Section 11: Comparing Java and C

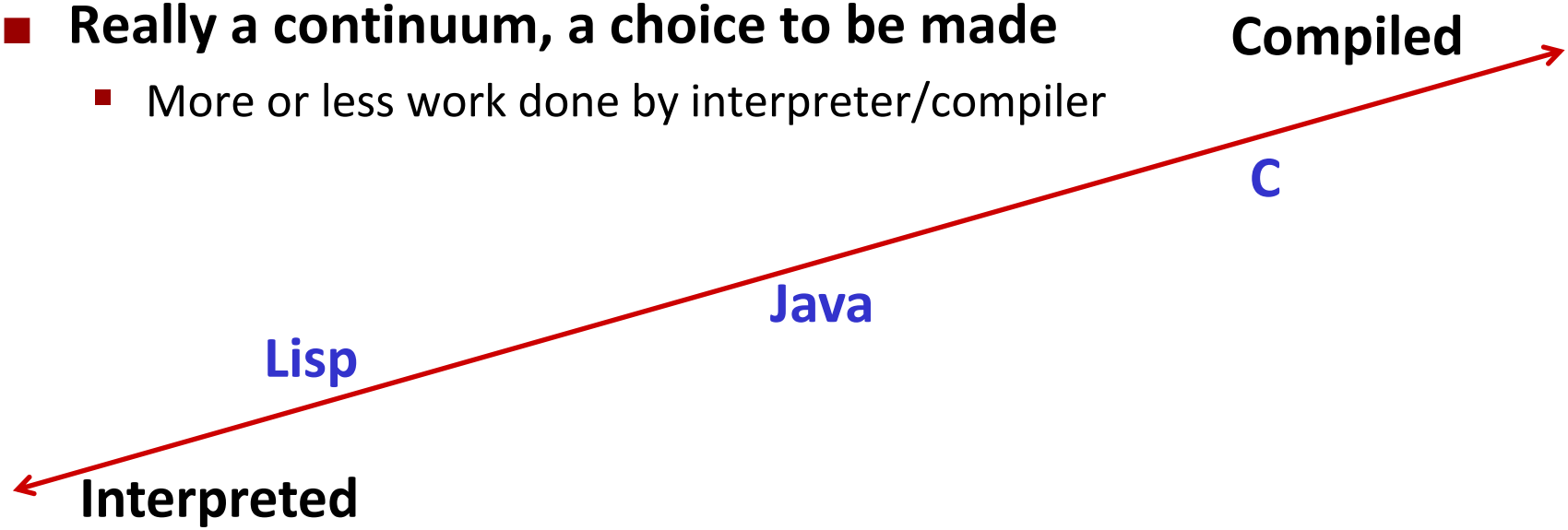
- Data representations in Java
- Pointers and references
- Method calls
- Virtual machines and runtime environment

Implementing Programming Languages

- **Many choices in how to implement programming models**
- **We've talked about compilation, can also *interpret***
  - Execute line by line in original source code
  - Less work for compiler – all work done at run-time
  - Easier to debug – less translation
  - Easier to run on different architectures – runs in a simulated environment that exists only inside the *interpreter* process
- **Interpreting languages has a long history**
  - Lisp – one of the first programming languages, was interpreted
- **Interpreted implementations are very much with us today**
  - Python, Javascript, Ruby, Matlab, PHP, Perl, …

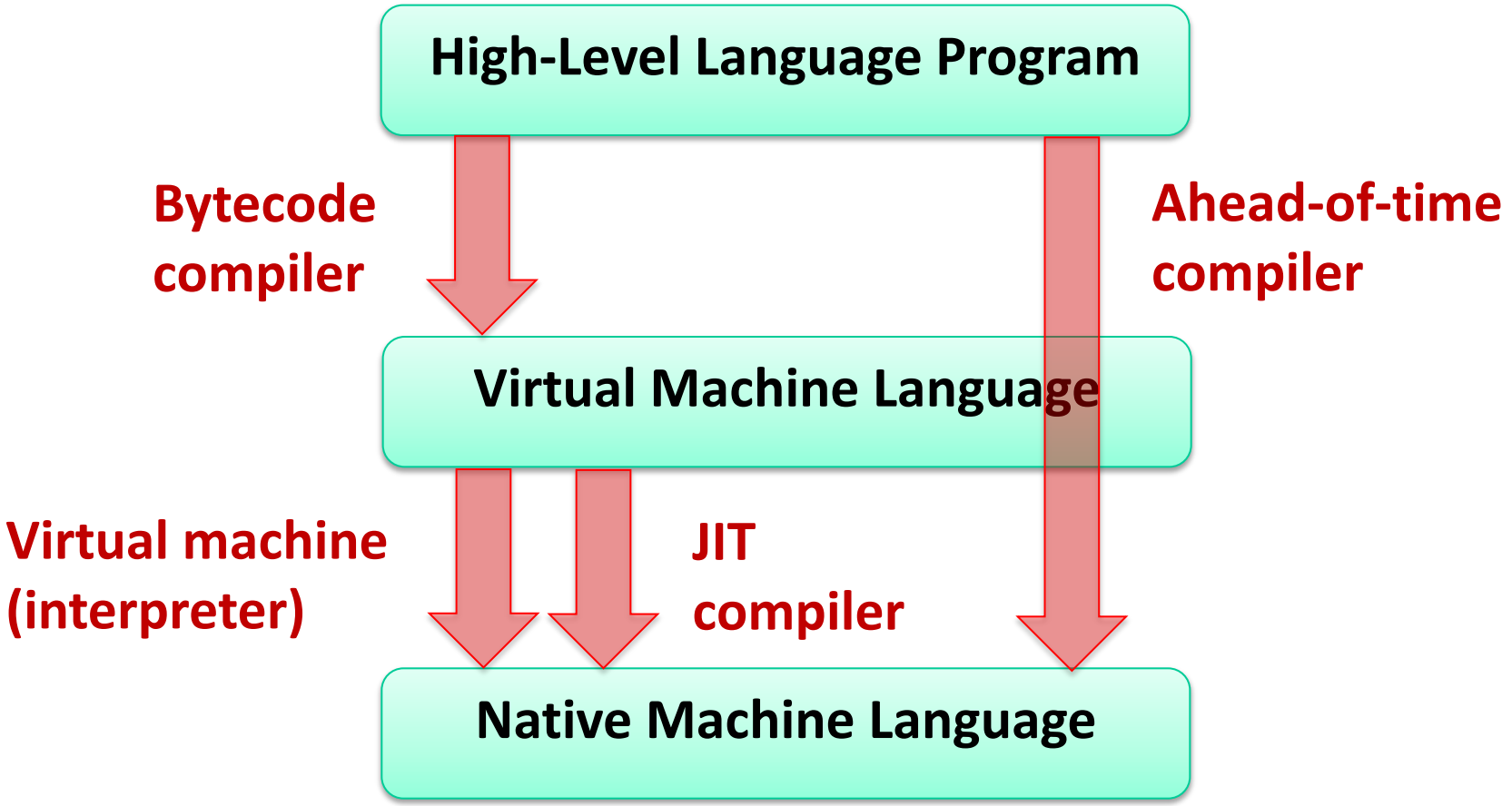
Interpreted vs. Compiled

- **Really a continuum, a choice to be made**
  - More or less work done by interpreter/compiler



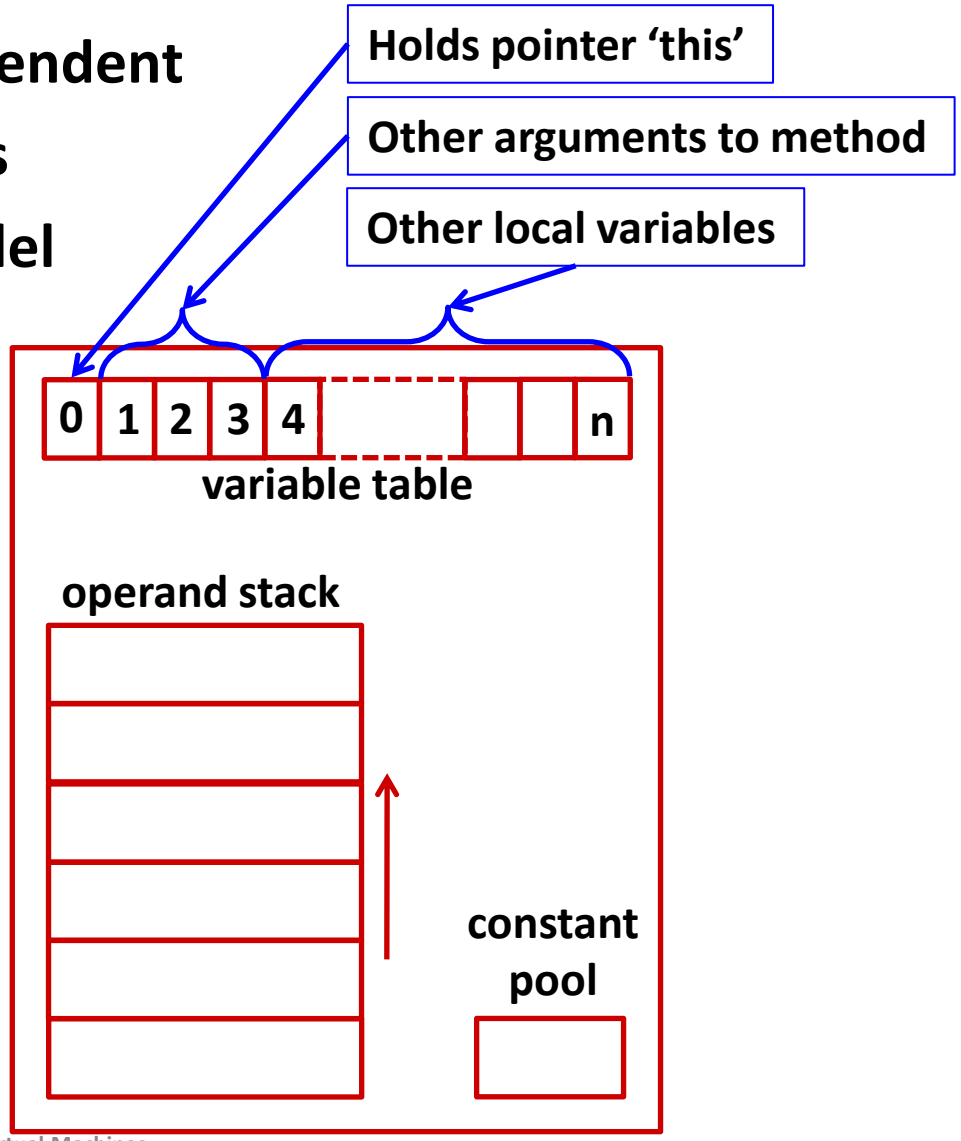
- **Java programs are usually run by a *virtual machine***
  - VMs interpret an intermediate, "partly compiled" language called *bytecode*
- **Java can also be compiled ahead of time (just as a C program is) or at runtime by a *just-in-time (JIT) compiler***

Virtual Machine Model



Java Virtual Machine

- Makes Java machine-independent
- Provides strong protections
- Stack-based execution model
- There are many JVMs
  - Some interpret
  - Some compile into assembly
  - Usually implemented in C



JVM Operand Stack Example

'i' stands for integer,
'a' for reference,
'b' for byte,
'c' for char,
'd' for double, …

No knowledge of registers or memory locations (each instruction is 1 byte – bytecode)

```
iload 1     // push 1st argument from table onto stack
iload 2     // push 2nd argument from table onto stack
iadd        // pop top 2 elements from stack, add together, and
            // push result back onto stack
istore 3    // pop result and put it into third slot in table
```

```
mov 0x8001, %eax
mov 0x8002, %edx
add %edx, %eax
mov %eax, 0x8003
```

A Simple Java Method

```
Method java.lang.String getEmployeeName()

0 aload 0       // "this" object is stored at 0 in the var table

1 getfield #5 <Field java.lang.String name> // takes 3 bytes
                // pop an element from top of stack, retrieve its
                // specified field and push the value onto stack.
                // "name" field is the fifth field of the class

4 areturn       // Returns object at top of stack
```

In the .class file:

2A	B4	00	05	B0

http://en.wikipedia.org/wiki/Java_bytecode_instruction_listings

Class File Format

- **Every class in Java source code is compiled to its own class file**
- **10 sections in the Java class file structure:**
  - Magic number: 0xCAFEBABE (legible hex from James Gosling – Java's inventor)
  - Version of class file format: the minor and major versions of the class file
  - Constant pool: set of constant values for the class
  - Access flags: for example whether the class is abstract, static, etc.
  - This class: The name of the current class
  - Super class: The name of the super class
  - Interfaces: Any interfaces in the class
  - Fields: Any fields in the class
  - Methods: Any methods in the class
  - Attributes: Any attributes of the class (for example the name of the source file, etc.)
- **A *.jar* file collects together all of the class files needed for the program, plus any additional resources (e.g. images)**

Other languages for JVMs

- **JVMs run on so many computers that compilers have been built to translate many other languages to Java bytecode:**
  - AspectJ, an aspect-oriented extension of Java
  - ColdFusion, a scripting language compiled to Java
  - Clojure, a functional Lisp dialect
  - Groovy, a scripting language
  - JavaFX Script, a scripting language targeting the Rich Internet Application domain
  - JRuby, an implementation of Ruby
  - Jython, an implementation of Python
  - Rhino, an implementation of JavaScript
  - Scala, an object-oriented and functional programming language
  - And many others, even including C

Microsoft's C# and .NET Framework

- C# has similar motivations as Java
- Virtual machine is called the Common Language Runtime; Common Intermediate Language is the bytecode for C# and other languages in the .NET framework