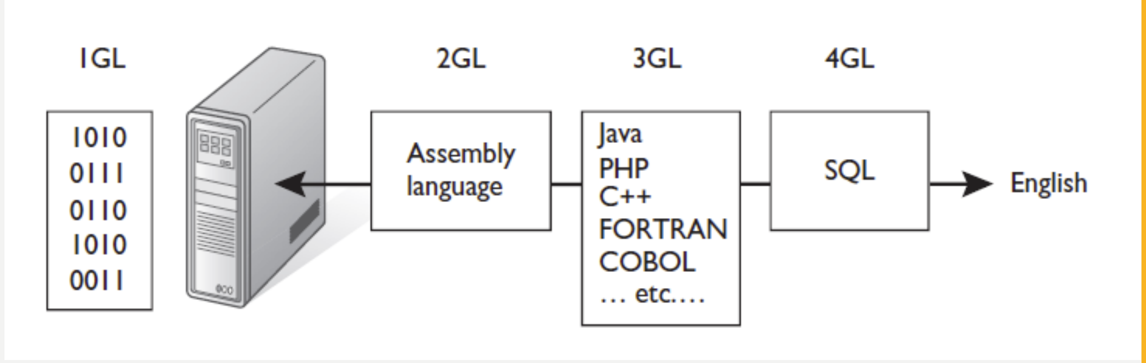
PROGRAMMING FUNDAMENTALS, PART 2

WHAT ALL PROGRAMMERS KNOW

COMPROMISES: MACHINE VS HUMAN



Django, React, Flask, Flutter(?)

JavaScript, Python, Java, Dart

Objective C

C++

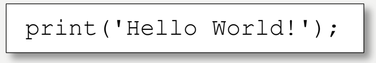
C

Assembly Language

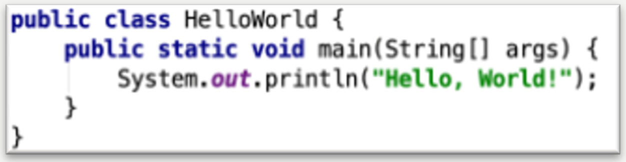
Machine Code

CPU

Frameworks

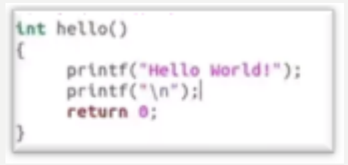


```
print('Hello World!');
```



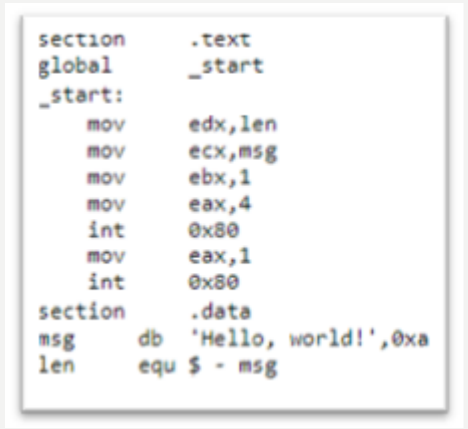
```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

High Level Languages

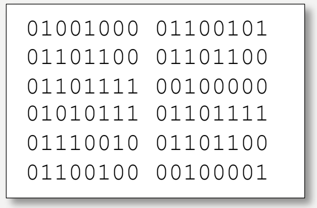


```
int hello()
{
    printf("Hello World!");
    printf("\n");
    return 0;
}
```

Low Level Languages



```
section     .text
global      _start
_start:
    mov     edx,len
    mov     ecx,msg
    mov     ebx,1
    mov     eax,4
    int     0x80
    mov     eax,1
    int     0x80
section     .data
msg     db  'Hello, world!',0xa
len     equ $ - msg
```



```
01001000 01100101
01101100 01101100
01101111 00100000
01010111 01101111
01110010 01101100
01100100 00100001
```

PROGRAMMING

HOW TO WRITE THE INSTRUCTIONS

- Source code

HOW TO CONVERT SOURCE CODE

- source code → machine code
- "interpret" and "compile"

HOW TO RUN THE MACHINE CODE

- EXECUTE

INTERPRETED AND COMPILED LANGUAGES

COMPILED

1. Source code
2. Run compiler; create the executable
3. Distribute and run the executable

INTERPRETED

1. Source code
2. Distribute
3. Interpret

Ready to run, but not portable

Optimized; runs faster

Private source code (proprietary)

Cross-platform (portable)

Simpler/faster to write source

Runs slower

Open source only

THIRD OPTION: HYBRID

- Both compiled and Interpreted
- Intermediate Language, a.k.a. Byte Code
- Just In Time (JIT) compilation

Compiled

- C
- C++
- Objective-C

Interpreted

- PHP
- JavaScript

Hybrid

- Java
- C#
- .NET
- Python