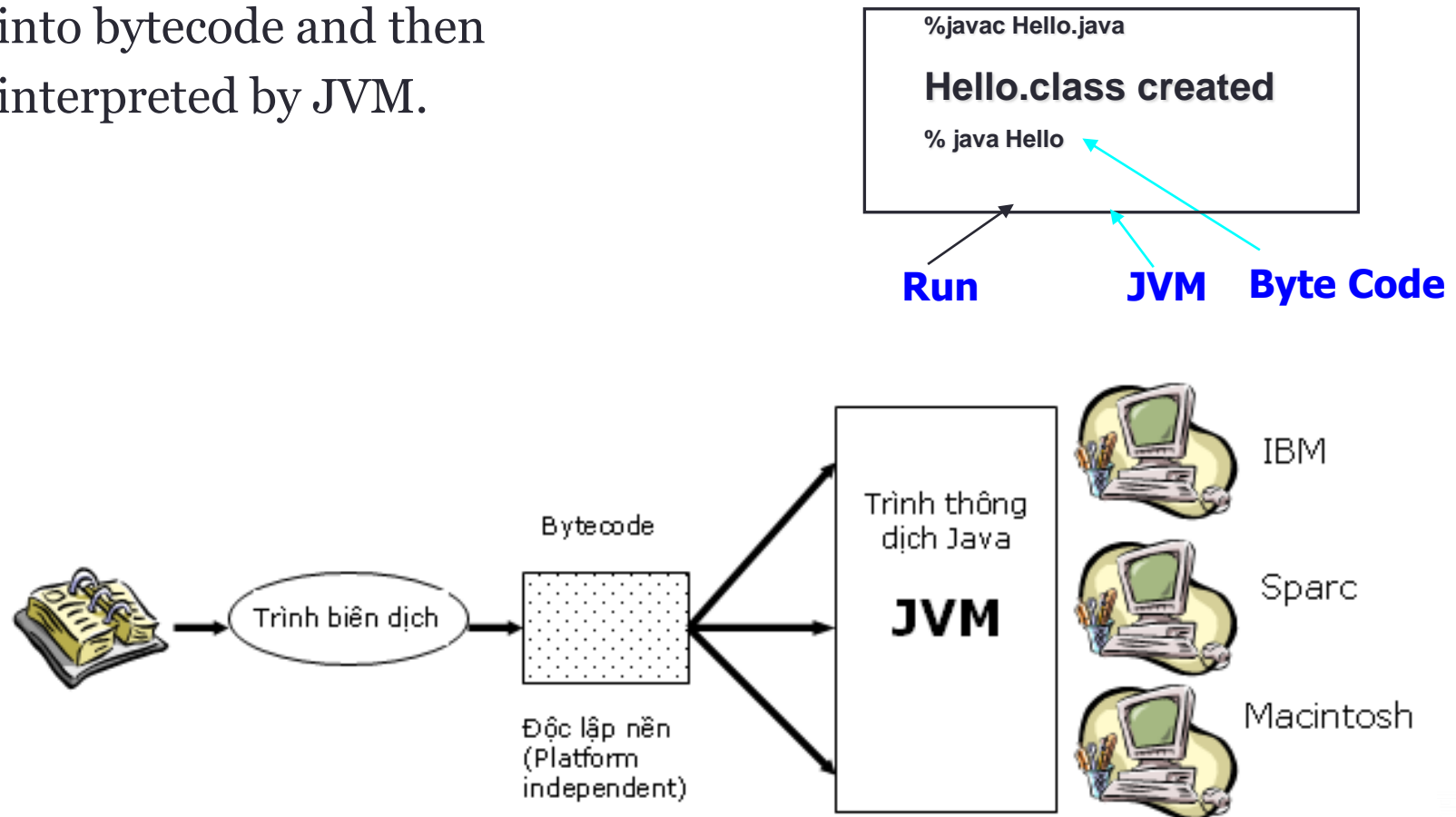
- a. Classical compiling model:
  - Source code is compiled into binary code.



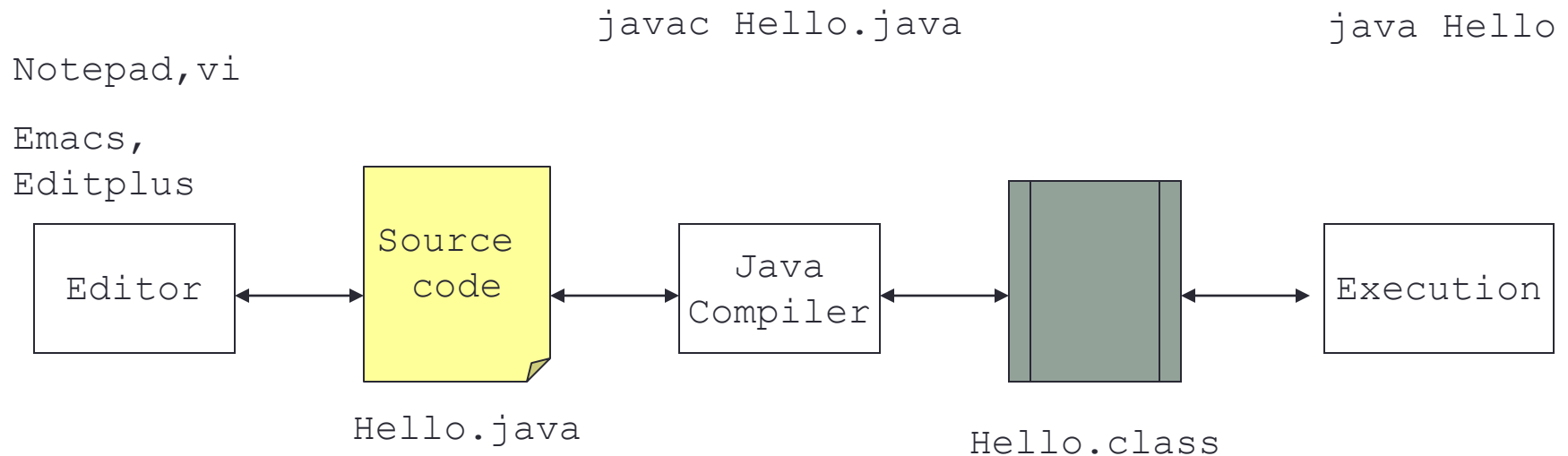
## 3.3. Compiling model of Java (2)

- b. Compiling model of Java:

- Source code is compiled into bytecode and then interpreted by JVM.



# Making procedure of Java Application



## 3.3. Compiling Model of Java (3)

- Java Virtual Machine:
  - JVM is the heart of Java language
    - Bring the feature “Write once, run everywhere”
  - Provides environment to execute instructions:
    - Load file .class
    - Manage memory
    - Garbage collections
  - The Interpreter “**Just In Time - JIT**”
    - Transform bytecode to machine code for each type of CPU.



## 3.4. Features of Java

- Java is designed to be:
  - A powerful programming language, full of OO features and completely OO.
  - Easy to learn, syntax is similar to C++
  - Platform independance
  - Support the development of applications in network environment
  - Ideal for Web application

## 3.4. Features of Java (2)

- Powerful
  - Class library: Hundreds of classes already written with utility functions.
  - Java uses pointer model without accessing directly to the memory; memory can not be over-written.
- Object-Oriented
  - Java supports software development by using OO
  - Software built in Java includes classes and objects

## 3.4. Features of Java (3)

- Simple
  - Keywords
    - Java has 50 keywords
      - Compared to Cobol VB that have hundreds of keywords
- Network capable
  - Java supports the development of distributed applications
  - Some applications of Java are designed in order to be accessed via Web browser.

## 3.4. Features of Java (3)

- Java has 50 key words
  - assert (New in 1.5) enum (New in 1.5)

<code>abstract</code>	<code>boolean</code>	<code>break</code>	<code>byte</code>
<code>case</code>	<code>catch</code>	<code>char</code>	<code>class</code>
<code>const</code>	<code>continue</code>	<code>default</code>	<code>do</code>
<code>double</code>	<code>else</code>	<code>extends</code>	<code>final</code>
<code>finally</code>	<code>float</code>	<code>For</code>	<code>goto</code>
<code>If</code>	<code>implements</code>	<code>import</code>	<code>instanceof</code>
<code>int</code>	<code>interface</code>	<code>long</code>	<code>native</code>
<code>new</code>	<code>package</code>	<code>private</code>	<code>protected</code>
<code>public</code>	<code>return</code>	<code>short</code>	<code>static</code>
<code>strictfp</code>	<code>super</code>	<code>switch</code>	<code>synchronized</code>
<code>this</code>	<code>throw</code>	<code>throws</code>	<code>transient</code>
<code>try</code>	<code>void</code>	<code>volatile</code>	<code>while</code>

## 3.4. Features of Java (5)

- Multi-threaded
  - Allows a program to run more than one task at the same time.
- Portable
  - Programs can be written once and run on different platforms
  - Based on compiler/interpreter model  
(WORE – Write Once, Run Everywhere)

## 3.4. Features of Java (6)

- Development Environment
  - Java Development Kit
    - Free on Sun Website: [java.sun.com](http://java.sun.com)
    - Including: Compiler, JVM and existing classes
  - Integrated Development Environments (IDEs): Providing:
    - Complex Text Editors
    - Debugging Tools
    - Graphics Development Tools

## 3.5. Applications in Java

- Application
  - Do not need to run on browsers
  - Can call functions through commands or option menu (GUI)
  - `main()` method is the starting point of the program execution
- Applet
  - GUI application running on browser in the client side.
  - Can be viewed by appletviewer or embedded in Web browser with JVM installed.

## 3.5. Applications in Java (2)

- Web application
  - Create dynamic content on Server instead of on browsers.
  - Used in Server application
  - Servlet: manage requests from browsers and send the responses back
  - JavaServer Page (JSP): HTML pages with embedded Java code.

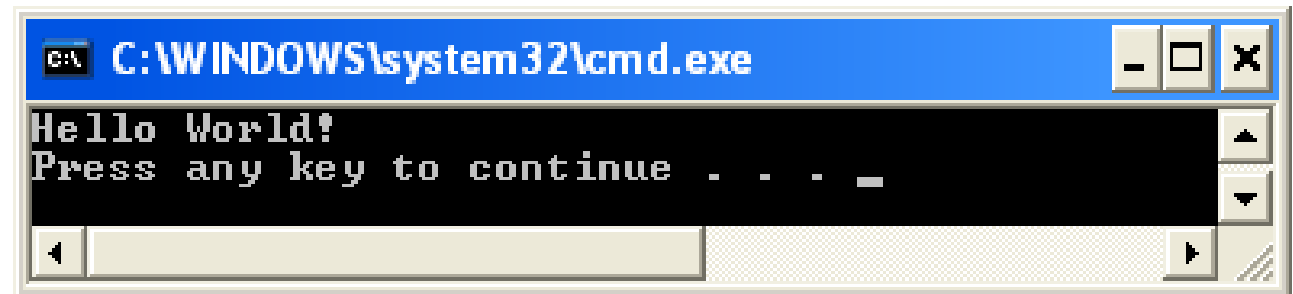


# Outline

1. Object-Oriented Technology
2. Object and Class
3. Java programming languages
- 4. Examples and Exercises

# Example 1 - HelloWorld

```
// HelloWorld.java
// Chương trình hiển thị dòng chữ "Hello World"
public class HelloWorld {
    /* Phương thức main sẽ được gọi đầu tiên
       trong bất kỳ ứng dụng Java nào */
    public static void main(String args[]){
        System.out.println( "Hello World!" );
    } // kết thúc phương thức main
} // kết thúc lớp HelloWorld
```



The screenshot shows a Windows Command Prompt window with the title bar "C:\WINDOWS\system32\cmd.exe". The window has a blue title bar, a black command area, and a white scroll bar on the right. The text "Hello World!" is displayed on the first line, and "Press any key to continue . . . \_" is on the second line. The cursor is at the end of the second line. The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

# Example 1 (Cont.)

- Comment
  - In one line: Starts with //
  - In multiple lines: /\* . . . \*/
- Java distinguish between lowercase and uppercase
- Keywords in Java:
  - class: Class definition
  - public: Access permission
- Class name containing main function must have the same name with the file .java.

# Installing and Running Java application

- Step 1: Install jdk, install environment variables (if using cmd)
- Step 2: Install Eclipse or Netbean IDE
- Step 3: Coding
- Step 4: Compile
  - cmd: `javac HelloWorld.java`
  - Eclipse/Netbean: Build automatically (Look at Console to see syntax errors if any)/F11 (Project) or F9 (File)
- Step 5: Run program
  - cmd: `java HelloWorld`
  - Eclipse/Netbean: Run as Java application (Alt+Shift+X+J)/F6 (Project) or Shift-F6 (File)

# Environment Variables

- `PATH = %PATH%;C:\Program Files\Java\jdkx.x\bin`
- `JAVA_HOME=C:\Program Files\Java\jdkx.x`
- `CLASSPATH = C:\Program  
Files\Java\jdkx.x\lib;.;C:\Program  
Files\Java\jdkx.x\include`

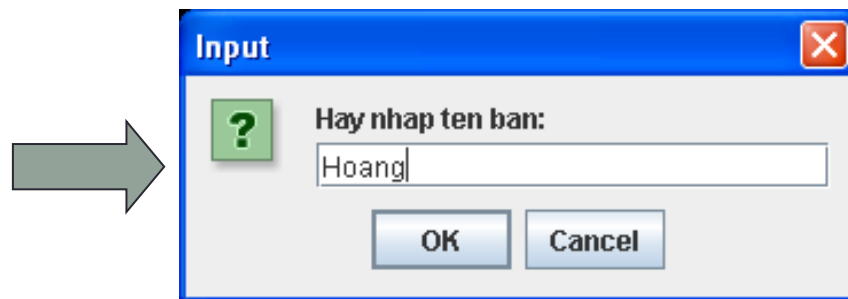
## Example 2 - GUI

```
import javax.swing.JOptionPane;  
public class FirstDialog{  
    public static void main(String[] args){  
        JOptionPane.showMessageDialog(null,  
                                     "Xin chao ban!");  
        System.exit(0);  
    }  
}
```



## Example 3 - Data Input/Output

```
import javax.swing.JOptionPane;  
public class HelloNameDialog{  
    public static void main(String[] args){  
        String result;  
        result = JOptionPane.showInputDialog("Hay nhap  
                                                ten ban:");  
        JOptionPane.showMessageDialog(null,  
                                     "Xin chao " + result + "!");  
        System.exit(0);  
    }  
}
```



# Example of Class and Object in Java

*Class declaration:* **each class** is, by default, an **extension of Object** (can be omitted)

*Class constructor:* **initialises** the various **fields**

*Class method:* **retrieves** and/or **modifies** the **state** of the class

```
public class Time extends Object {
    private int hour;
    private int minute;
    private int second;

    public Time () {
        setTime(0, 0, 0);
    }

    public void setTime (int h, int m, int s) {
        hour = ( ( h >= 0 && h < 24 ) ? h : 0 );
        minute = ( ( m >= 0 && m < 60 ) ? m : 0 );
        second = ( ( s >= 0 && s < 60 ) ? s : 0 );
    }
}
```

*Class fields:* **private** means they **can not be accessed** from **outside** the class



# Java: Program and Objects

```
public class Test {  
  
    public static void main (String args[]) {  
        Time time = new Time();  
  
        time.hour = 7;  
        time.minute = 15;  
        time.second = 30;  
    }  
}
```

```
Test.java:6: hour has private access in Time  
            time.hour = 7;  
              ^  
Time.java:7: minute has private access in Time  
            time.minute = 15;  
              ^  
Time.java:8: second has private access in Time  
            time.second = 30;  
              ^  
3 errors
```