Introduction to Computers and Java
HARDWARE

SOFTWARE
HARDWARE

CPU

Memory
HARDWARE
HARDWARE
HARDWARE
HARDWARE

RAM
## Memory

<table>
<thead>
<tr>
<th>Addresses</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>256</td>
<td>01000001</td>
</tr>
<tr>
<td>260</td>
<td>01000010</td>
</tr>
<tr>
<td>264</td>
<td>01000010</td>
</tr>
<tr>
<td>268</td>
<td>01000001</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
• Bit – a digit with a value of 0 or 1
• Byte – Consists of 8 bits
• Address – The numbered location where each byte resides
• All data is encoded as a 0 or 1
  – Everything is a number
• When more than 1 bytes is needed then several adjacent addresses are used
**Files and Folders**

- **File** – large group of bytes stored in secondary (auxiliary) memory
  - Files have names
  - Most files have extensions
- **Folder (Directory)** – group together multiple files
- **Java programs are stored in files**
  - Source code have the extension “.java”
  - Byte-Code have the extension “.class”

<table>
<thead>
<tr>
<th>Memory</th>
<th>Addresses</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>256</td>
<td></td>
<td>01000001</td>
</tr>
<tr>
<td>260</td>
<td></td>
<td>01000010</td>
</tr>
<tr>
<td>264</td>
<td></td>
<td>01000010</td>
</tr>
<tr>
<td>268</td>
<td></td>
<td>01000001</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Running Software

Secondary

Main

CPU
CPU
Main
Secondary

Running Software

Load
Run Code
Running Software

- Secondary
- Main
- CPU

Flow:
- Load from Secondary to Main
- Run Code from Main to CPU
- Store Info from CPU to Main
**SOFTWARE**

Algorithm (ˈælɡəˌrɪðm) noun
A set of instructions to solve a problem
Program (prōˌgram) noun
A set of instructions for a computer to follow.
Programming Languages
Programming Languages
LOW LEVEL
LOW LEVEL

FUNCTION: INCH - Input character
INPUT: none
OUTPUT: char in acc A
DESTROYS: acc A
CALLS: none
DESCRIPTION: Gets 1 character from terminal

C010 B6 80 04 INCH  LDA A ACIA  GET STATUS
C013 47                ASR A    SHIFT RDRF FLAG INTO CARRY
C014 24 FA             BCC    INCH    RECEIVE NOT READY
C016 B6 80 05          LDA A ACIA+1 GET CHAR
C019 84 7F             AND A #$7F  MASK PARITY
C01B 7E C0 79          JMP OUTCH  ECHO & RTS
### LOW LEVEL

**Machine Code**

**Programming Languages**

<table>
<thead>
<tr>
<th>CODE</th>
<th>OP CODE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>C010 B6 80 04</td>
<td>INCH</td>
<td>GET CHAR</td>
</tr>
<tr>
<td>C013 47</td>
<td>ASR A</td>
<td>SHIFT RDRF FLAG INTO CARRY</td>
</tr>
<tr>
<td>C014 24 FA</td>
<td>BCC INCH</td>
<td>RECVIEVE NOT READY</td>
</tr>
<tr>
<td>C016 B6 80 05</td>
<td>LDA A ACIA+1</td>
<td>GET CHAR</td>
</tr>
<tr>
<td>C019 84 7F</td>
<td>AND A #$7F</td>
<td>MASK PARITY</td>
</tr>
<tr>
<td>C01B 7E C0 79</td>
<td>JMP OUTCH</td>
<td>ECHO &amp; RTS</td>
</tr>
</tbody>
</table>
High Level Programming Languages
High Level Programming Languages

Java
C#
C++
Python
Programming Languages

High Level
High Level

Nouns and Verbs
High Level Syntax
Programming Languages

Compiler
Project
Project

Classes

Methods
Classes

Methods

• Source Code in files with “.JAVA” extension
• The filename must MATCH the name of the class
• Everything is an “Object”
Compilation

Java Source Code (.JAVA)

Running
Java

Compilation

Java Source Code (.JAVA)

Java Compiler

Running
Java Source Code (.JAVA) → Java Compiler → Intermediate Byte Code (.CLASS) → Running
Compilation

Java Source Code (*.JAVA)

Java Compiler

Intermediate Byte Code (*.CLASS)

Running

Java Virtual Machine (JVM)
Running

Compilation

Java Source Code (.JAVA)

Java Compiler

Intermediate Byte Code (.CLASS)

Running

Java Virtual Machine (JVM)
• Running (Executing) – Is when the computer is following the instructions in a program
• Statement – An instruction to the computer. Most end with a semicolon “;”
  – int i;
  – double j;
• Syntax – The grammar rules for a programming language
• Comments – Code ignored by the compiler that is generally used to explain the code.
  – Single line comments use the “//”
  – Multiline comments use “/ * */”
• Arguments – Information found inside of parenthesis “()” that provide information for methods or other statements
  – if(<<ARGUMENT>>)
  – System.out.println(<<ARGUMENT>>)
• Bug – an error in a program
• Debugging – the process of removing errors
• There are 3 major classes of errors
  – Syntax
  – Runtime
  – Logic
• Syntax Errors—Grammatical mistakes in a program
  – Very common at first
  – These prevent a program from compiling and running
  – Common
    • Missing a Semicolon at the end of a statement
    • Using the wrong, misspelled, repeated, or incorrectly capitalized identifier
    • Mismatched curly braces “{“}, parenthesis “(“), single quotes “‘“), double quotes “““), etc.

```
int i  //Missing a semicolon
double j = 0.0;
J = 1.0; //Wrong identifier
System.out.println(i; //Missing parenthesis
```
- **Runtime Errors** – Errors detected when the program is running but not during compilation
  - The code will compile but crashes at some point
  - When this happens the computer detects the error and terminates the program
- Common
  - Divide by 0
  - Calling a method from a NULL object
  - An index going outside the bounds of an Array

```java
double j = 1.0 / 0.0; // Divide by 0
Scanner keyboard; // This has not been constructed so it is NULL
keyboard.hasNextLine(); // Calling method from NULL object
int[] a = {5, 4, 3, 2, 1}; // An array
a[5] = 2; // Index 5 is out of bounds
```
• Logic Errors – despite the program compiling and running, it produces incorrect results
  – Arguably the hardest to fix
  – Common:
    • Order of operations error
    • Round off mistakes
    • Incorrectly using types or methods

```java
double f = 72.0; // 72 degrees fahrenheit
double c = f - 32.0 * 5.0/9.0; // Order of operations error
double c2 = (f-32.0)*(5/9); // 5/9 = 0 so it will always be 0
double c3 = (f-32.0)*(5.0/9.0); // Correct
```
HELLO WORLD!