





MAGICAL







INPUT



INPUT









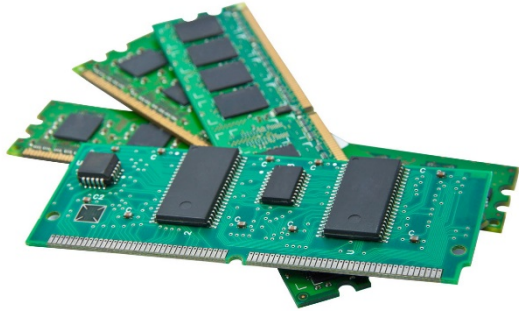


HARDWARE

CPU



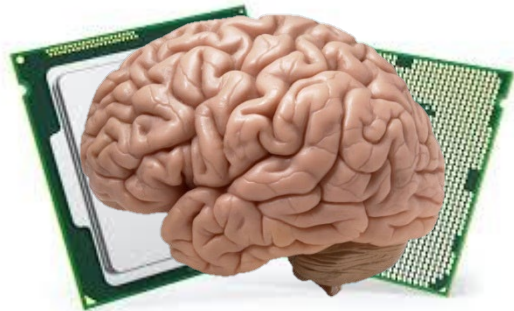
Memory



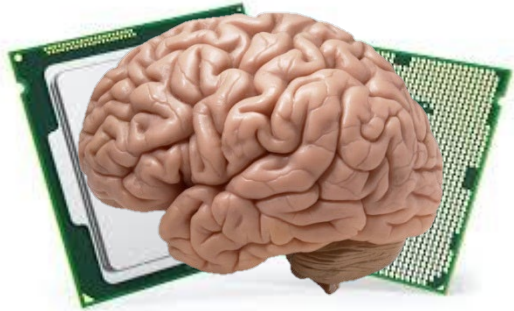
HARDWARE



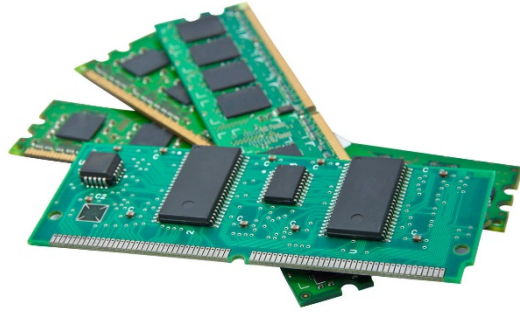
HARDWARE



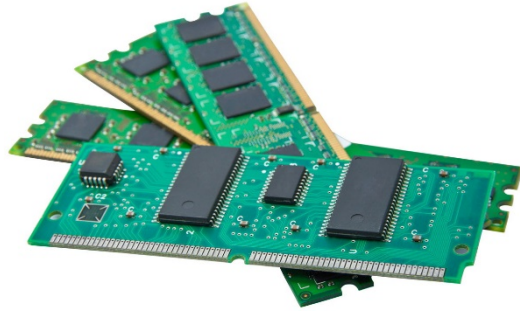
HARDWARE



HARDWARE



HARDWARE



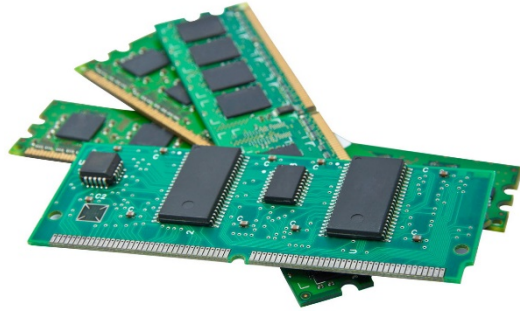
RAM

HARDWARE

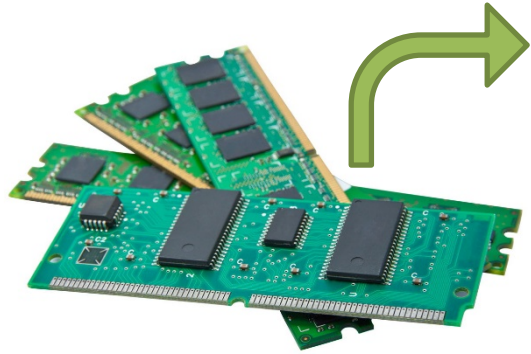


Secondary

HARDWARE



HARDWARE



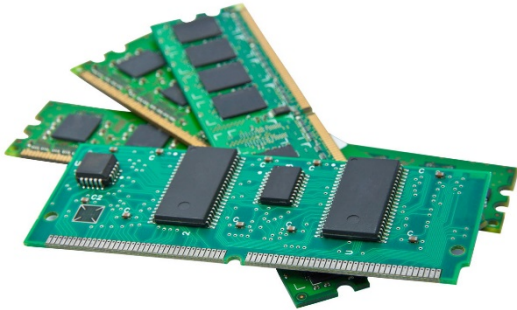
Memory	
Addresses	Values
...	
256	01000001
260	01000010
264	01000010
268	01000001
...	

Running Software

Secondary



Main



CPU

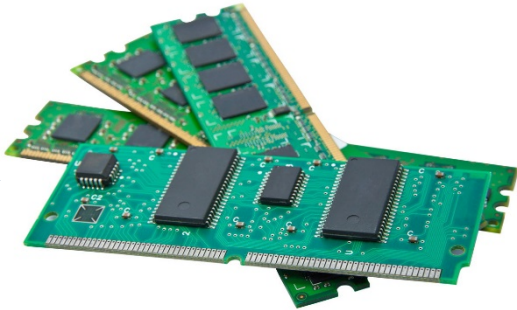


Running Software

Secondary



Main



CPU



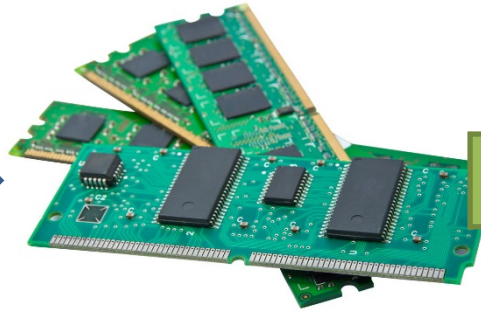
Running Software

Secondary



Load

Main



Run Code

CPU



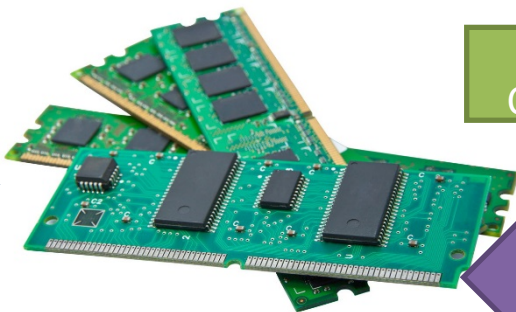
Running Software

Secondary



Load

Main



CPU

Run Code


Store Info







Algorithm ('algə,riT_Həm) noun
A set of instructions to
solve a problem



Program (prō,gram) *noun*
A set of instructions for a computer to follow.



Programming Languages

Programming Languages





LOW LEVEL

Programming Languages

LOW LEVEL

```

1001 1001001011 01 0 01010 010110 01100 00010 01010101110 001010 00100 000111
01 1010 001 111010 001001010 101011000101010 010 10001001 010 001 01001 0 100010
00 01 1 01 1010 010 00101 110100 1 1000 010 11 0 0101000 10110 01011 010010010101
001 10101001011 10 01001001 0001011 00 010100 100100 00100 01001010010010 010
10 10 0 00 011001 0 010101011 010110010010110101011101 00 001010010010001111
0 011100110 0 0010 0010 0 101000101010 001101010100001011010101 100100 01010101
00 010101000 0 1 010 00 00 010010 111000 0 0100100000100 01010 1001001 010010
001011 1 11010 010 01010110000101001000110101000 00110101 1100 001010100
0 010101000 1110 0100 0010 010111 00101000100100 10010010 0100100000001
001011 01000 11 0100010 10100 010110000101000 0 000100 000100010010000 0
0010111 011010001 0010011010000101 01010 1 1010 0100001011010 01110 1001001010
100 010010 101 10 1 000 0 010 011 1 10001000101010 01111 1000 001 0100 01000111
0 1 10110001 1 1010100 0 10100 0101 00 1 1010001000010100 0100000001 01 0010
0 10 1 10 110 0100 0100101 0 0001000 00 110101 100010 1101010 110 10 10010101
0010111 01101010010 001 011010001 00 0100110101 0 0011101010 10 10 100 0101
0 0 1 101010 11010100 00100101011100010 01000100 000100 001010010 100 0010 010
0 1 10110 0010111001 1001 0101 01 101110010 0100 100 000100100100010000 00 0 010
01 11 10 1010 0 10 0 1 10100 00010 0 10 0 110 0100000 01110101 100 00010 1
0 0101010 011 101010 10 10 00001 10001010 0 1001 001001010 0 0 0010 0 10
001011101110101001100 01010 000100 0100 10 01010000 0110101011 01 00 10
10 1010100 111010 010 0 0111010100 00 10110 0101011101 01001001 100 00 11
010 01 010101 101010 100101 01010 000101010 1001000 00 00101001001010 1 0010
0 10 111011 010 001100101 11 10 010 001 100 110 1 1 000111 010 01 10010 1001 101
100100100101110 00101010 11 01 1100 000 011 010 0111010 0100100001 0000111
001 11 101 0010101000 0 01101 0 10 010100 1010 010000 0110101 110 0 1 010 01
00 010101 0 111010 010010 0 0 0 11 00 01 1 001 0100010 00101 01 0 1000001

```

```

*****
* 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	RECIEVE 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

Programming Languages

LOW LEVEL

```

1001 100100101 01 0 01010 0101110 01100 00010 01010101110 001010 00100 000111
01 1010 001 111010 001001010 10101100001010 010 10001001 010 001 01001 0 10010
00 01 1 01 1010 010 00101 110100 1 1000 010 11 0101000 10110 01101 0101000101
001 10101001011 10 01001001 0001011 00 010100 100100 00100 01001 010010010 010
10 10 0 00 0110010 0 01010101 010110010010110101011101 00 0010100 1001000111
0 011100110 0 0010 0010 0 1010001010010 001101010100000111010101 100100 010101
00 010101000 0 010 00 00 010010 111000 0 0100100000100 01010 1001001 010 0010
001011 1 11010 010 01010110000100010001101010000 011010101 1100 001010 0
0 01010100 01110 1000 010 01011 00101001000 1001000 00100100000001
00101 01000 11 0100 0 0100 01000001000000 0 0100001000100000 0
0010111 0110101001 0100 010 0100 10 01110 1001001010
100 010010 101 10100 0100 0100 00 001 0100 01000111
0 1 10110001 1 1 10100 0 101 0 101 010010010001 0100 010000001 01 0010
0 10 10 110 0100 01001010 0 000 0100 00 110101 1000 10 11010 110 10 100 0101
0010111 01101010010 001 0111 001 00 01001 0101 0 001 11 01010 10 10 100 0101
0 0 1 1010 11101000 001 01000 00100010 100 0010 010
0 101010 001011101 1001 01 01 0010001000100010001000 0 0 010
01 11 10 1010 0 10 0 1 10 00000000000000000000011101 100 000010 1
0 0101010 011 101010 10 10 000000000000000000000000000000 0 0 0010 0 10
001011101110101001100 010 010 0100 01001 0 01010000 0110101011 01 000 10
10 1010100 111010 010 0 0111010100 00 1010 0101011101 01001001 0100 00 11
010 01 010101 10101 100101 01010 000101010 1001000 00 0010100001010 1 0010
0 10 111011 010 001100101 11 10 01 001 100 110 1 1 000111 10 01 10010 1001 101
100100100101101 00101010 11 01 1000 000 011 010 0111010 0101000001 000011
001 11 101 010101000 01101 0 10 010100 1010 010000 0110101 110 0 1 010 01
00 010101 0 111010 010101 0 010 11 00 01 10001 000010 00101 01 0 1000001

```

Machine Code

```

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

```

Assembly

```

C010 B6 80 04 INCH INCH GET STATUS
C013 47 ASR A SHIFT RDRF FLAG INTO CARRY
C014 24 FA BCC INCH RECIEVE 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

```




Programming
Languages

High Level



Programming Languages



High Level





Programming
Languages

High Level



High Level

Nouns and Verbs



High Level

Syntax

Programming Languages



Programming Languages



Compiler







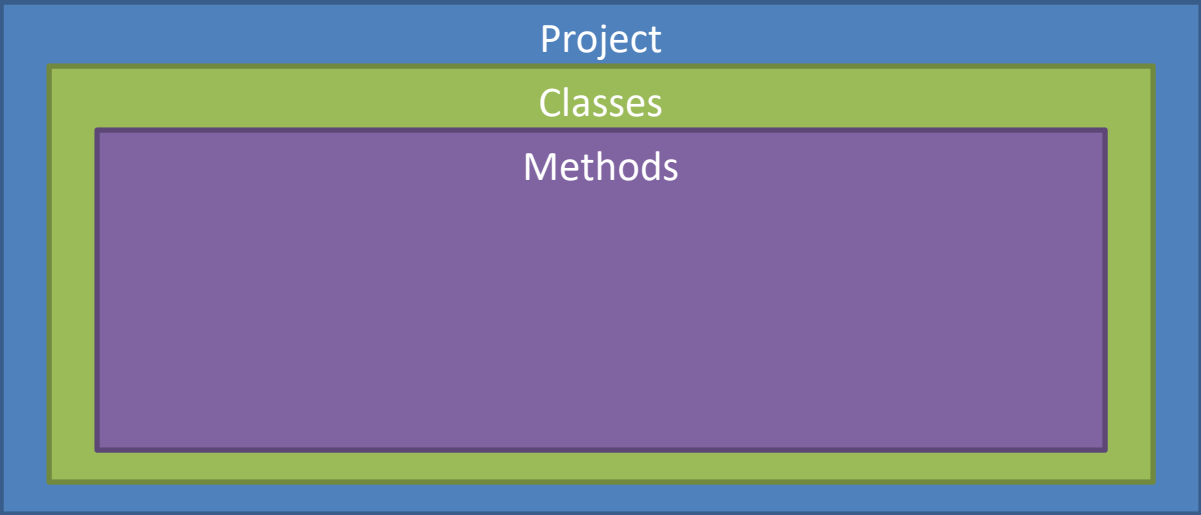
Project



Project

Classes

A diagram illustrating the relationship between a project and its classes. It features a large blue rectangular border. Inside this border, at the top center, is the word "Project" in white text. Below "Project" is a large green rectangular area. At the top left corner of this green area, the word "Classes" is written in white text.





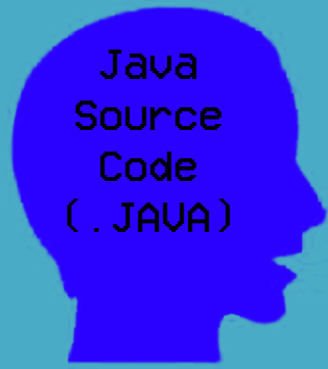
Classes

Methods

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



Compilation

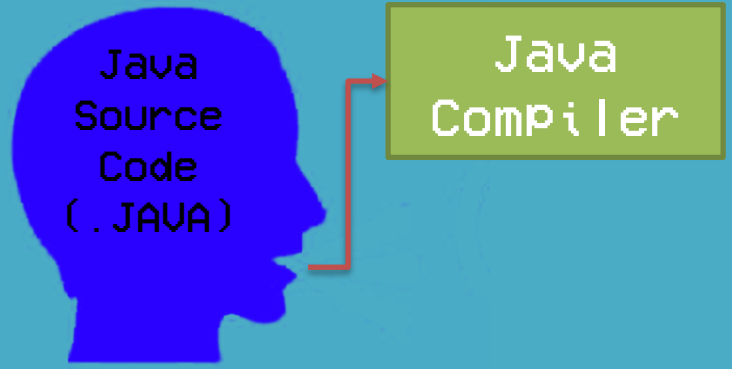


Running





Compilation

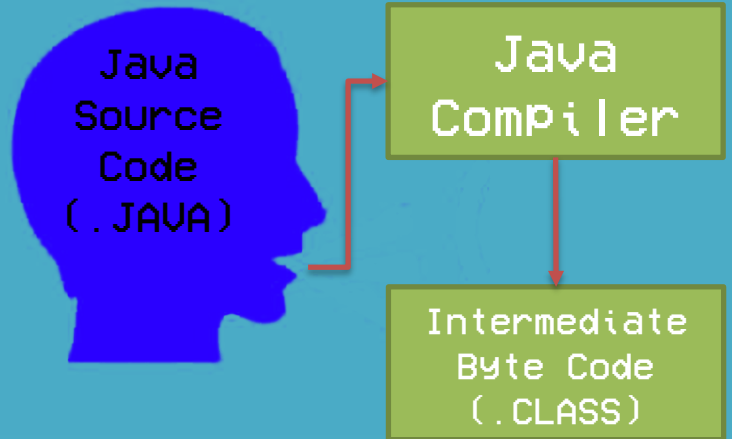


Running





Compilation

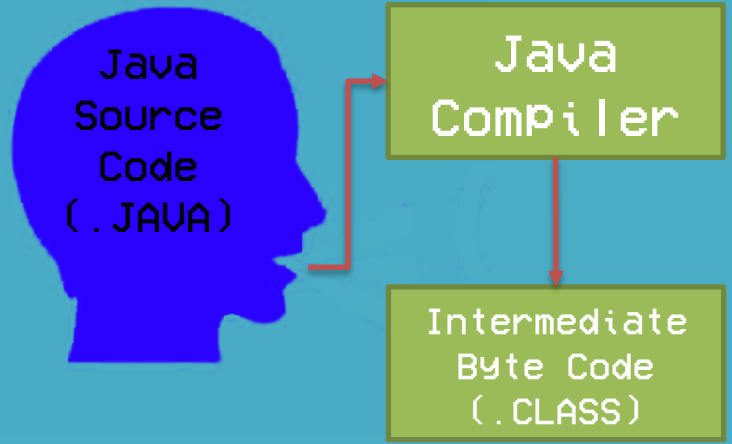


Running





Compilation



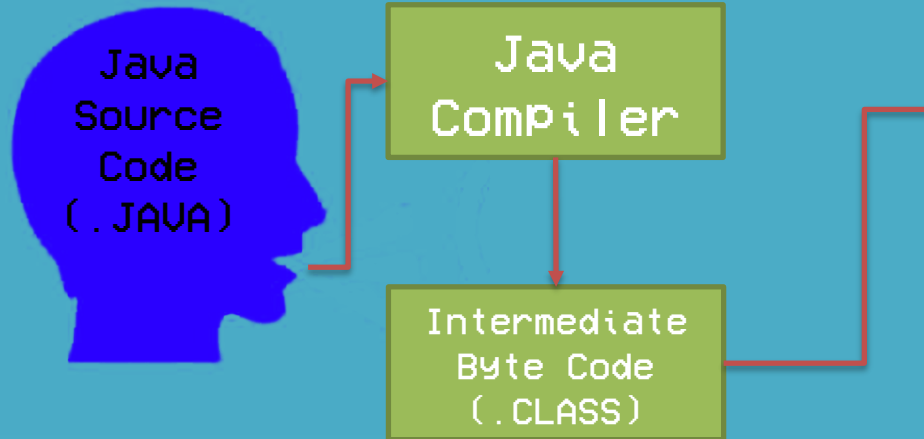
Running

Java Virtual Machine (JVM)





Compilation



Running

