Algoritma & Pemrograman Saintifik

Gatot F. Hertono, Ph.D

Departemen Matematika

SCMA601401

What is Algorithm?

How do you make this?

… or this?

Or fixing this problem?

What is Algorithm?

Algorithm

- is any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output.
- is thus a sequence of computational steps that transform the input into the output.
- is a tool for solving a well - specified computational problem.
- Any special method of solving a certain kind of problem (Webster Dictionary)

Sources: Design and Analysis of Computer Algorithm, Pradondet Nilagupta

Exercise: Designing an Algorithm

Design an algorithm to:

Make this

Or

Fix this

What is a program?

- A program is the expression of an algorithm in a programming language
- a set of instructions which the computer will follow to solve a problem

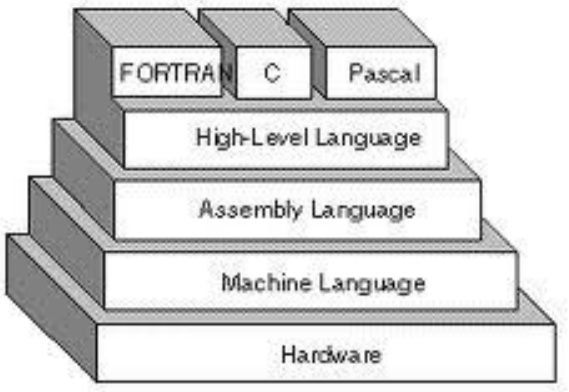
Sources: Design and Analysis of Computer Algorithm, Pradondet Nilagupta

Algorithm and Program

Write a program to compute: $\sum_{i=1}^{n} i = 1 + 2 + 3 + \cdots + n$

Algorithm	Program (in Matlab syntax)
1. Masukkan bilangan integer *n;* 2. Tentukan *Sum* = 0; 3. Mulai dari *i*=1 s/d *n* lakukan *Sum* = *Sum* + *i* ; 4. Cetak hasil *Sum*	n = input('masukkan suatu bilangan integer: ') Sum = 0; for i=1:n Sum = Sum + i; end; sprintf('Hasil: %d', Sum)

Programming Language

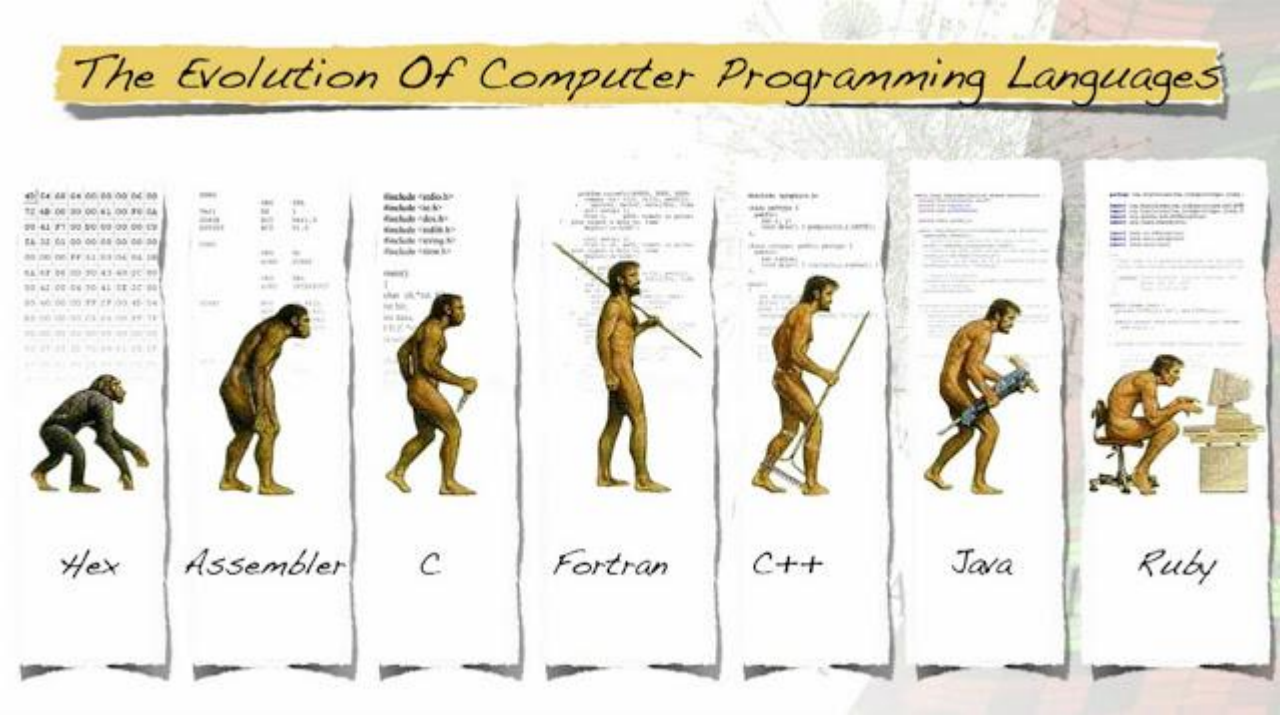


A Programming language

- is a vocabulary and set of grammatical rules for instructing a computer to perform specific tasks.
- The term *programming language* usually refers to high-level languages, such as BASIC, C, C++, COBOL, FORTRAN, Ada, and Pascal. Each language has a unique set of keywords (words that it understands) and a special syntax for organizing program instructions.
- High-level programming languages, while simple compared to human languages, are more complex than the languages the computer actually understands, called *machine languages*. Each different type of CPU has its own unique machine language.
- Lying between machine languages and high-level languages are languages called assembly languages. Assembly languages are similar to machine languages, but they are much easier to program in because they allow a programmer to substitute names for numbers. Machine languages consist of numbers only.
- Lying above high-level languages are languages called *fourth-generation languages* (usually abbreviated *4GL*). 4GLs are far removed from machine languages and represent the class of computer languages closest to human languages.
- Regardless of what language you use, you eventually need to convert your program into machine language so that the computer can understand it. There are two ways to do this:
  - compile the program, or
  - *interpret* the program
- The question of which language is best is one that consumes a lot of time and energy among computer professionals. Every language has its strengths and weaknesses. For example,
  - FORTRAN is a particularly good language for processing numerical data, but it does not lend itself very well to organizing large programs.
  - Pascal is very good for writing well-structured and readable programs, but it is not as flexible as the C programming language.
  - C++ embodies powerful object-oriented features, but it is complex and difficult to learn.
- The choice of which language to use depends on the type of computer the program is to run on, what sort of program it is, and the expertise of the programmer.

Sources: http://www.webopedia.com/TERM/P/programming_language.html

Programming Language



Notes:

- This course will not teach a programming language.
- You are free to choose any programming language to use.
- We use **MATLAB** as a tool to learn programming.

Algorithms = Logic + Control

An algorithm can be regarded as consisting of

- **a logic component**, which specifies the knowledge to be used in solving problems, and
- **a control component**, which determines the problem-solving strategies by means of which that knowledge is used.

The logic component determines the meaning of the algorithm whereas the control component only affects its efficiency.

The efficiency of an algorithm can often be improved by improving the control component without changing the logic of the algorithm.

We argue that computer programs would be more often correct and more easily improved and modified if their logic and control aspects were identified and separated in the program text.

Sources:Algorithm = Logic + Control, Robert Kowalski

Examples

How do you compute:

n !

and

The n^{-th} Fibonacci number