

Programming in Java: lecture 1

- Overview of the course
- Java Virtual Machine (JVM)
- Building blocks of programs
- Object Oriented Programming
- Eclipse
- Hello World

Slides made for use with "Introduction to Programming Using Java" by David J. Eck
Some figures are taken from "Introduction to Programming Using Java" by David J. Eck

Overview of the course

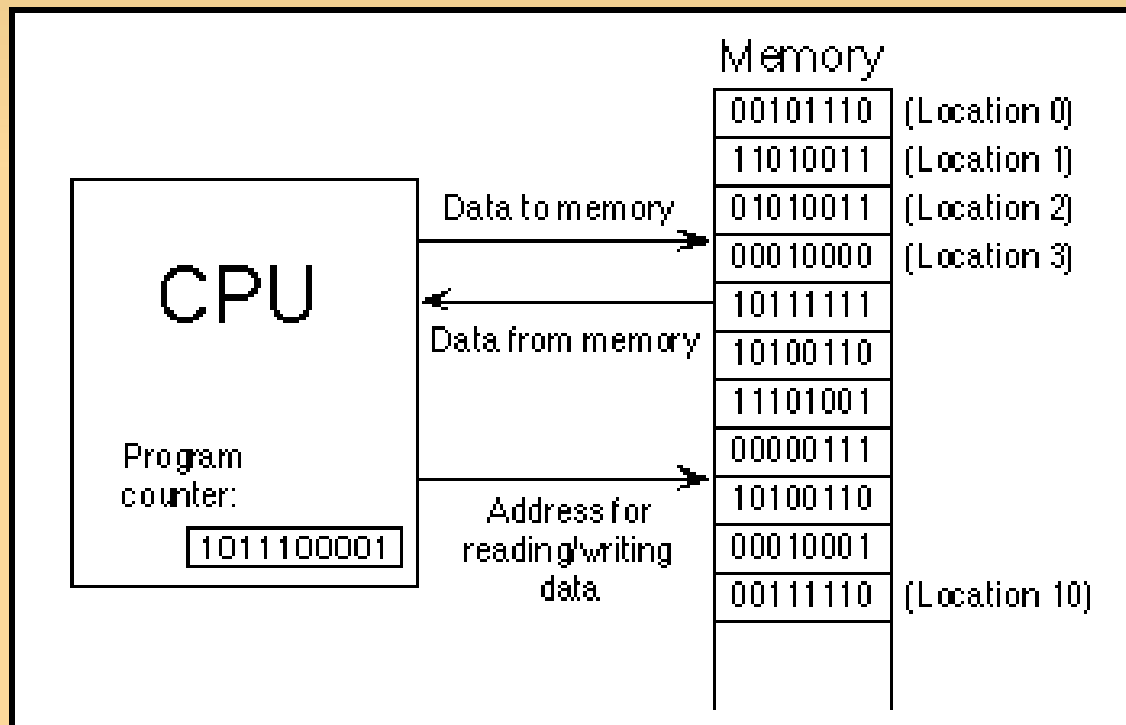
- Purpose: Learn to program
 - Basic Programming
 - Control structures, data types
 - Searching and sorting
 - Recursion
 - Knowledge of Object Oriented Programming
 - Inheritance and Polymorphism
 - Later you will have: OOP and OOA&D
- Exam: Written test: 48 hours home assignment

Java Virtual Machine

- Why a virtual machine
- What do we mean by “virtual”
- Explain a regular machine
- Java and Java Byte Code

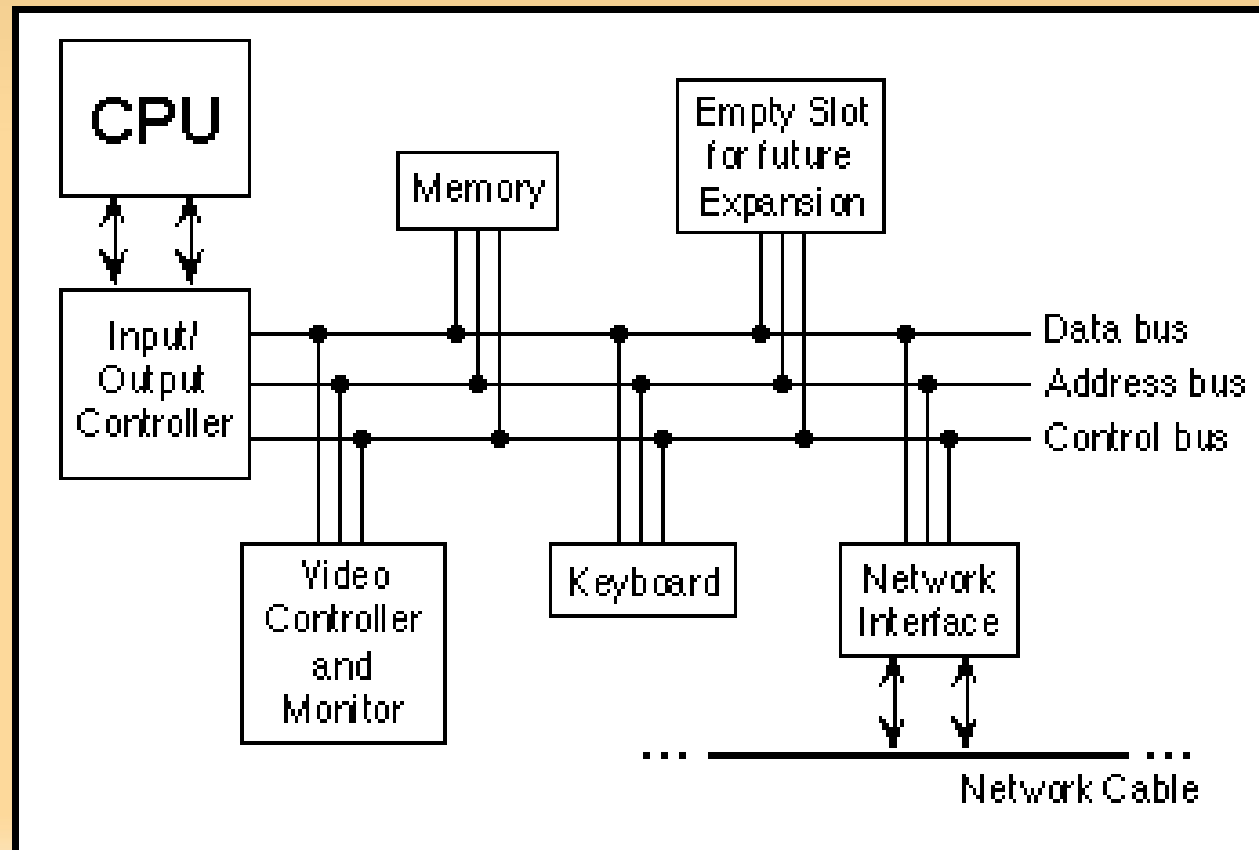
CPU

- Fetch execute cycle
- Machine language



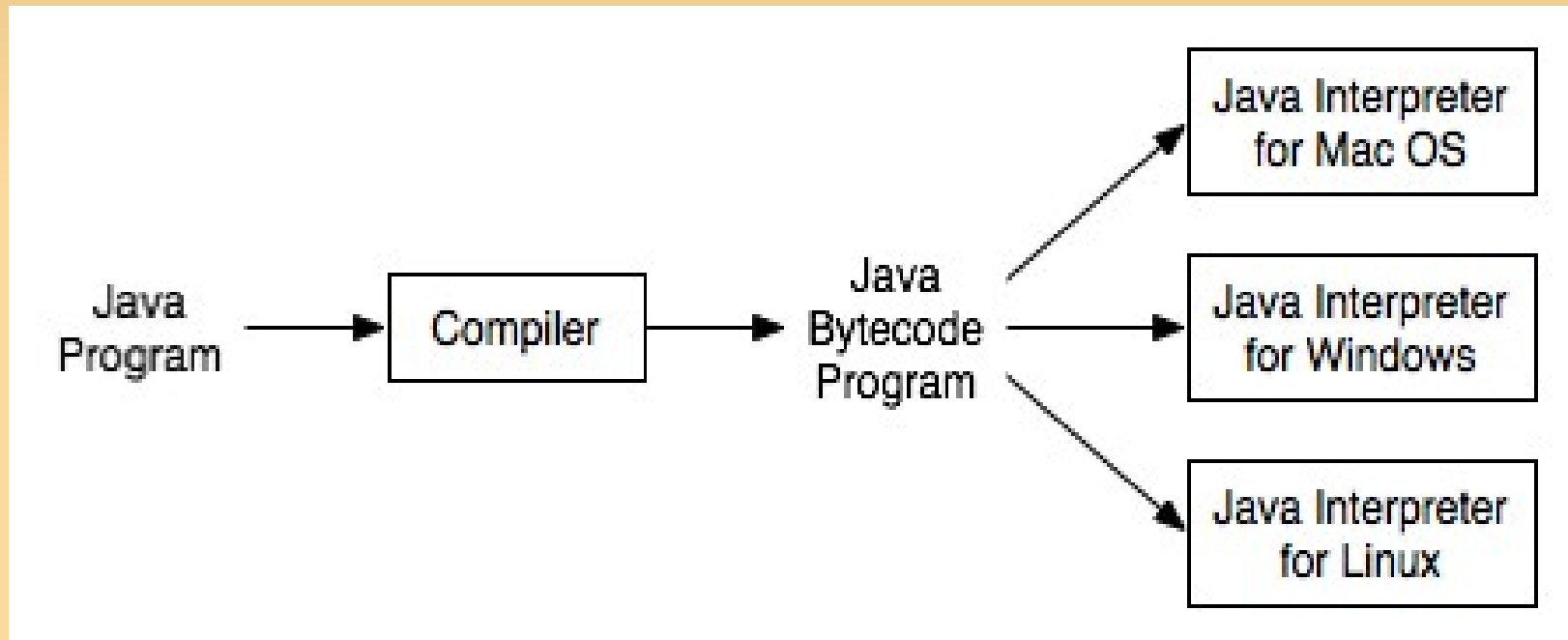
Machine Architecture

- Basic Computer Architecture
- Asynchronous events



Java Virtual Machine

- Why a virtual machine?



Compilation

Java Byte Code:

```
0:  iconst_2
1:  istore_1
2:  iload_1
3:  sipush 1000
6:  if_icmpge 44 ←
9:  iconst_2
10: istore_2
11: iload_2
12: iload_1
13: if_icmpge 31
16: iload_1
17: iload_2
18: irem      # remainder
19: ifne     25
22: goto     38
25: iinc     2, 1
28: goto     11
31: getstatic #84; //Field java/lang/System.out:Ljava/io/PrintStream;
34: iload_1
35: invokevirtual #85; //Method java/io/PrintStream.println:(I)V
38: iinc     1, 1
41: goto     2
44: return
```

Java Code:

```
for (int i = 2; i < 1000; i++) {
    for (int j = 2; j < i; j++) {
        if (i % j == 0)
            continue outer;
    }
    System.out.println (i);
}
```

Building blocks of programs

- Data
 - Variables
 - Types
- Instructions
 - Control structures
 - organize code
 - Subroutines
 - reuse

```
Java Code:  
for (int i = 2; i < 1000; i++) {  
    for (int j = 2; j < i; j++) {  
        if (i % j == 0)  
            continue outer;  
    }  
    System.out.println (i);  
}
```


History of Programming

- Structured programming
 - Divide problem into smaller problems
 - top-down approach
 - Focus on instructions, not data
- Object Oriented Programming
 - Model the problem area
 - bottom-up approach
 - Focus on data, not instructions

Object Oriented Programming

- What is an object?
 - Represents real world objects
 - Data and associated methods (functions).
 - Data hiding
 - Polymorphism
 - Classes
 - Inheritance

Data Hiding

- Ensuring
 - modularity
 - data integrity
- Enabling
 - reuse
 - local modifications

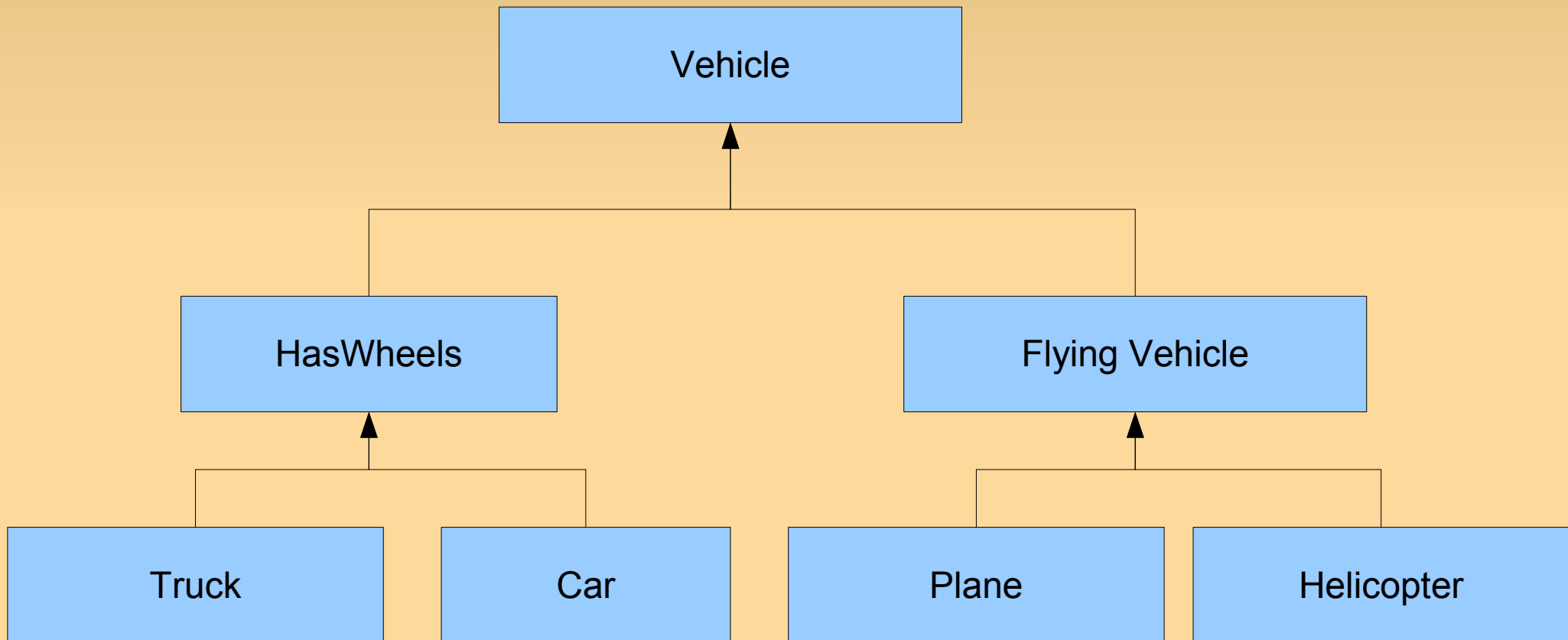
Polymorphism

- The same message send to different objects will have different effects
- Code that operates on data types that we have not defined yet

Classes

- Template
- Description of a group of objects
- Example: Vehicle

Inheritance



Command Line Interface

- Windows: Run Program (cmd)
- Linux: xterm, gterm, ...
- Mac OS: Terminal
- Javac – compiler
 - > javac HelloWorld.java
- Java – execution
 - > java HelloWorld
 - Hello World!

Packages

- Packages
 - > package mypackage;
- Compilation with packages
 - Windows
 - > javac mypackage\HelloWorld.java
 - Linux
 - > javac mypackage/HelloWorld.java

Eclipse Demo

Hello World Example

```
// A program to display the message
// "Hello World!" on standard output
public class HelloWorld {

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

} // end of class HelloWorld
```