

# Temu 02

## Pemrograman & Paradigma OOP

**Ali Tarmuji, S.T., M.Cs.**

[alitarmuji@tif.uad.ac.id](mailto:alitarmuji@tif.uad.ac.id)

# Bahan Diskusi

---

- Sejarah bahasa pemrograman
- Bahasa perograman berbasis objek
- Paradigma “Berorientasi Objek”
- Abstraksi
- Contoh aplikasi berbasis objek

---

# SEJARAH BAHASA PEMROGRAMAN

# Mengapa bahasa pemrograman?

---

- ❑ Bahasa merupakan sarana untuk berkomunikasi
- ❑ Untuk 'berkomunikasi' dengan komputer kita perlu menguasai 'bahasa komputer'
- ❑ Bahasa yang dimengerti komputer adalah bahasa pemrograman
- ❑ Memprogram adalah proses berkomunikasi dengan komputer
- ❑ Tujuan memprogram computer: memerintahkan komputer untuk melakukan tugas-tugas komputasi dan input-output sesuai yang diinginkan pemrogram



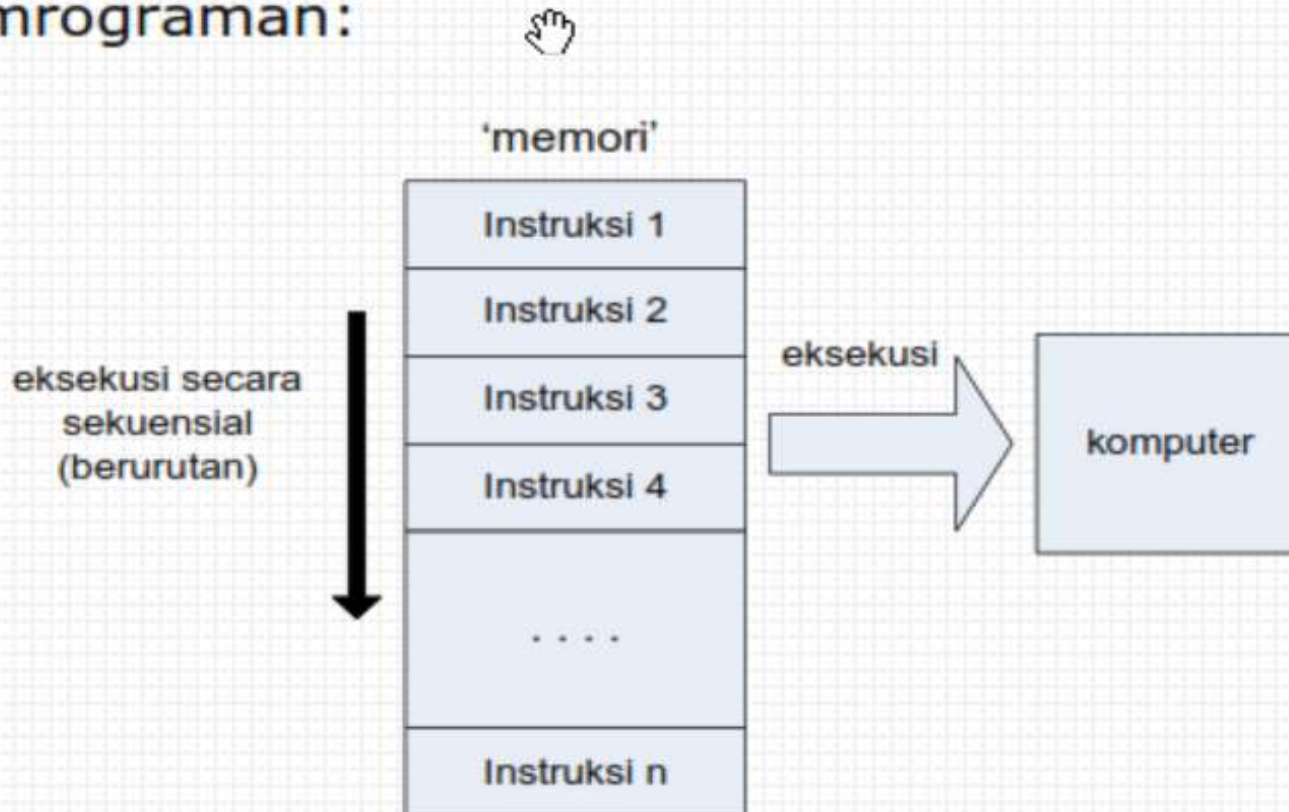
# Komputer & Pemrograman

---

- Komputer merupakan salah satu produk teknologi tinggi yang dapat melakukan hampir semua pekerjaan diberbagai disiplin ilmu, tetapi komputer hanya akan merupakan barang mati tanpa adanya bahasa pemrograman untuk menggambarkan apa yang kita kerjakan, sistem bilangan untuk mendukung komputasi, dan matematika untuk menggambarkan prosedur komputasi yang kita kerjakan

# Komputer & Pemrograman

- ❑ Bagaimana komputer menjalankan instruksi-instruksi pemrograman??
- ❑ Gambaran sederhana komputer dan instruksi pemrograman:





# Definisi Bahasa Pemrograman

- ❑ **Bahasa (*language*)** adalah suatu sistim untuk berkomunikasi. Bahasa tertulis menggunakan simbol (yaitu huruf) untuk membentuk kata
- ❑ **Bahasa Pemrograman (*programming language*):**
  - Notasi yang dipergunakan untuk mendeskripsikan proses komputasi dalam format yang dapat dibaca oleh komputer dan manusia
  - Kumpulan perintah-perintah bermakna, berstruktur tertentu (syntax) yang dapat dimengerti komputer yang berguna didalam penyelesaian masalah
- ❑ **Bahasa Natural** dirancang untuk memfasilitasi komunikasi antar manusia
- ❑ **Bahasa Pemrograman** dirancang untuk memfasilitasi komunikasi antara manusia dengan komputer



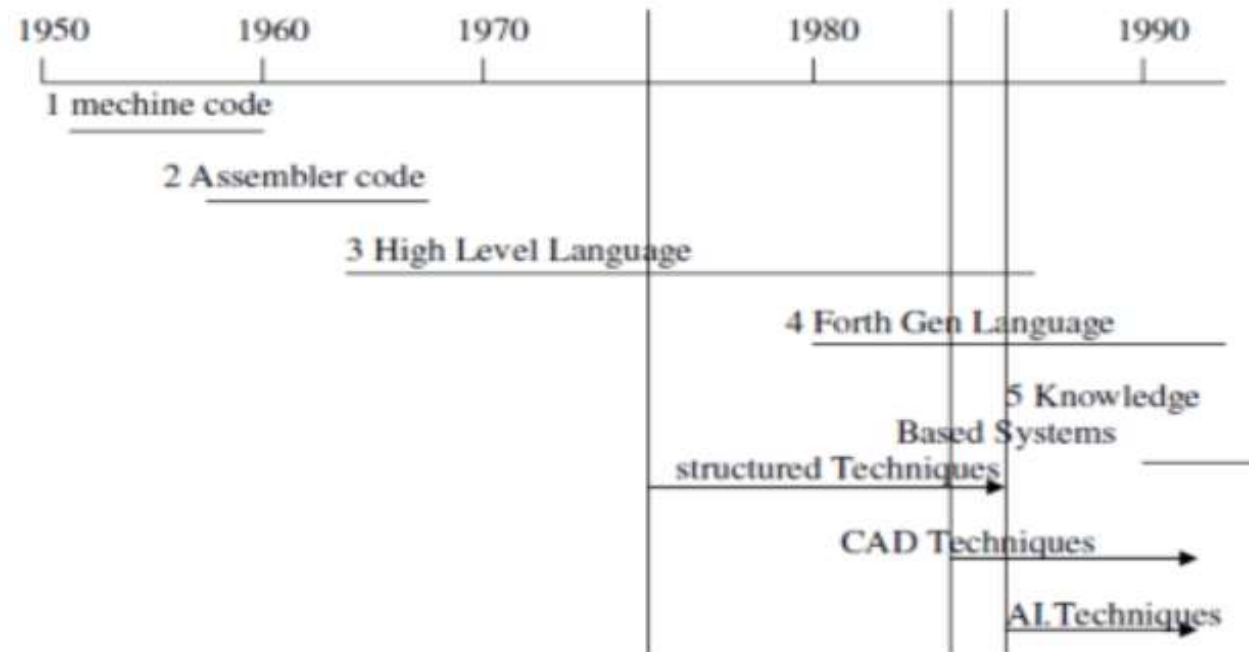
# Definisi Bahasa Pemrograman

- ❑ Dalam pengertian luas pemrograman meliputi seluruh kegiatan yang tercakup dalam:
  - Pembuatan program, termasuk analisis kebutuhan (requirement's analysis)
  - Keseluruhan tahapan dalam perencanaan (planning), perancangan (design) dan pewujudannya (implementation).
  
- ❑ Dalam pengertian yang lebih sempit, pemrograman merupakan:
  - Pengkodean (coding atau program writing = penulisan program)
  - Pengujiannya (testing) berdasarkan rancangan tertentu.

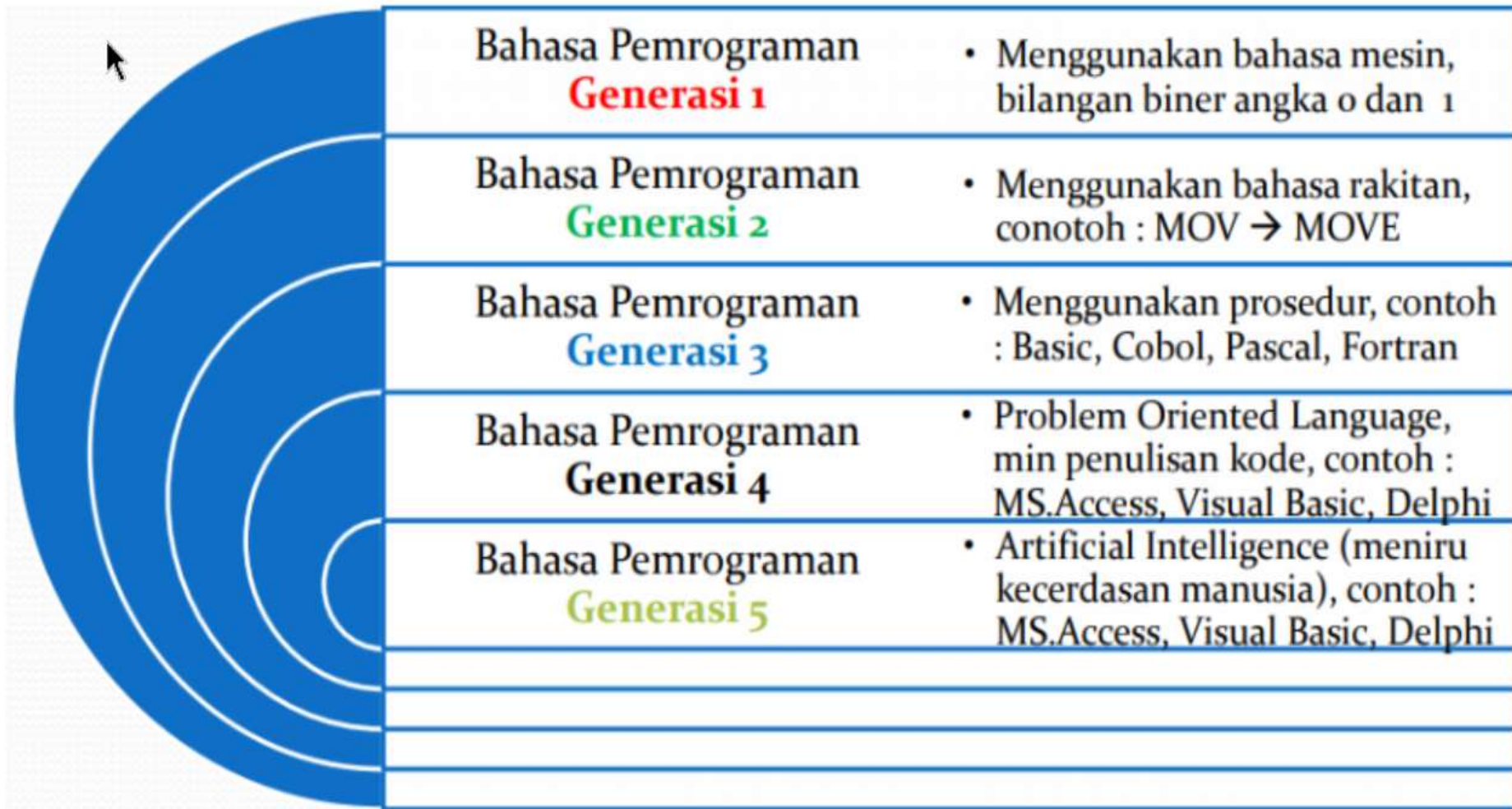


# Lima Generasi Bahasa Pemrograman

1. Bahasa Mesin: Generasi pertama
2. Bahasa Assembly: Generasi kedua
3. Bahasa Tingkat-tinggi : Generasi ketiga
4. Bahasa Generasi Keempat
5. Bahasa Generasi Kelima



# Generasi Bahasa Programming





# Lima Generasi Bahasa Pemrograman

---

- ❑ Ukuran 'kecanggihan' suatu bahasa pemrograman tersebut diukur dari kemudahan programmer menyusun suatu program
- ❑ Lebih 'tidak canggih' bukan berarti tidak 'powerful' → kadang-kadang dalam membuat suatu program dibutuhkan penggunaan bahasa tingkat 1 atau 2 atau kombinasi antara bahasa tingkat tinggi dengan bahasa assembly.  
Tergantung tujuan programnya dan komputer jenis apa yang diprogram.



# Machine Language

---

- ❑ Bahasa mesin adalah bahasa internal komputer yang mengeksekusi secara langsung tanpa terjemahan (translation)
- ❑ Disebut generasi pertama karena merupakan jenis yang paling awal dikembangkan: tahun 1940-an dan awal 1950-an semua program harus dikodekan dalam bahasa mesin
- ❑ Pemrograman dalam bahasa mesin:
  - menyita waktu dan kondusif untuk membuat kesalahan
  - berbeda untuk setiap jenis komputer, sehingga bergantung pada komputer dan tidak standar
- ❑ Semua program harus ada dalam bahasa mesin agar dapat dieksekusi, sehingga bahasa lain yang ditulis programmer perlu diterjemahkan oleh komputer ke bahasa mesin untuk eksekusi

# Machine Language

- ❑ Merupakan kode-kode bilangan biner, terdiri dari kombinasi bilangan '1' dan '0'
- ❑ Setiap kombinasi mewakili satu instruksi
- ❑ Instruksi bahasa mesin tergantung dari jenis komputer yang dipakai(machine dependent)
- ❑ contoh: instruksi transfer data dari akumulator ke register:
  - Komputer intel 8051 <sup>1)</sup> : '01000111B' (47H)
  - Komputer Z80 <sup>2)</sup> : '00010011B' (13H)
- ❑ Pemrograman bahasa mesin: instruksi-instruksi langsung dimasukkan ke memori untuk dieksekusi komputer

## **Note:**

- 1) Intel 8051 adalah komputer generasi lama yang sekarang digunakan sebagai mikrokontroler
- 2) Z80 adalah komputer generasi lama produksi Zilog



# Assembly Language

- ❑ Merupakan penyempurnaan bhs generasi pertama, sudah memasukkan unsur kata bahasa Inggris dalam bentuk singkat
- ❑ Masih bersifat machine dependent
- ❑ Penulisan sudah jauh lebih mudah dari bahasa mesin, tetapi programmer tetap harus memahami perangkat keras komputer
- ❑ Beberapa variabel masih mengacu pada register, alamat memori maupun I/O
- ❑ Contoh: instruksi transfer data dlm bhs assembly:
  - Komputer 8051 : MOV R1,#22H
  - Komputer AVR <sup>3)</sup> : LDI R1,0x22



## **Note:**

3) AVR digunakan sebagai mikrokontroler produksi Atmel




# Assembly Language

---

- ❑ Bahasa assembly sangat menyerupai bahasa mesin, sehingga untuk menjadi programmer bahasa assembly yang cakap kita harus memahami arsitektur mesin, yakni bagaimana mesin itu secara fisik memproses data
- ❑ Sama seperti bahasa mesin, bahasa assembly tergantung komputer (tidak portable)
- ❑ Untuk menerjemahkan kode-kode diperlukan program khusus yang disebut **ASSEMBLER**
- ❑ Bahasa assembly masih digunakan karena begitu mirip dengan bahasa mesin dengan kode yang sangat efisien
- ❑ Untuk membuat system software lebih disukai menggunakan bahasa assembly karena sangat efisien dalam penggunaan komputer (butuh memori yang kecil)

# High-level Language

- ❑ Memasukkan lebih banyak unsur kata bahasa Inggris yang digunakan sehari-hari dan mempunyai sintaksis yang lebih baik
- ❑ Merupakan bahasa pemrograman yang digunakan sekarang pada umumnya untuk memprogram komputer
- ❑ Contoh macam-macam bahasa tingkat tinggi: dsb
- ❑ Bahasa mesin dan assembly terlalu sulit, sehingga muncul third-generation languages (3GLs) yang lebih mudah untuk program dan portable
- ❑ "High-level" → mudah dipelajari & program tingkat-tinggi memerlukan proses penerjemahan oleh komputer yang sangat rumit
- ❑ Memiliki kemampuan untuk merepresentasikan **algoritma yang kompleks** 
- ❑ *Human-oriented readability; Machine-independent*
- ❑ Program penerjemahnya disebut **COMPILER** atau **INTERPRETER**
- ❑ Contoh: FORTRAN (FORMula TRANslator), Cobol, **Pascal**, BASIC, C, C++, dan Object-oriented programming language
- ❑ Bahasa C disebut bahasa "tingkat-menengah" karena format instruksinya dengan bahasa tingkat-tinggi sekaligus bisa berinteraksi langsung dengan hardware



# Fourth-Generation Language

- ❑ Mudah untuk dipelajari dan dipahami
- ❑ Tepat untuk pengaksesan database
- ❑ Memfokuskan pada memaksimalkan produktivitas manusia dari pada minimisasi waktu komputer
- ❑ Nonprosedural
- ❑ Tersedia dalam software paket yang dapat digunakan untuk mengembangkan aplikasi yang diinginkan
- ❑ Contoh:
  - Query language seperti SQL (structured query language), QBE (query-by-example) dan INTELLECT
  - Report generator
  - Application generator seperti MANTIS dan ADS
  - Matlab



# Fifth-Generation Language

- ❑ Sering digunakan untuk akses database atau membuat sistem pakar (*expert system*) atau *knowledge-based system*
- ❑ Dalam konsep, ditunjukan untuk bahasa alami (*natural languages*) yang semirip mungkin dengan hubungan kemanusiaan
- ❑ Contoh: LISP dan Prolog
- ❑ Sekarang ini banyak sistem pakar dikodekan baik dalam LISP maupun Prolog, meski untuk hal yang sama bisa ditulis dalam C atau C++.
- ❑ Usaha yang sekarang dilakukan adalah memperbaiki bahasa AI (*artificial intelligence*) dengan mengkombinasikan kemampuan terbaik dari LISP dan Prolog

# Penerjemah Bahasa Pemrograman

## □ **Interpreter**

- Menganalisis dan mengeksekusi setiap baris dari program tanpa melihat program secara keseluruhan
- Sebuah program yang *dapat mengerti* sebuah bahasa dan *mengeksekusi program* yang ditulis dengan bahasa tersebut
- Contoh: Basica, Foxpro, Matlab

## □ **Compiler**



- Merupakan suatu program yang menterjemahkan bahasa program (source code) ke dalam bahasa objek (object code) secara keseluruhan program
- Program yang *menterjemahkan* program yang ditulis dengan sebuah bahasa *menjadi program* yang ditulis oleh bahasa lain
- Contoh: Turbo Basic, Pascal, C/C++, dll



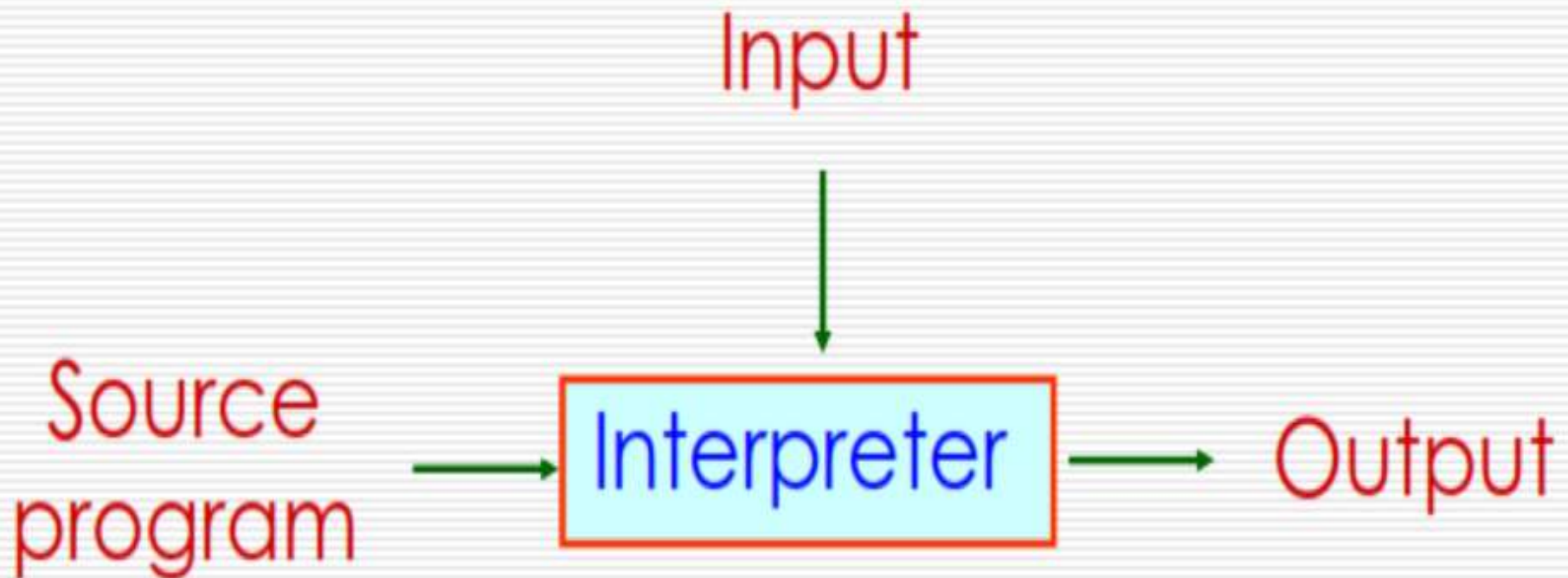
# Penerjemah Bahasa Pemrograman

Interpreter	Compiler
Menterjemahkan intruksi per intruksi	Menterjemahkan secara keseluruhan
Bila terjadi kesalahan interpretasi dapat langsung diperbaiki	Bila terjadi kesalahan kompilasi maka source program harus diperbaiki dan di kompilasi ulang
Tidak dihasilkan object program	Dihasilkan object program
Tidak dihasilkan executable program	Dihasilkan executable program
Proses pengerjaan lebih lambat	Proses pengerjaan program lebih cepat
Seource program terus dipergunakan	Source program tidak di pergunakan hanya bila untuk perbaikan saja
Keamanan dari program kurang terjamin	Keamanan dari program lebih terjamin



# Interpreter

---



# Compiler

---

