

Let's talk about computers:

A computer understands a very small number of commands. However, the instruction set is flexible enough that it can be used to perform an enormous number of tasks. The challenge is to put the commands together in the proper order.

A computer can execute commands extremely quickly.

If you store the instructions in memory, then the computer can fetch an instruction, execute it, then fetch another instruction extremely rapidly.

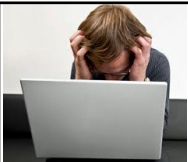
Sep 14-9:21 PM

Therefore, even a very long list of simple instructions can be executed quickly.

SO... what we need to do is

1. Understand the instruction set
2. Break down a complex task into a series of simpler tasks
3. Express the simple tasks as a sequence of instructions from the instruction set.

Sep 14-9:42 PM



Computer Programmers write the instructions.

BIG BIG BIG problem: Each computer has its OWN UNIQUE instruction set!

Solution:

Invent an "intermediate language" which is flexible enough to perform well AND easy enough to understand THEN write a "translator" (interpreter or compiler) which converts the intermediate language into the computer's unique instruction set.

Sep 14-9:49 PM

You can have a different interpreter/compiler for each computer so that the programmer can write in the "intermediate" language and let the computer actually translate it into machine code.

The "intermediate language" is what we call a programming language.



Sep 14-9:59 PM

Programming languages frequently look like a combination of English and algebra (with some extra odd stuff thrown in.)

BASIC
 COBOL
 FORTRAN
 C
 C++
 PASCAL
 JAVA



Sep 14-9:59 PM

'What's a Microcontroller - first program.bs2
'Basic Stamp sends message to Debug Terminal

```
{ $Stamp BS2 }
{ $PBASIC 2.5 }
```

```
DEBUG "Hello, it's me, your BASIC Stamp! "
```

```
END
```

Sep 14-9:11 PM

Formatters and Control Characters

```
DEBUG CR, "What's 7 x 11?"  
DEBUG CR, "The answer is : "  
DEBUG DEC 7*11
```

Sep 14-9:17 PM