Language Survey

Name	Capability	Difficulty	Flexibility	Speed
Assembly language	almost none	Extreme	complete	Ludicrous
C/C++/ Fortran	Very little	medium	Almost complete	great
Java / Javascript/ Perl / python / php	better	medium	Almost complete	good
Cobol	limited	medium	Very little	good

Not quite programming languages, but close:

- html, ruby → web development
- flash → animations / games
- xml → document format
- jcl ('job control language') → used in mainframe computers

There are (probably) more than 100 programming languages. Some of these are real and used daily, some are theoretical.

Excel - more accurate to call this a computing tool, but you can write your own programs that run in Excel using Visual Basic

Matlab - this is both a programming tool and a language.

Precision

- We are interested in traditional floating point computing (as opposed to symbolic manipulators like Maple / Mathematica).
- In a floating point environment, all numbers are represented in decimal form

$$1/3 = 0.33\overline{3}$$

we can't store $1/3$ exactly. At some point, the decimal approximation must be truncated.

- until very recently, computations were performed in <u>single precision</u> most of the time

Single precision → 7 or 8 decimal digits

This is because computer memory was either large/bulky and/or very expensive.

- Now, the default precision in computations is <u>double precision</u>

double precision - 15 or 16 decimal digits

This is possible because memory has gotten much smaller/cheaper.

It is also necessary because of the number of calculations required to solve complex problems.

$x = 1.2345678901234\,56$
$y = 0.0123456789012\,3456$ } double precision

The result of $x+y$ will not be exact (requires 18 digits and we can only store 16).

Each operation has the potential to increase the error by 1 digit in the last decimal place.

1 operation ⇒ 1 digit in last place
2 ——— ⇒ 2 digits ———
3 ——— ⇒ 3 digits ———
10 operations ⇒ 10 digits in last place
= 1 digit in next to last place
100 operations = 1 digit in 3rd to last place

A complex solution process can easily require $10^{12}, 10^{13}$ calculations

$x = 1.2345678901234 5$

↑ Error can creep up into these digits in the worst possible case

We use lots of digits because in a statistical sense, the error will usually stay in the lower 7-8 digits.