Pertemuan Pertama (Sabtu, 12 Februari 2022)

1. **Introduction**
2. Alur pembelajaran di kelas

Di kelas ini kita akan belajar dan berdiskusi mengenai pemrograman dengan menggunakan bahasa Python

Pembahasan Meliputi :

* Operasi dasar pada Python.
* Struktur data dan penggunaannya.
* Iterasi dan rekursi.
* Mengorganisir dan memodularisasi program.
* *Best practice* menggunakan Python.
* Efisiensi program.

1. Pengertian dan kegunaan python

[Python](https://www.python.org/) adalah bahasa pemrograman yang dikembangkan oleh Guido van Rossum pada tahun 1990.

Python:

* Menekankan kepada keterbacaan program.
* Dibandingkan dengan bahasa pemrograman terdahulu seperti C, C++, Java, dll, program pada Python lebih mudah untuk dibaca.
* Python merupakan program Open Source, yang berarti pengguna dapat secara gratis menggunakan dan bebas untuk memodifikasinya.

1. Motivasi menggunakan Python

Belakangan ini juga, penggunaan Python semakin banyak di dalam komunitas Data Science dan Machine Learning. Untuk keperluan pengolahan data untuk kebutuhan Data Science dan juga Machine Learning, Python memiliki ketersediaan library dan framework yang lengkap seperti:

* [Scikit-learn](https://scikit-learn.org/)
* [PyTorch](https://pytorch.org/)
* [TensorFlow](https://www.tensorflow.org/)
* [Pandas](https://pandas.pydata.org/)
* [Numpy & Scipy](https://www.scipy.org/)

1. Komputer dan kemampuannya

Komputer hanya memiliki kemampuan mengingat dan menghitung tanpa memiliki kreativitas akan ide baru.

Dalam kegiatan sehari-hari yang menggunakan komputer:

* kita melihat komputer dapat melakukan banyak aktivitas yang kompleks seperti mengedit foto
* membuat dokumen
* *browsing* internet
* maupun memainkan *game*.

Operasi perhitungan yang dijalankan computer adalah : (primitive)

* Operasi aritmetik sederhana.
* Komparasi.
* Pengulangan.

1. **Knowledge**
2. Pengetahuan Komputer

Tidak seperti manusia, komputer tidak memiliki *knowledge*:

* Komputer hanyalah mesin untuk menyimpan dan menghitung.
* Komputer menjalankan perintah yang dimasukkan oleh *user*.
* "*Knowledge*" yang dimiliki komputer sebatas informasi dan aturan yang disimpan di dalam memori.
* Sebagai *user*, kita memasukkan *knowledge* kita ke komputer.

Secara garis besar, jenis pengetahuan dapat dibagi menjadi 2:

* *Declarative Knowledge*

*Declarative Knowledge* merupakan tipe pengetahuan yang berupa **pernyataan suatu fakta**, sehingga dalam tipe ini tidak ditunjukkan langkah-langkah untuk mencapai suatu fakta tersebut. Contoh dari *Declarative Knowledge*:

"Baju saya berada di dalam lemari."

"Kue itu terbuat dari tepung dan telur."

"ATM terdekat berada di SPBU Kuningan."

* *Imperative Knowledge*

Imperative Knowledge merupakan tipe pengetahuan yang menunjukkan langkah-langkah untuk mencapai suatu tujuan. Jika diibaratkan, *Imperative Knowledge* adalah resep masakan.

Contoh:

1. Beli bahan-bahan yang dibutuhkan.
2. Campur tepung, telur, dan susu ke dalam satu wadah.
3. Aduk hingga semua bahan tercampur rata.
4. Tuang adonan kedalam cetakan.
5. Masak hingga matang.
6. Penerapan Knowledge Dalam Pemrograman

#### **Declarative Knowledge**

* Nilai pangkat dua suatu bilangan x adalah y sehingga x∗x=y
* Nilai pangkat tiga suatu bilangan x adalah y sehingga x∗x∗x=y
* etc

#### **Imperative Knowledge**

Untuk mencari hasil perpangkatan dari suatu bilangan diperlukan langkah-langkah berikut:

* Masukkan suatu bilangan x dan nilai dari pangkatnya z
* Jika nilai z adalah 0, maka nilai dari perpangkatan adalah 1.
* Jika nilai z adalah 1, maka nilai dari perpangkatan adalah x.
* Jika nilai z adalah bilangan bulat positif mulai dari 2, maka kalikan x dengan dirinya sendiri sebanyak z-1 kali.
* etc

Untuk keperluan programming dipakai tipe imperative knowledge sebagai cara menyampaikan instruksi yang kita inginkan ke komputer. Declarative knowledge tidak memiliki arti banyak bagi komputer.

### **Kesimpulan**

Untuk menyuruh komputer melakukan tugas yang kita inginkan, maka digunakan pendekatan berupa imperative knowledge yang berupa "resep" untuk melakukan sesuatu. Secara sederhana "resep" dapat dijabarkan menjadi 3 poin, yaitu:

1. Urutan dari **langkah-langkah sederhana**.
2. **Urutan dari proses kontrol** yang menspesifikasi kapan tiap langkah dieksekusi.
3. Memiliki informasi **kapan untuk berhenti** dan **kapan untuk jalan**.
4. **Komputer adalah mesin**
5. **Jenis/tipe computer**

 Secara historis terdapat dua tipe komputer yaitu:

* Fixed Program
* Stored Program

Komputer *fixed program* adalah komputer yang sangat spesifik untuk suatu tugas tertentu. Contoh paling sederhana dari jenis ini adalah **kalkulator**. Sedangkan komputer stored program adalah komputer pada umumnya yang kita kenal (PC).

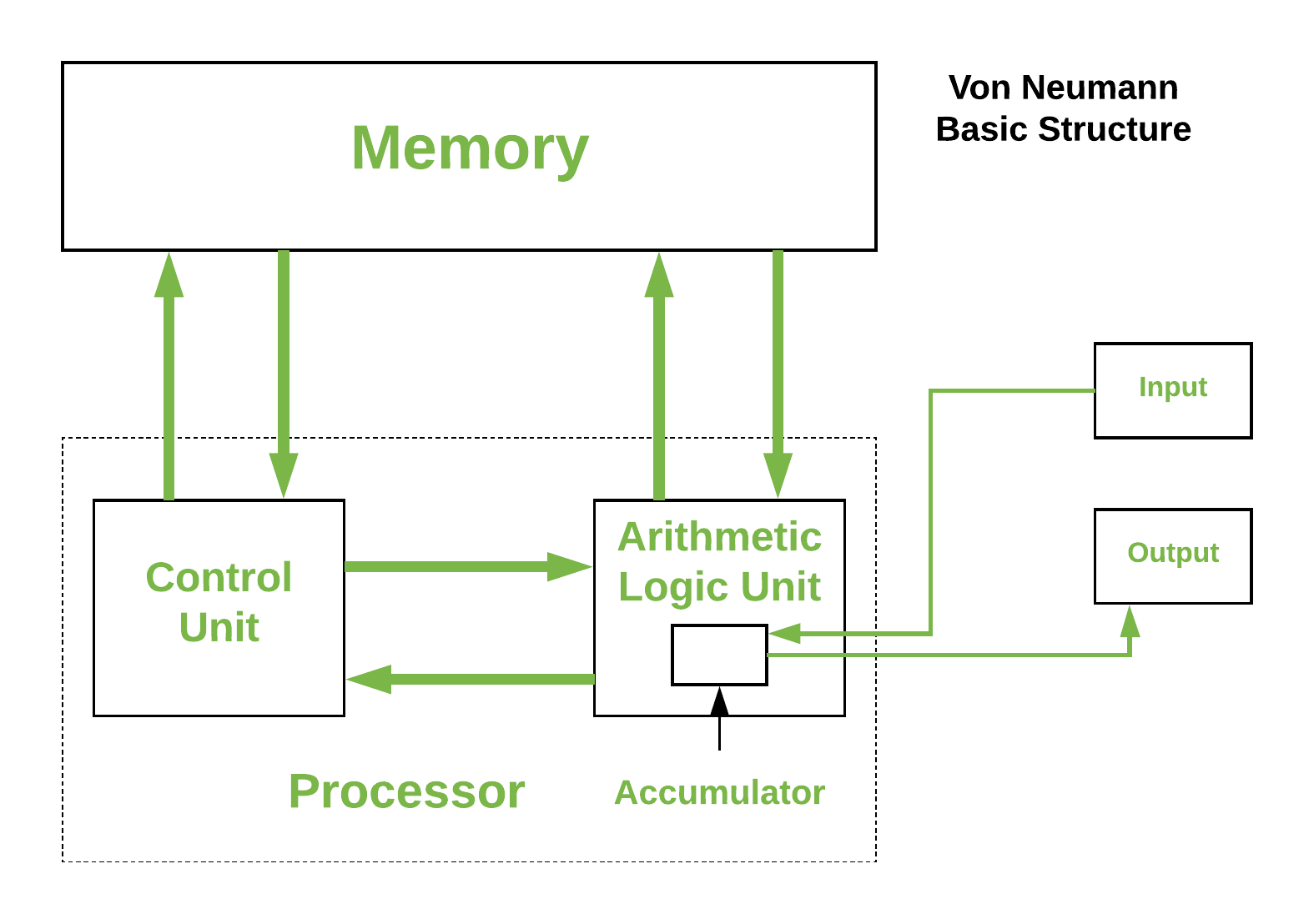
Perbedaan *stored* dan *fixed program*:

* Stored program dapat melakukan kombinasi perintah yang lebih kompleks.
* Kombinasi perintah dapat disimpan sebagai program.
* Program yang disimpan dapat diaktifkan maupun dinonaktifkan.
* Dapat berganti-ganti program sesuai kebutuhan.

1. **Komponen dasar pada computer**

Di dalam komputer *stored program*, komputer yang kita kenal pada umunya, terdapat beberapa komponen inti yang terdiri dari 3 komponen besar yaitu:

* Interface (Untuk berinteraksi dengan *user*)
* Prosesor (Untuk mengolah informasi)
* Memory (Untuk menyimpan informasi)



* Secara sederhana, di dalam memori disimpan dua jenis informasi:
  + Data (informasi yang ingin kita olah) seperti kumpulan dari angka, kata, gambar, dll.
  + Algoritma (urutan dari instruksi yang ingin dijalankan).
* Untuk mengolah data tersebut digunakan prosesor.
* Untuk berinteraksi dengan user digunakan *interface* berupa, input dan output.

**CONTOH**

Menghitung rata-rata dengan Microsoft excel:

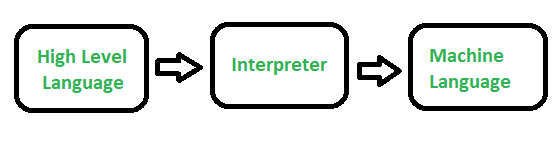
* *Interface* input adalah keyboard dan mouse.
* *Interface* output adalah layar monitor.
* Angka yang anda masukkan ke Excel, disebut dengan **Data**.
* Perintah rata-rata di dalam Excel adalah **Algoritma**.
* Untuk menghitung rata-rata, prosesor akan mengambil **data** dan **algoritma** sesuai perintah **input** di dalam **memori**.
* Prosesor melihat urutan perintah di dalam **algoritma** dan mengolah **data** sesuai algoritma tadi.
* Setelah selesai diproses, **data** bisa disimpan kedalam **memori**.
* Atau ditampilkan sebagai **output** di monitor.

### **Bahasa Pemrograman**

* Seperti disebutkan di contoh, prosesor melihat urutan perintah dalam **algoritma** untuk melakukan sesuatu.
* Namun, **algoritma** tersebut ditulis di dalam bahasa pemrograman, yang mudah ditulis dan dibaca oleh manusia.
* Bagaimana cara prosesor dapat mengetahui perintah yang kita tulis?

Dengan bahasa pemrograman!

* Gampangnya, bahasa pemrograman(interpreter) adalah suatu program di dalam komputer yang bertugas menerjemahkan instruksi dari user ke mesin.



1. **Operasi Dasar**

### **Operasi Dasar**

Lalu apa itu operasi dasar?

* Kumpulan dari operasi-operasi yang dimiliki oleh suatu bahasa pemrograman.
* Dapat digunakan untuk menjadi dasar sesuatu yang lebih kompleks

Illustrasi:

* Anggap program yang ingin dibuat sebagai rumah.
* Operasi dasar = Bahan dasar rumah (Batu bata, semen, baja, kayu, batu, seng, dll).
* Dari bahan-bahan tersebut dapat membuat: Pondasi, tiang, tembok, pintu, atap
* Satukan bahan-bahan tersebut, menjadi: ruang tidur, ruang makan, kamar mandi, dll.
* Satukan ruang-ruang tersebut, jadi rumah.

1. **Bahasa Pemrograman**

Bagaimana cara agar algoritma kita dimengerti komputer?

* Digunakan bahasa pemrograman sebagai perantara *user* dan komputer.
* Perintah yang ditulis dalam bahasa pemrograman akan diterjemahkan oleh interpreter menjadi bahasa mesin.

Bahasa pemrograman memiliki elemen dan operasi dasar yang menjadi *building block* program yang akan dibuat:

* Kombinasi dari operasi dan elemen dasar disebut dengan *expressions*.
* Definisi formal *expression* : Kombinasi kompleks yang legal dari operasi dan elemen dasar di dalam bahasa pemrograman.

Untuk membentuk suatu *expression* yang legal, maka perlu diperhatikan aspek-aspek berikut:

### **Operasi dan elemen dasar (Basic Primitives)**

Basic primitives merupakan elemen-elemen dasar suatu bahasa.

Analogi basic primitives:

* Bahasa Indonesia : kata, angka dan tanda baca.
* Bahasa pemrograman : angka (integer,float), teks(string), dan operasi-operasi lainnya (+-\*/\*\*//).

### **Syntax**

Secara sederhana, sintaks adalah kombinasi yang valid dari elemen-elemen dasar suatu bahasa. Contoh dalam bahasa Indonesia:

* "Budi membeli nasi goreng di pasar." adalah contoh kalimat yang valid secara sintaks.
* "Budi nasi goreng di pasar membeli." adalah contoh kalimat yang tidak valid secara sintaks.

### **Semantik Statik**

Istilah statik semantik disini merujuk kepada apakah suatu expression atau kalimat memiliki arti. Contoh paling mudah dalam bahasa inggris:

* "I am tired." adalah kalimat yang valid secara sintaks dan valid secara statik semantik.
* "You is tired." adalah kalimat yang valid secara sintaks (pronoun + verb + adj), namun memiliki error pada statik semantik.

Contoh dalam bahasa pemrograman:

valid secara statik semantik

Error statik semantik

* string tidak bisa digabung dengan integer

### **Semantik**

Dalam bahasa manusia, seperti bahasa Indonesia maupun inggris, walaupun secara sintaks dan semantik statik sudah benar, namun masih terdapat kemungkinan untuk suatu kalimat bermakna ganda atau ambigu. Sebagai contoh:

* "Kakakku adalah seorang penjahit jas wanita." Kalimat ambigu karena dapat bermakna bahwa "Kakakku adalah seorang penjahit jas khusus wanita." atau "Kakakku adalah seorang wanita penjahit jas."
* "Pamanku adalah seorang manager perusahaan terkenal." dapat bermakna "Pamanku yang terkenal, adalah seorang manager perusahaan" atau "Pamanku adalah seorang manager dari perusahaan terkenal"
* Untungnya, dalam bahasa pemrograman, suatu expression yang secara sintaks dan statik semantik benar, hanya memiliki satu arti.
* Namun, biasanya kesalahan programmer adalah dalam menerjemahkan apa yang dia pikirkan menjadi suatu program.
  1. **Error dalam pemrograman**

### **Beberapa kemungkinan Kesalahan pada program sederhana.**

* Error dalam sintaks
  + Sering terjadi dan mudah untuk diketahui
* Error statik semantik
  + Beberapa bahasa pemrograman memeriksa hal ini sebelum menjalankan perintah.
  + Dapat menyebabkan error yang tidak terprediksi.
* Tidak ada error semantik, namun berbeda dengan yang kita maksud.
  + Program error.
  + Program tidak bisa berhenti.
  + Program memberikan hasil, tetapi tidak sesuai harapan.

Rangkuman

* Pada dasarnya, yang dilakukan oleh komputer adalah mengingat dan melakukan kalkulasi.
* Kalkulasi yang dilakukan oleh komputer dikerjakan berdasarkan perintah yang disimpan didalam komputer.
* Untuk menyampaikan suatu perintah ke komputer digunakan algoritma, yang pada dasarnya adalah "resep" untuk melakukan sesuatu.
* Untuk menyampaikan algortima tersebut kedalam komputer, dibutuhkan perantara yang berupa bahasa pemrograman.
* Bahasa pemrograman terdiri dari beberapa aspek yaitu:
  + Elemen dasar (*Basic Primitives*)
  + Sintaks
  + Statik Semantik
  + Semantik
* Agar algoritma yang kita rancang tersampaikan dengan baik ke komputer, maka seorang programmer harus memperhatikan aspek-aspek di dalam bahasa pemrograman tersebut.