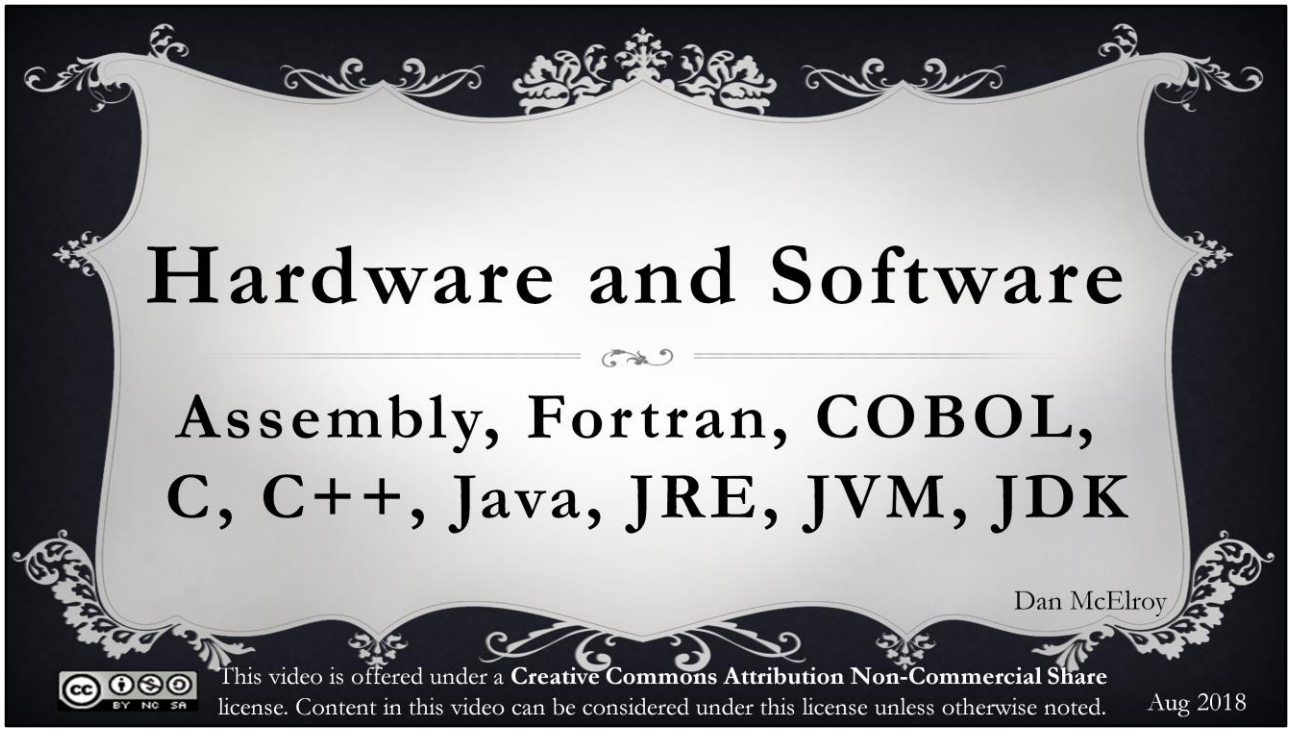



Hello Programmers! As a PROGRAMMER, you need to know how to develop programs from scratch, and many times you need to know how to maintain and fix programs that other people have written.

You are the inventor, the creator, the maintainer, the fixer and the documenter. As you perform your duties on the job, you may end up not only writing code but also going to many meetings and writing reports.

# Topics in the Discussion

- You are the programmer
- The Central Processing Unit (CPU)
- Hardware
- Software
- Programming Languages

Topics in the Discussion
You are the programmer
The Central Processing Unit (CPU)
Hardware
Software
Programming Languages

Photo by Robert Jack

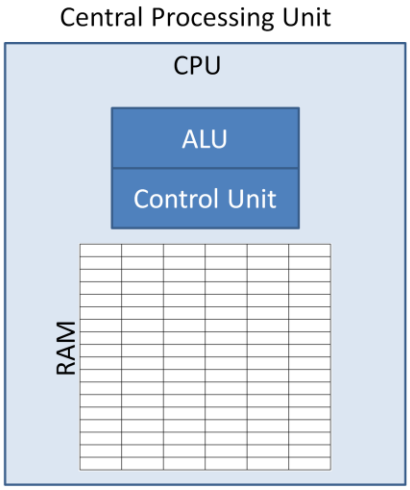
Users of a program are like the drivers of a car. The driver is provided with a few fairly simple to learn interfaces to drive an automobile that is actually a very complex machine. As programmers, we will be creating the user interface that may look simple when the program is running but is a lot more complex inside.

# Under the Hood

The driver of an automobile does not need to know whether the car has disk or drum brakes, but only needs to know how to stop the car. The driver does not need to know whether the engine has a carburetor or fuel injection but only need to know how to use the accelerator to make the car go. However, the automotive designer and technician needs to know both.

This video covers information about computer systems, that I call "Under the Hood". We will cover some of the hardware and software basics and a little on how to choose a programming language.

## Computer Hardware



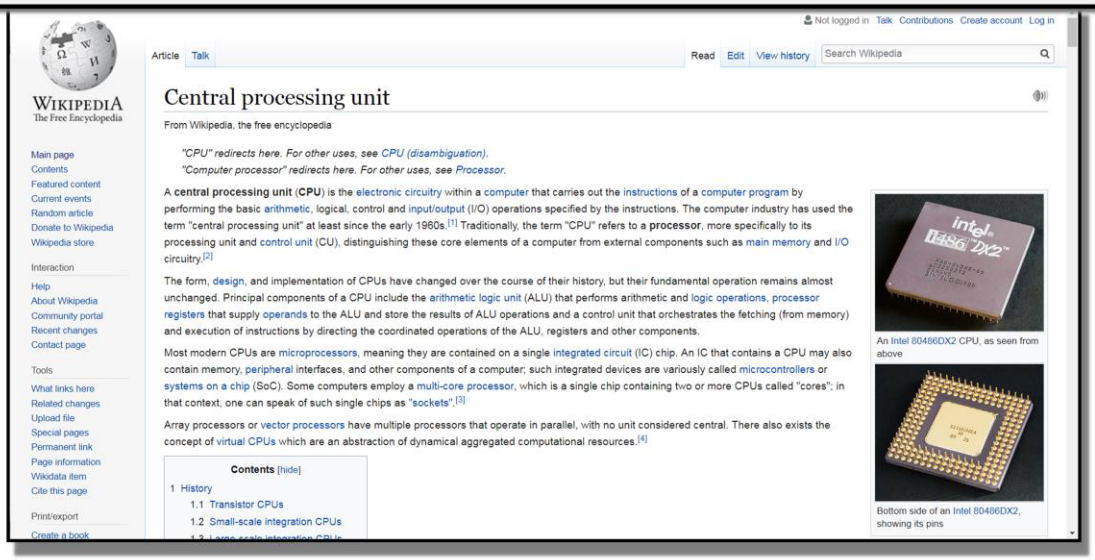

The Central Processing Unit is also known as the CPU. It contains the Arithmetic Logic Unit, called ALU, the control unit and the main memory, or Random Access Memory called RAM.

Both the program code and data are stored in the RAM.

The control unit contains a "Program Counter" that is used to step sequentially through memory, reading and executing instructions. This is called the Fetch-and-execute cycle. The instructions may read or store data from or to a memory location, cause the ALU to perform some type of math operation or even direct an input or output circuit to move data to or from the computer.

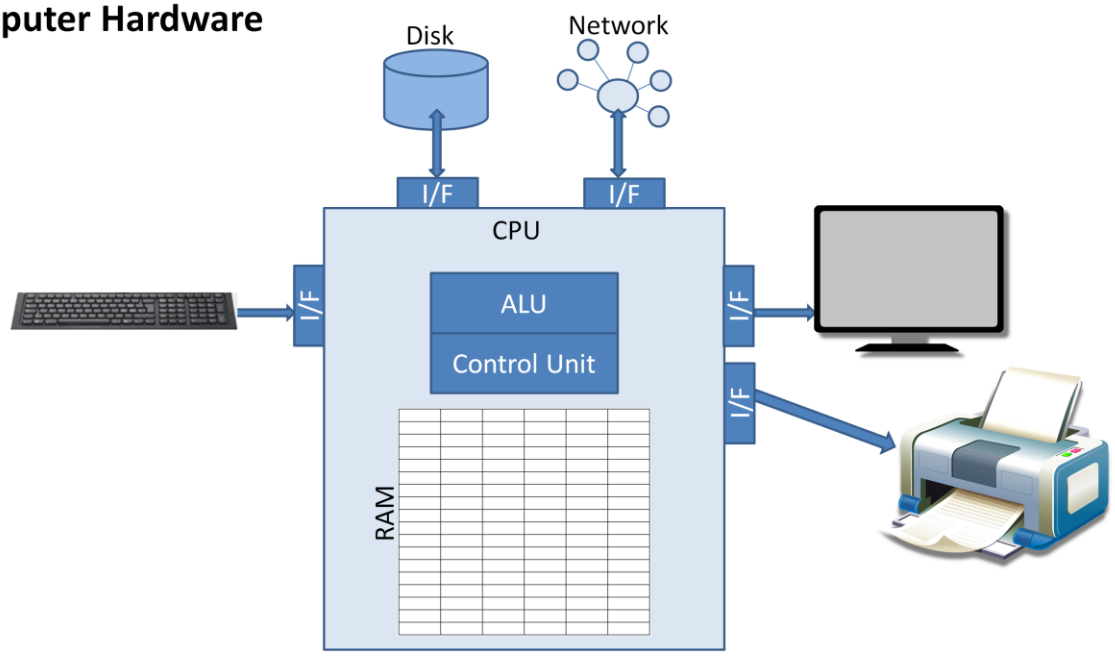
The ALU has the circuitry that performs arithmetic operations. The LOGIC circuity is used to make comparisons to determine whether a program should do one set of steps or another. For example, if a banking program is processing requests for cash from an ATM machine, a decision needs to be made about is there enough money in the account to process the request. The program either directs the ATM to dispense cash or send a message denying the request.

https://en.wikipedia.org/wiki/Central_processing_unit

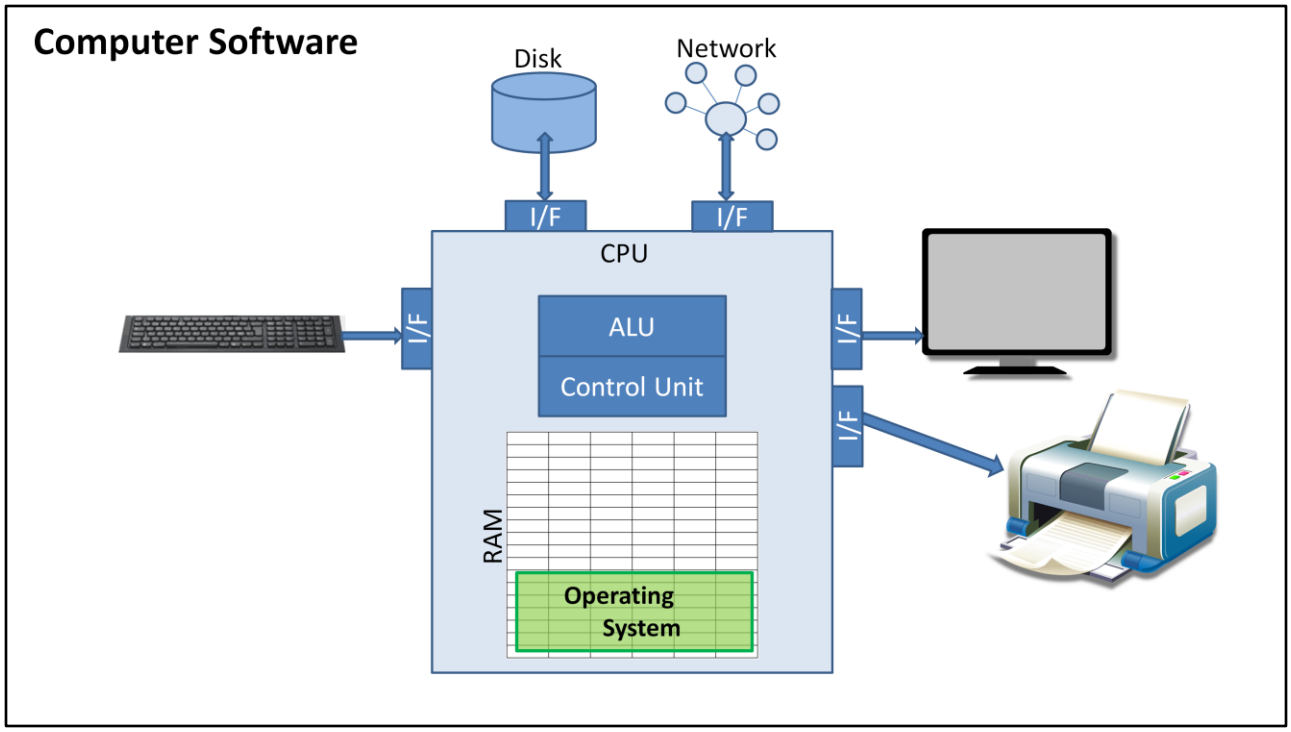



Wikipedia has a great article on the Central Processing Unit located at
**https://en.wikipedia.org/wiki/Central_processing_unit**
It is very interesting to read if you want to learn more about the details of the CPU.

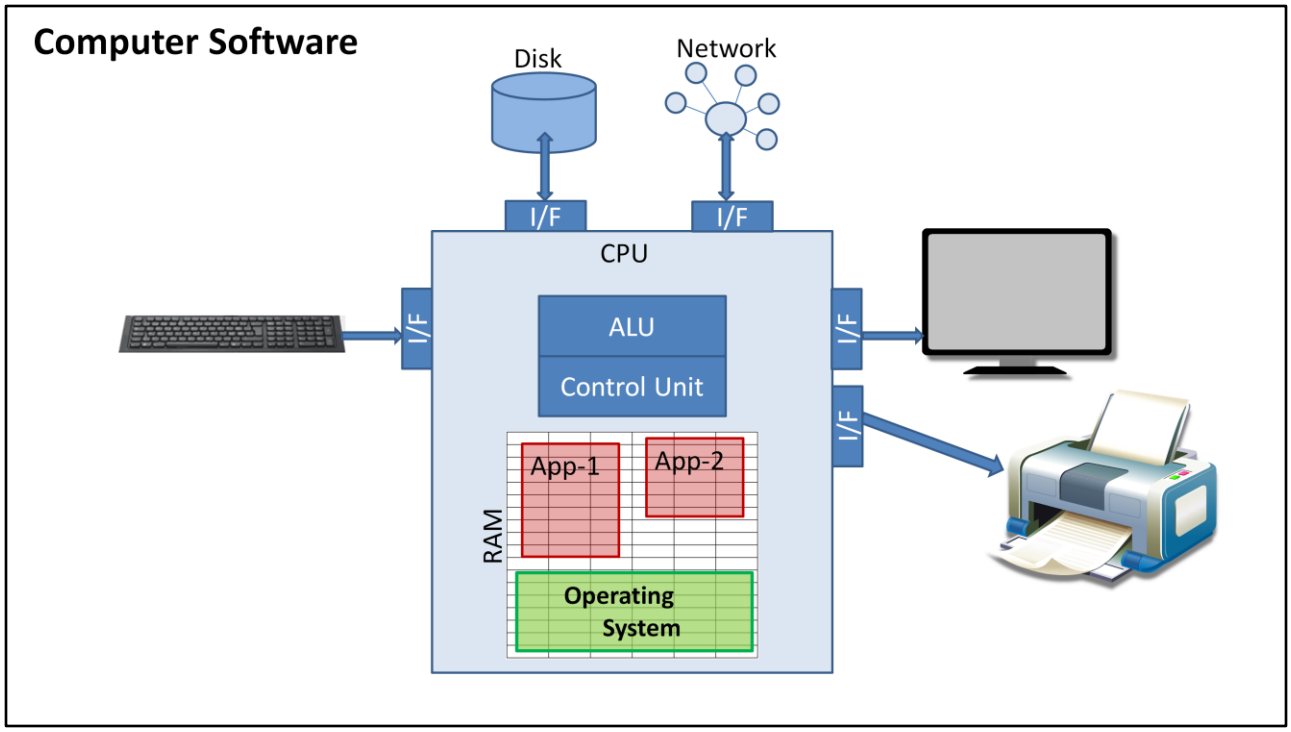
## Computer Hardware




Additional INTERFACE circuitry is provided to connect the CPU to external devices. These may be the keyboard, the display, a printer, disk or a network card. Nothing happens without the software directing the hardware to do something. As programmers, it our job to make the computer do something.



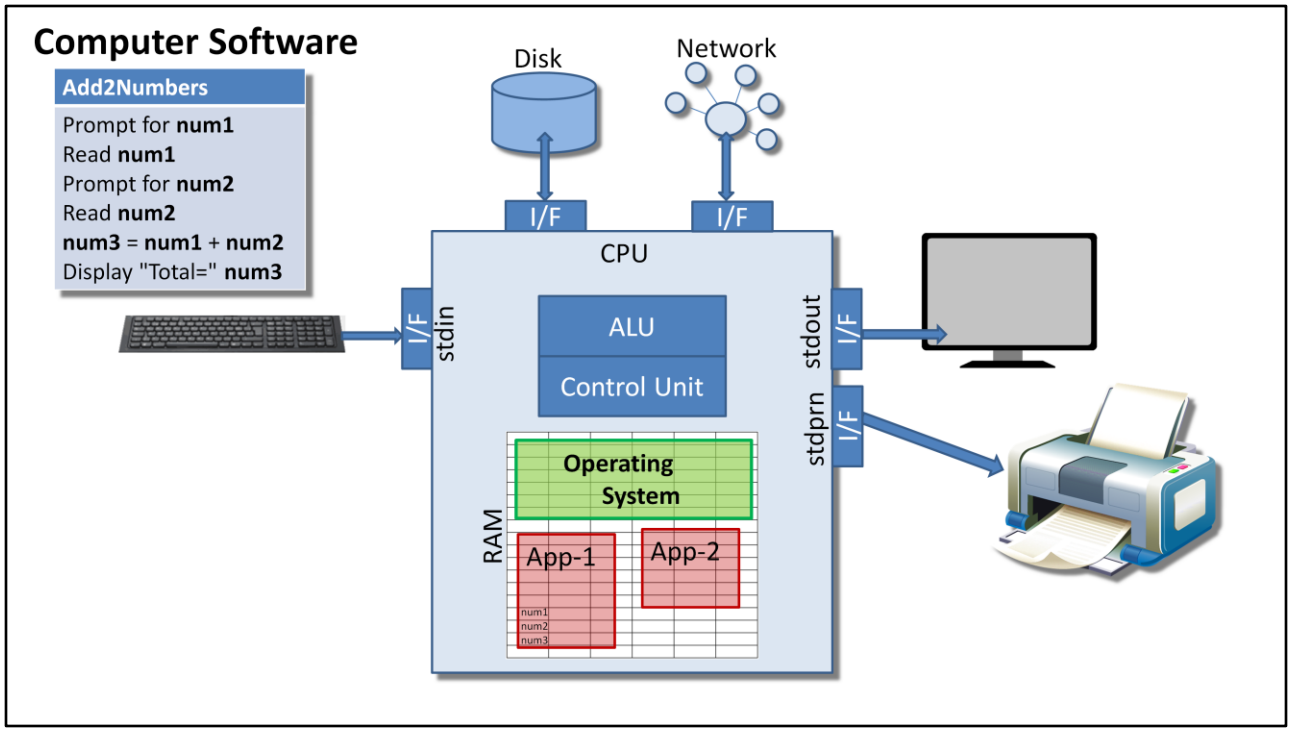

An operating system is not needed on very small systems such as software buried inside a computer keyboard, a microwave oven or a TV remote control.




As programmers started writing software for larger systems, they found that there was a lot of code that was similar for all of their programs. For example, the code for the keyboard, card reader, disk and the printer was the same. If one programmer was able to get these to work for a payroll program, the next programmer would ask to use these parts of the code in a program that did inventory control. A separation of duties began to form. All of the code to interface with devices is called the Operating System or OS, also known as 'System Software'. The code that actually processes the data is called 'Application Software'. The operating system defines an interface used by the application programs called 'Application Program Interface' or API.

Some of the main functions of the operating system are:
**Memory Management –** Set aside some RAM to load each program and prevent one program from stepping out of its reserved area onto another program's memory.
**Device Control –** The OS has exclusive access to each device. Application programs must request the OS to read or write data to a device. The part of the operating system that does this is called a 'Device Driver'. When a new device is added to the system, a device driver needs to be added into the OS to make it work.
**Disk Management –** The OS keeps track of where files and folders are located on the disk. Application programs access the disk by sending requests to the OS to open, read, write or close files.
**Process Management –** The CPU can perform one task at one time. The OS makes it look like it is doing more than one task by giving each one a little slice of time as it

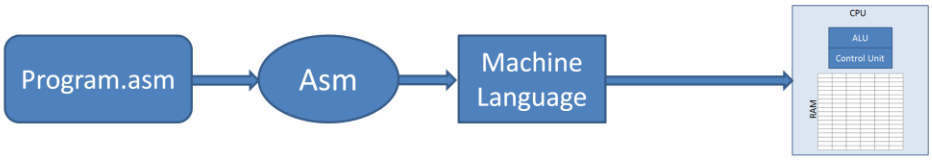
rotates through each task. Some tasks may be given higher priority.
**Print control –** The OS has a print spool (Simultaneous Peripheral Output Off Line) that will output data to the printer from one program at a time to keep the printouts from getting mixed within themselves.




Here is a sample program that just reads two numbers, adds them together and displays the sum. This program needs to set aside three variables in RAM and names them num1, num2 and num3.

Display a prompt for **num1**
Read a value from the keyboard and save it in the variable named **num1**
Prompt for **num2**
Read a value from the keyboard and save it in the variable named **num2**
Add the two numbers together and store in the variable num3 **num3** = **num1** + **num2**
Display "Total=" **num3**

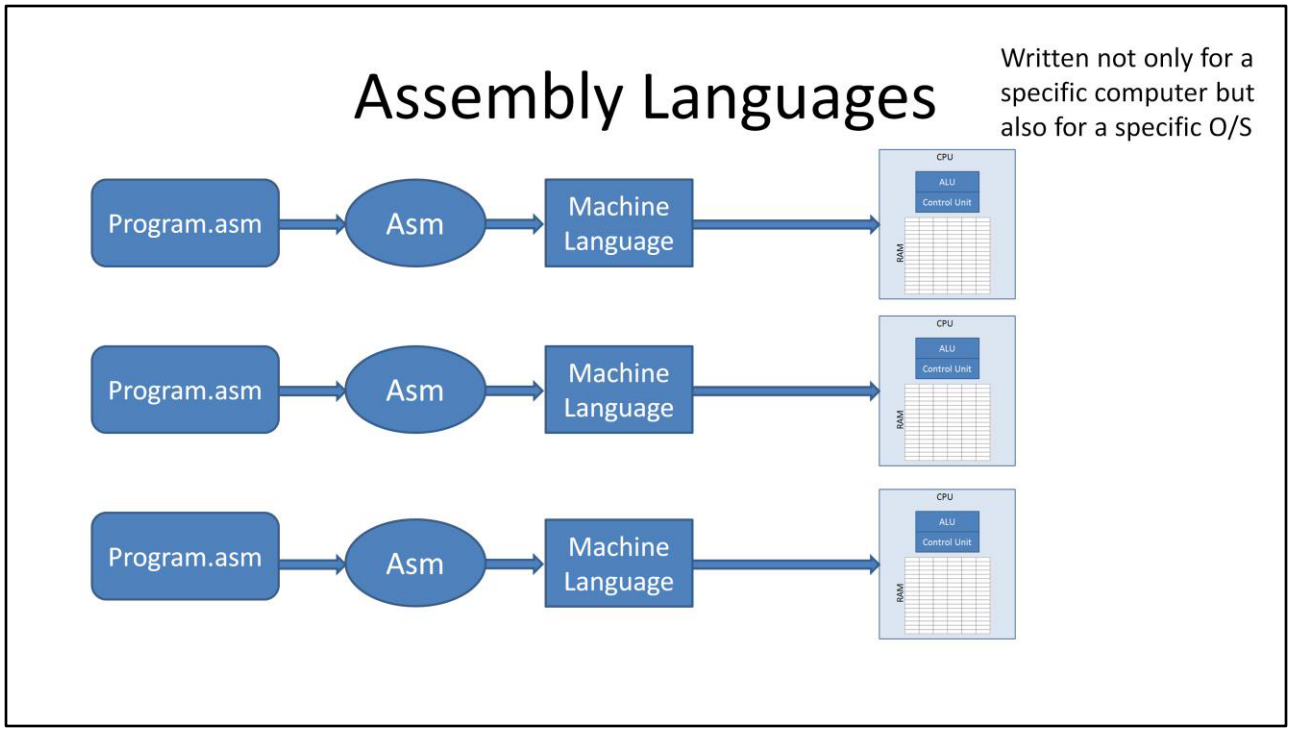
# Assembly Languages



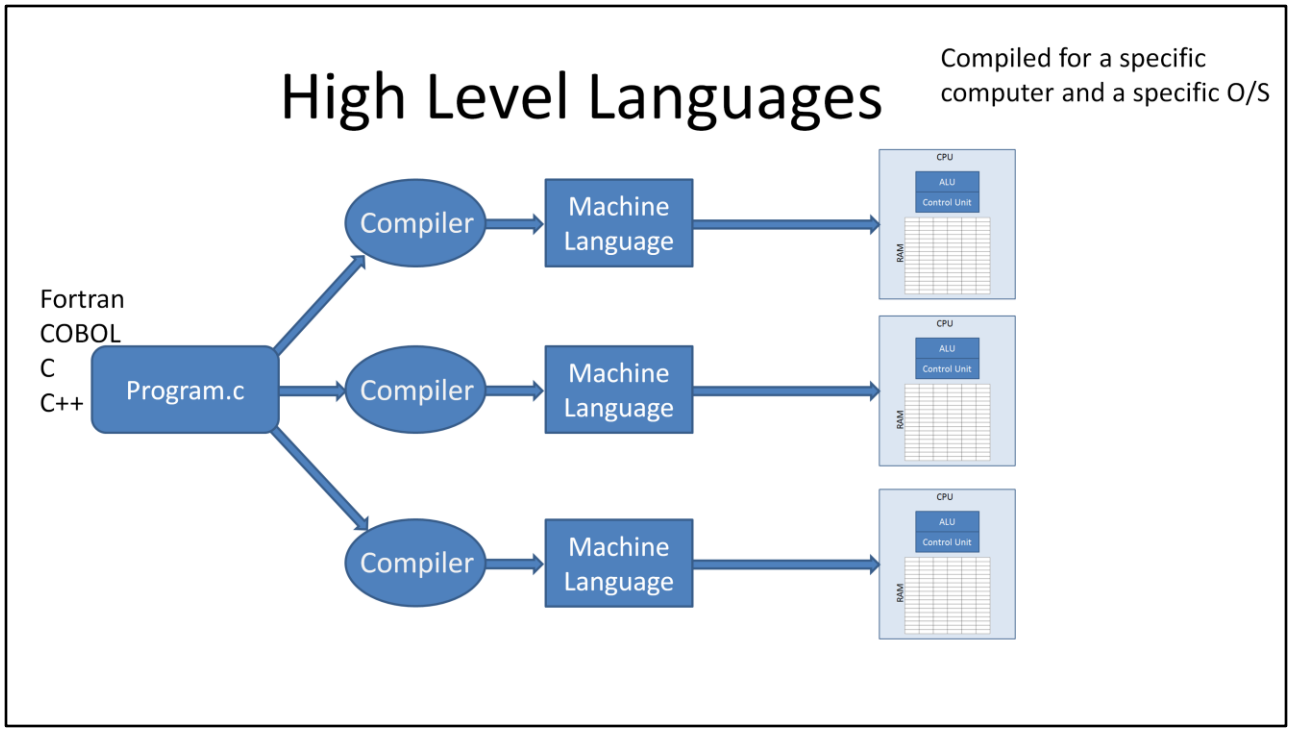

The earliest computer programs were written in machine language. Programmers needed to know the binary code needed for each instruction to save or load data from a selected memory location, make the ALU do something, or work with a device. Assembly languages were developed that gave names for each machine instruction. Programming became much easier. Some small programs are still written in machine language.

It could take several pages of code in and assembly language just to read two numbers represented as a collection of character codes from the keyboard, convert them individually into integers, add them together, convert the integer result back into a series of printable characters and them display them on the output device.

Writing in assembly language is very tedious, requires a lot of code to accomplish a small amount of work and as a result more prone to errors.



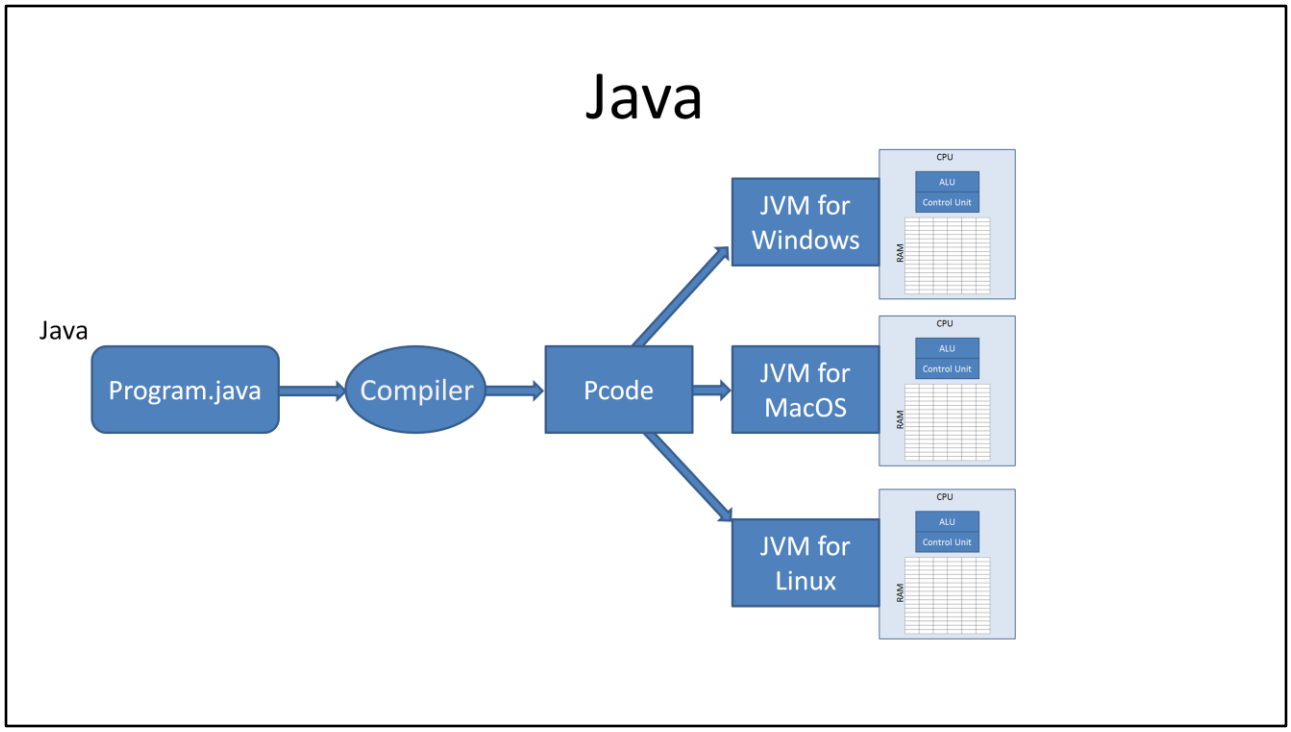

One of the problems with machine language is that the hardware logic for each computer is usually very different. This requires that a new set of software codes are required for each different machine. Once you know how to organize programs and write them in assembly language, moving from one set of instructions to another is not too difficult. I have already written programs in twelve different assembly languages and forgotten most the details in each. However, since I already know how to program in assembly language, it would not be too difficult for me to relearn any of the ones I have already used or even pick up a new one.




High-level languages were developed to make programming easier. Formulas could now be used instead of a large collection of assembly language code. The first high-level languages were a scientific language called FORTRAN and a business language called COBOL. FORTRAN was short for Formula Translator and COBOL was short for Common Business Oriented Language. Both became extremely popular for many years. A big advantage in using a high-level language is that once the program has been written all you needed is to compile it into the machine language for a new machine.

Many other high-level languages were developed for specific purposes. The C-language was developed at Bell Labs by Dennis Ritchie. The original Unix code was written in assembly language for the PDP-7 computer but became useless when he wanted to have Unix run on a PDP-8. So he created the C-language specifically for writing an operating system. C became popular in the universities because of Unix. Other high-level languages are derived from C, such as Objective-C, C++, Pascal, Python and C++.

A big disadvantage of most languages is that if a company wants to sell their software to different customers, they need to give away all their secrets by shipping the source code written in the high-level language, or get a machine similar to the customer's to recompile the program for their machine.




Java is a derivative of C and C++. It was developed by James Gosling, Mike Sheridan, and Patrick Naughton at Sun Microsystems. It was originally named Green, then renamed to Oak but those names were already taken. The language was eventually named Java after the coffee that Gosling was drinking. One of the biggest advantages is that Java is "Write Once, Run Anywhere" – A Java program is compiled into Pcode which is then run on a virtual machine. The Java Virtual Machine – JVM – is customized for each different type of computer or operating system. This allows a company to ship the Pcode to any customer without giving away the Java source code that shows any secret details of how the program works.

# Advantages and Disadvantages

| | C or C++ | Java |
|---|---|---|
| Binary output & Distribution | Compiles to binary machine language<br>* Device drivers<br>* Closer to machine logic level | Compiles to Pcode<br>Compile once for all CPUs (JVM) |
| Speed | Very fast | Almost as fast now with JIT |
| Pointers | Pointers<br>* Direct access of all memory | No pointers<br>* Pointers can cause problems |
| Garbage collection | C/C++ can cause memory leaks<br>* Memory management is required<br>* Garbage collection can be added | No memory leaks<br>* Automatic memory cleanup |
| Amount of memory needed | Uses less memory. Does not need a virtual machine for execution | JVM is needed in addition to the compiled Java code |
| Object Oriented Program | C++ can use objects but not required | Java is totally object oriented |

There are several advantages of C or C++ when compared to Java.
Since C and C++ compile directly to the binary machine language, C and C++ can be used to write device drivers that directly interface to an operating system. Java can not be used in this manner because the JVM sits in between the Pcode and the operating system and its hardware.

Since C and C++ compile directly to machine code, they can execute at the full speed of the computer. Java Pcode is executed by the JVM and then in turn by the actual computer. Newer versions of the JVM have something called Just In Time or JIT which speeds things up by converting the Pcode into machine language when possible.

C and C++ use pointers which are great for accessing any part of memory, but this can cause problems for application programs that try to access memory that was not allocated to them by the operating system. Java uses something equivalent, called REFERENCES.

Java has something called 'garbage collection' that frees up any memory that the program is no longer using.

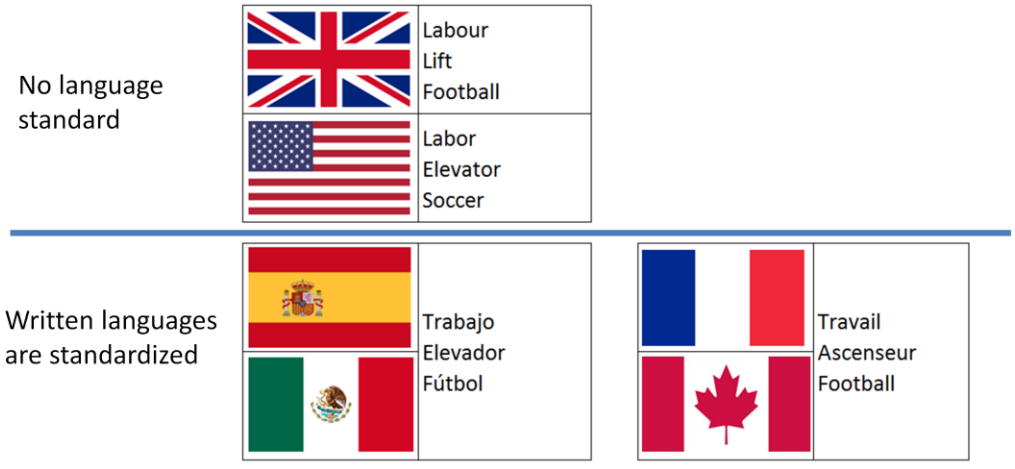
# Advantages and Disadvantages

| | C or C++ | Java |
|---|---|---|
| Binary output & Distribution | | piles to Pcode<br>pile once for all CPUs (JVM) |
| Speed | | ost as fast now with JIT |
| Pointers | | ointers<br>nters can cause problems |
| Garbage collecti | | emory leaks<br>tomatic memory cleanup |
| Amount of mem needed | | is needed in addition to the<br>piled Java code |
| Object Oriented Program | C++ can use objects but not required | Java is totally object oriented |


It is up to the programmer in C or C++ to free up any memory that it is no longer using. When it does not happen, it is referred to as a "Memory Leak"
If a C or C++ program keeps allocating memory and forgets to release all or some of it, the operating system will eventually run out of memory, causing not only the naughty program to fail, but it could bring down the entire system.

If you have ever seen the "Blue Screen of Death", this may be caused by a program that has "Memory Leaks"

# Written Language Differences




Although an American can go to Brittan, speak English and usually be understood, there are several differences between British and American English. Sometimes a different word is used for the same thing, other times the exact same word is spelled differently, such as LABOUR and LABOR, COLOUR and COLOR.

Noah Webster published his modified English dictionary in 1828, a few years after the war of 1812 in which the American burnt down the city of York, now Toronto in Canada, followed by the British burning down the Capitol and Whitehouse in Washington DC. Although both sides claimed a type of victory, neither side actually won the war and the British and Americans have been best of friends ever since. But the changed Noah Webster put in his dictionary have remained. Many other human languages such as Spanish and French have standardization committees that at least define how the languages are to be written.

C and C++ suffer a similar fate to English in that although there have been many attempts to standardize the language, many of the support subroutines and functions remain different between versions supplied by Microsoft, Apple, GNU Linux, etc.

The developers of Java did not want to suffer a fate similar to C and C++. All implementations of Java must be compatible. Even after Microsoft agreed to the requirements for Java, they started customizing it for their own needs. They were sued by Sun Microsystems and lost. Microsoft's hacked up version of Java is now

known as C# (C-sharp).

# Programming Language Popularity Contest

There is no true way to determine which language is the most popular in industry. The TIOBE index bases popularity on the number of web searches a language receives. The ratings change each year. Python continues to move up.

1. Java
2. C
3. C++
4. Python
5. Visual Basic .NET

See the top 50 at https://www.tiobe.com/tiobe-index/

There is no true way to determine which language is the most popular in industry. The TIOBE index bases popularity on the number of web searches a language receives. The ratings change each year. Python continues to move up.
Java
C
C++
Python
Visual Basic .NET
If you look at a different listing of popularity, you may see them show up in different orders.

## Which language should you learn and use?

1. Learn one of the top five languages and learn it well.
2. Learn a second language. It may be easier to learn a new language if it is closely related to one you know.
3. The language you will use for a project on your job will most likely be the one chosen for the project by the senior engineer. This is the one that you will use.

Which programming language should you learn and use?
Learn a second language. It may be easier to learn a new language if it is closely related to one you know.
The language you will use for a project on your job will most likely be the one chosen for the project by the senior engineer. This is the one that you will use.

## Which language should you learn and use?

- Java - “Write once, run anywhere” – Java has been the most sought after language for job searches. Java has very strict rules for data types.
- Python is easy to learn as your first language. It is very loosely typed (referring to data types).
- C is one of the more difficult languages to learn, but most modern programming languages are based on it.

* Java - “Write once, run anywhere” – Java has been the most sought after language for job searches. Java has very strict rules for data types.
* Python is easy to learn as your first language. It is very loosely typed (referring to data types).
* C is one of the more difficult languages to learn, but most modern programming languages are based on it.


In summary, Welcome to the Wonderful World of Programming. I look forward to see the results of all those keystrokes you pressed turning them into fantastic programs.