

# *Algorithmic Thinking*

Tim Penyusun Materi KU1102-Pengenalan Komputasi  
Institut Teknologi Bandung © 2019

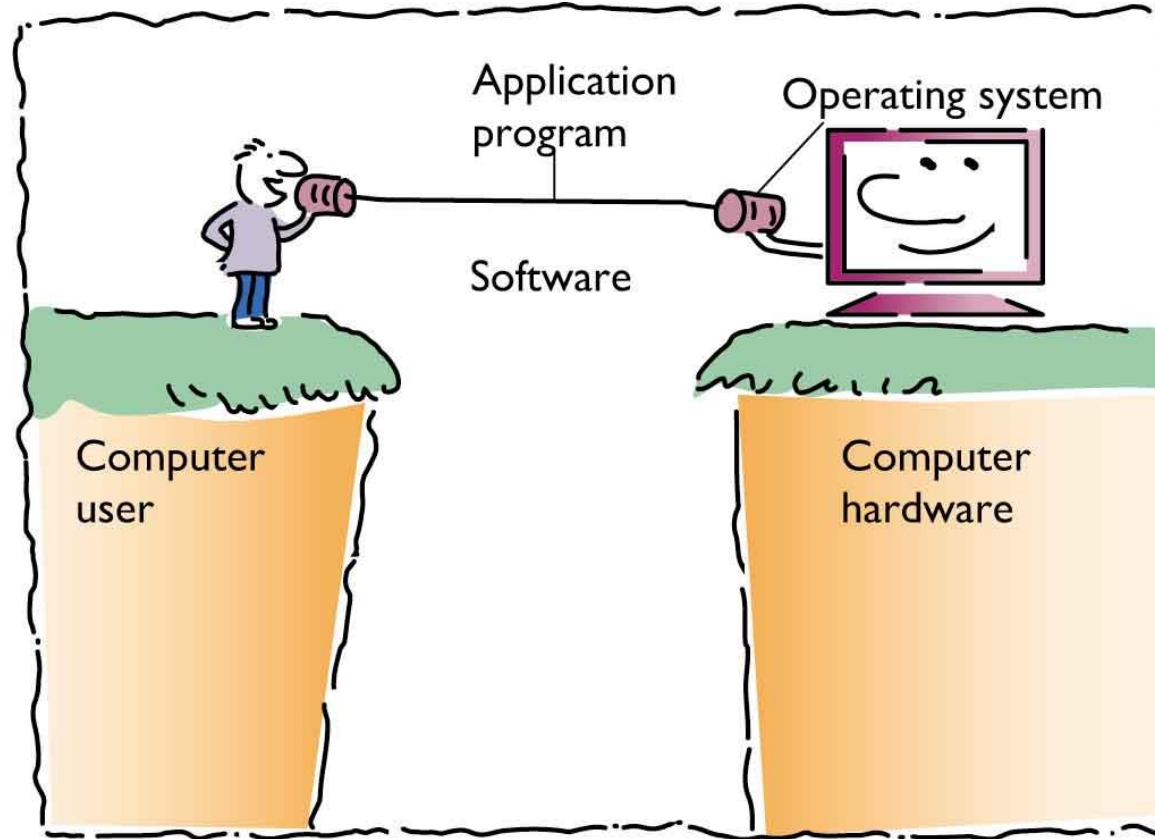


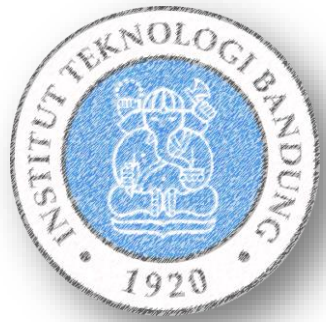


# Tujuan

- Mahasiswa mampu menjelaskan konsep algoritma sebagai teknik computational thinking dalam pemecahan persoalan
- Mahasiswa mampu menjelaskan konsep algoritma (sequential control, selection, repetition, dan control abstraction) untuk mendekomposisi secara algoritmik solusi untuk persoalan sederhana
- Mahasiswa mampu menerjemahkan konsep algoritma sederhana dalam conceptual tool (flow chart/activity diagram/pseudocode)

# User – Software – Hardware

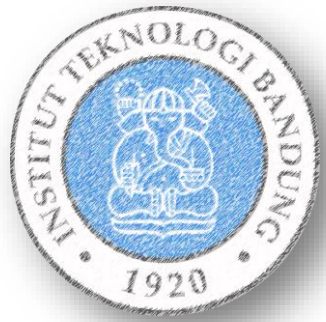




# Perangkat Lunak

- **Perangkat Lunak (*software*)** memungkinkan pengguna mengkomunikasikan suatu persoalan kepada komputer dan komputer memberikan solusinya kepada pengguna
  - Tanpa perangkat lunak, komputer hanya mesin bodoh!

**Software = program + data + dokumentasi**

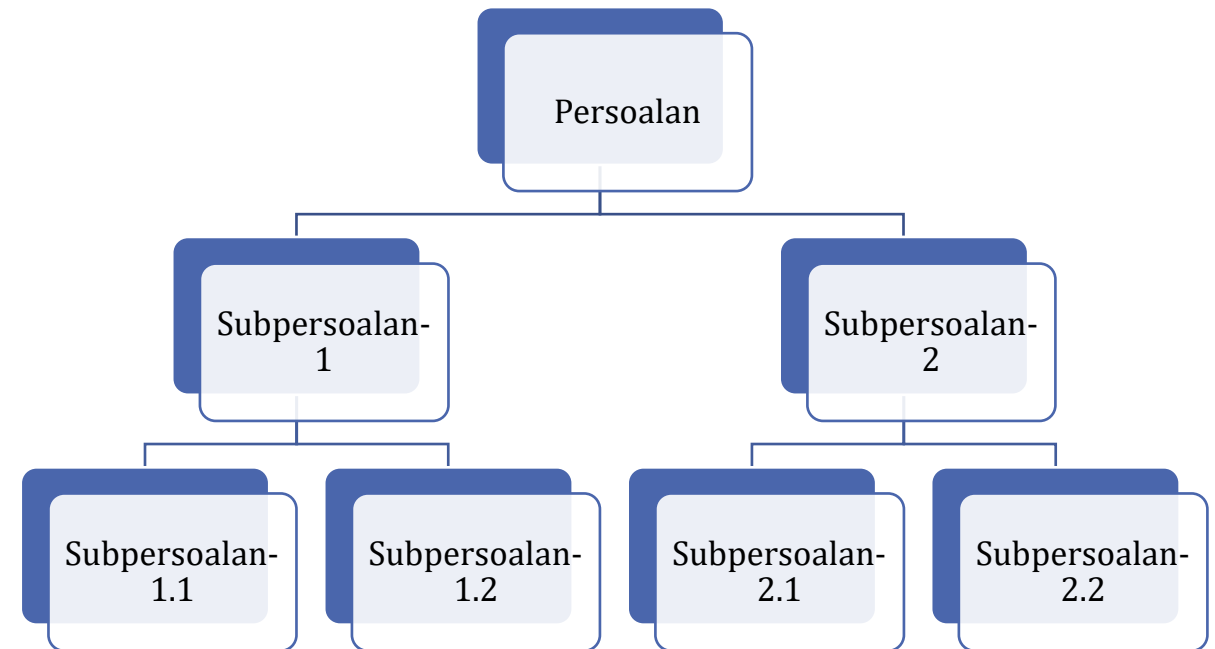


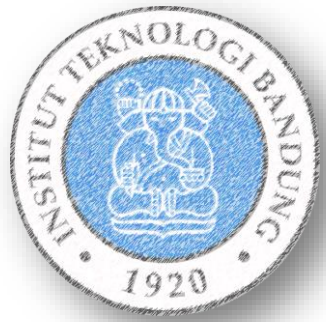
# *Problem Solving* dengan Pemrograman

- **Pemrograman (*programming*)**: adalah salah satu bentuk penyelesaian persoalan (*problem solving*) di mana persoalan serta solusinya direpresentasikan dalam bentuk yang bisa diproses oleh **komputer**
- Secara umum terdiri atas langkah-langkah sbb.:
  - Memahami persoalan
  - Menyusun rencana untuk menyelesaikan persoalan → **algoritma**
  - Menyusun solusi berdasarkan rencana → **program komputer**
  - Mengevaluasi solusi

# Dari Persoalan Menjadi Program (2)

- Berdasarkan pemahaman terhadap persoalan, *programmer* menyusun daftar persoalan dan mendekomposisi menjadi sub-persoalan
  - Setiap sub-persoalan berpotensi menjadi modul program
  - Dalam kuliah ini, modul program akan disusun dalam **subprogram** (*tunggu beberapa minggu lagi*)
- **Top down design**: Mulai dari persoalan besar didetilkkan sampai pada akhirnya menjadi langkah-langkah penyelesaian sub-persoalan → **algoritma**





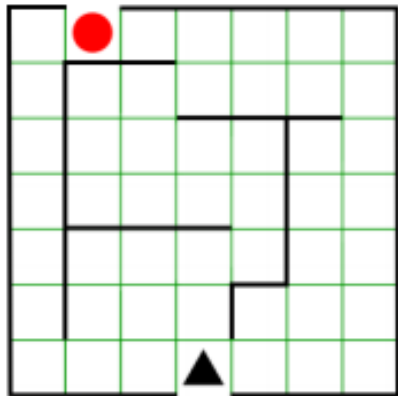
# Dari Ide Menjadi Algoritma

- **Algoritma**: himpunan prosedur langkah per langkah untuk menyelesaikan suatu [sub]persoalan
- Dapat ditulis dengan menggunakan teks atau gambar:
  - **Pseudocode** (contoh: notasi algoritmik) → teks; persilangan antara bahasa manusia dan bahasa pemrograman
  - **Flowchart** → diagram
- Algoritma disusun dengan memanfaatkan *control structure* yang menentukan bagaimana urutan langkah dieksekusi
  - *Sequence*: langkah-langkah yang dieksekusi berurutan
  - *Conditional* (percabangan): pilihan langkah
  - *Repetition/loop* (pengulangan): pengulangan langkah



# Contoh-1: Keluar dari Labirin

Mira perlu menemukan jalan untuk keluar dari sebuah labirin dan meminta anda untuk memberikan arahan. Dia memasuki labirin dari bawah (segitiga hitam) dan harus mencapai pintu keluar pada bagian atas (lingkaran merah besar).







## **Tantangan:**

Urutan gerakan manakah yang jika diulang dua kali oleh Mira dapat membuatnya mencapai pintu keluar?

Sumber: Bebras Indonesia ([www.bebras.or.id](http://www.bebras.or.id)). Kode Soal: I-2017-CH-01A

Namun, Mira hanya dapat mengingat empat gerakan berikut:

Kode Gerakan	Artinya	Ilustrasi
A	Berjalan satu langkah maju dan menghadap ke kiri	
B	Berjalan satu langkah maju dan menghadap ke kanan	
C	Berjalan dua langkah maju dan menghadap ke kiri	
D	Berjalan dua langkah maju dan menghadap ke kanan	

Mira bisa membentuk sebuah gerakan panjang yang dibentuk dari empat gerakan A, B, C, atau D. Satu gerakan bisa diulang beberapa kali. Urutan gerakan ini dapat membawa Mira keluar dari labirin dengan dua kali perulangan.



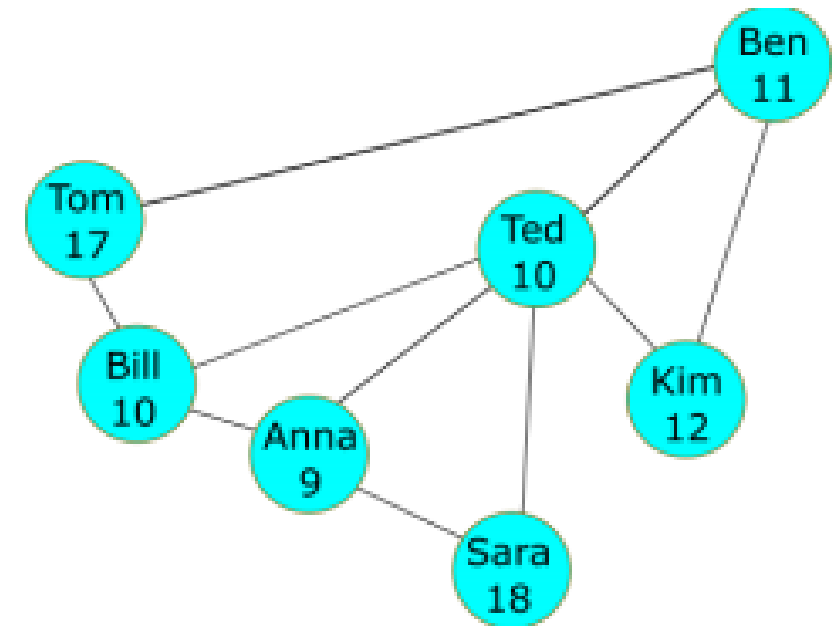
# Contoh-2: Klub Berbagi Buku

Ada tujuh (7) siswa yang gemar membaca buku dan mereka membentuk klub untuk berbagi buku. Jika ada satu buku baru diperoleh (dan dibaca) seorang siswa, kemudian ia akan meneruskan meminjamkan ke anggota klub lainnya dengan cara berikut. Tidak setiap siswa menjadi sahabat siswa lainnya, maka seorang siswa hanya meneruskan meminjamkan buku ke siswa yang bersahabat dengannya. Jika seorang siswa mempunyai beberapa sahabat, maka sahabat yang paling muda yang akan dipinjami terlebih dulu, yang belum pernah meminjam buku itu. Kalau semua sahabatnya sudah pernah meminjamnya, maka ia akan mengembalikan ke siswa yang sebelumnya meminjamkan buku itu kepadanya.

Diagram disamping menunjukkan tujuh siswa dan garis-garis menunjukkan hubungan "sahabat" itu. Setiap simpul berisi informasi nama dan umur.

## ***Tantangan:***

Ben selesai membaca sebuah buku baru dan ingin berbagi dengan semua anggota klub dan selain Ben belum ada yang pernah membacanya. **Siapa yang akan menjadi pembaca terakhir dari buku tersebut?**



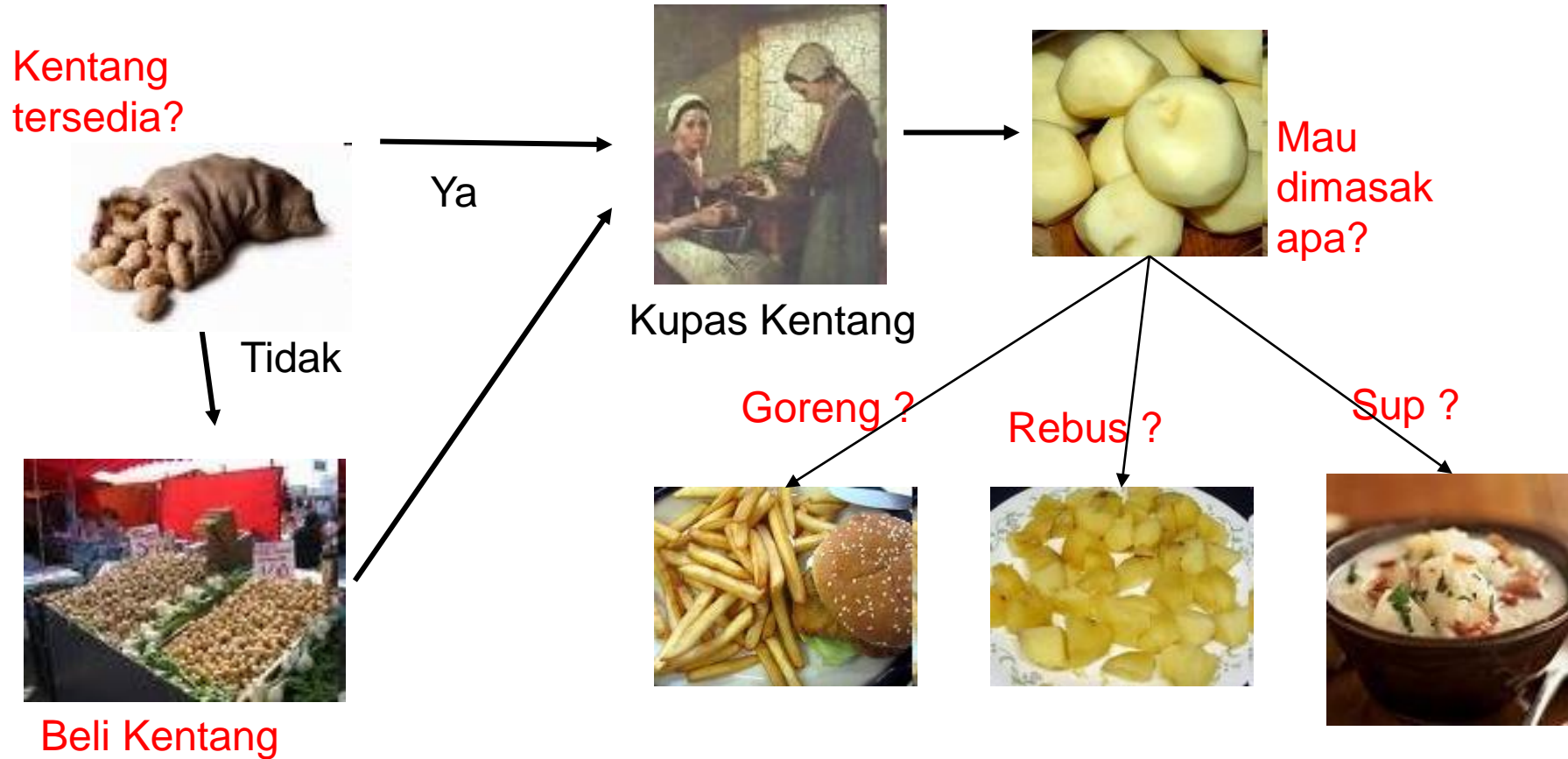
Sumber: Bebras Indonesia ([www.bebras.or.id](http://www.bebras.or.id)). Kode Soal: I-2017-IR-07



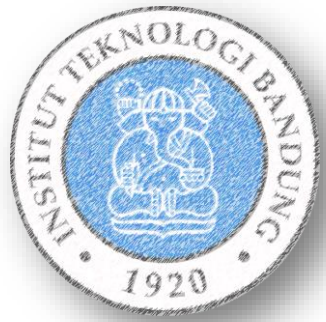
## Contoh-3: Memasak Kentang untuk Makan Malam

- Untuk makan malam, sejumlah kentang harus dikupas dan dimasak
- **Keadaan awal:** kantong kentang tersedia di dapur
  - Belum jelas apakah kentang tersedia cukup atau tidak
- **Keadaan akhir:** masakan dengan bahan dasar kentang tersedia dan siap dihidangkan untuk makan malam

# Memasak kentang untuk makan malam

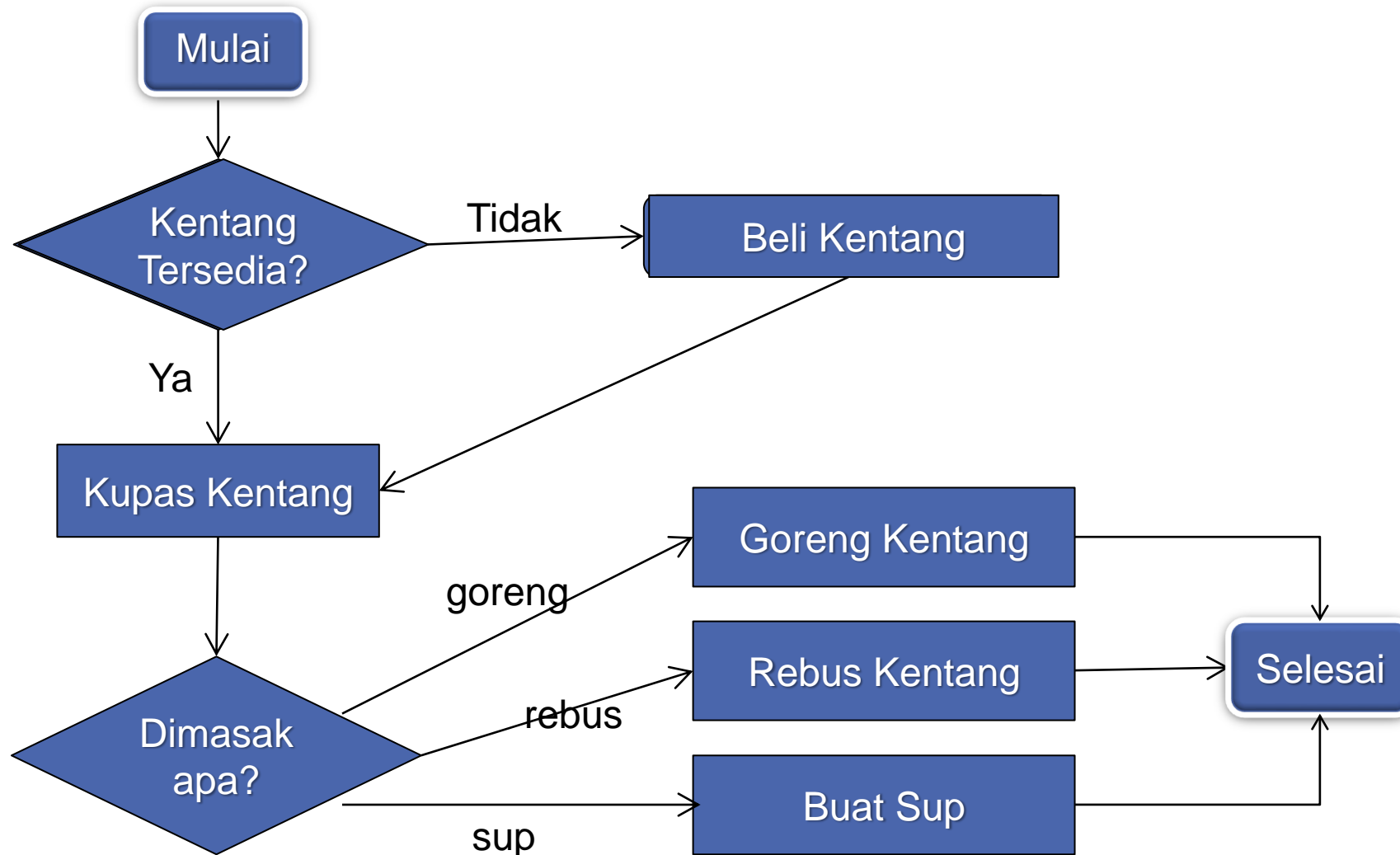
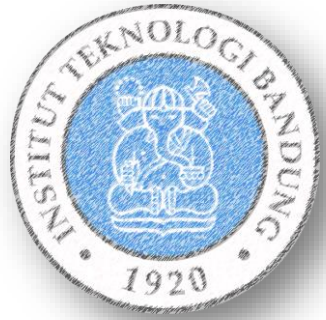


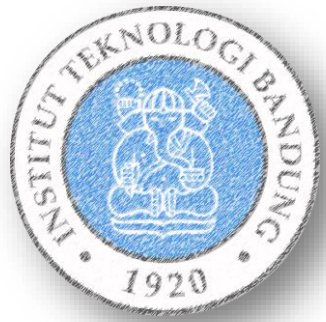
# Memasak Kentang untuk Makan Malam - Pseudocode



```
if kentang_tersedia? = tidak then  
    Beli_Kentang  
{ Di titik ini kentang sudah tersedia }  
Kupas_Kentang  
if pilihan_masakan = goreng then  
    Goreng_Kentang  
else if pilihan_masakan = rebus then  
    Rebus_Kentang  
else { pilihan_masakan = sup }  
    Buat_Sup
```

# Memasak Kentang untuk Makan Malam - Flowchart

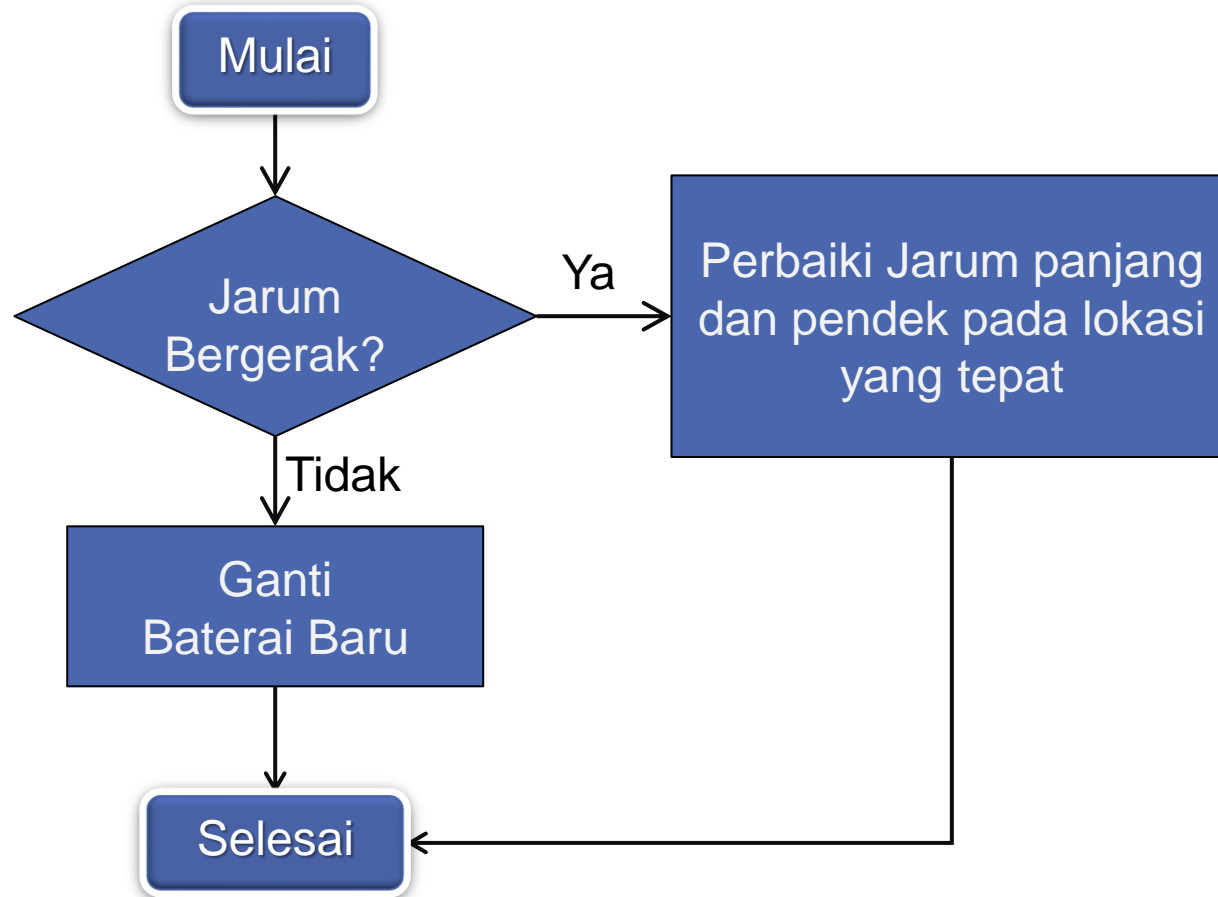




## Contoh-4: Perbaiki Jam Dinding

- **Keadaan awal:** Jam dinding tidak menunjukkan waktu yang tepat
- **Keadaan akhir:** Jam dinding menunjukkan waktu yang tepat
- Bila jarum tidak bergerak, ganti baterai
- Jika bergerak berarti baterai masih hidup tinggal dilakukan perbaikan letak jarum jam

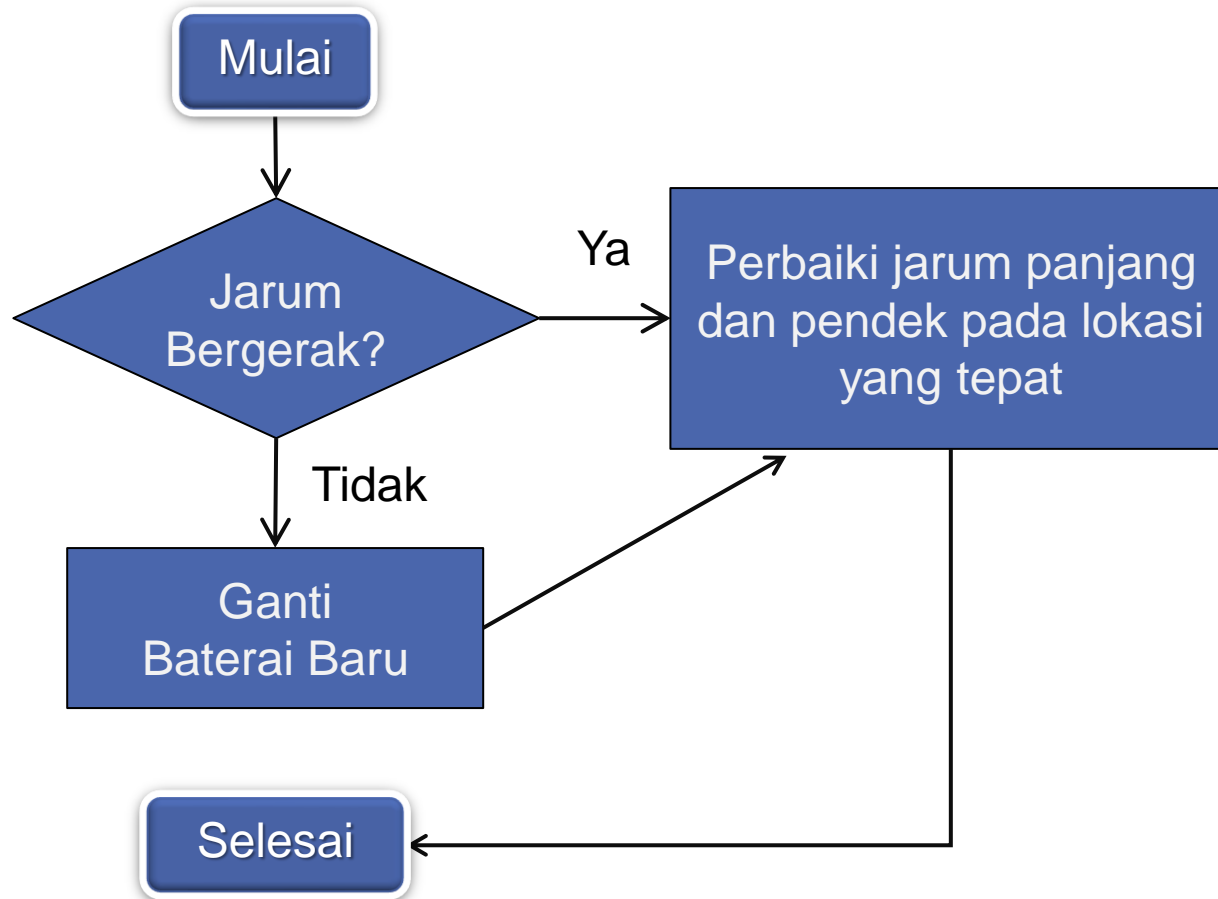
# Perbaiki Jam Dinding: Flowchart-1



Apa yang salah dengan solusi ini??

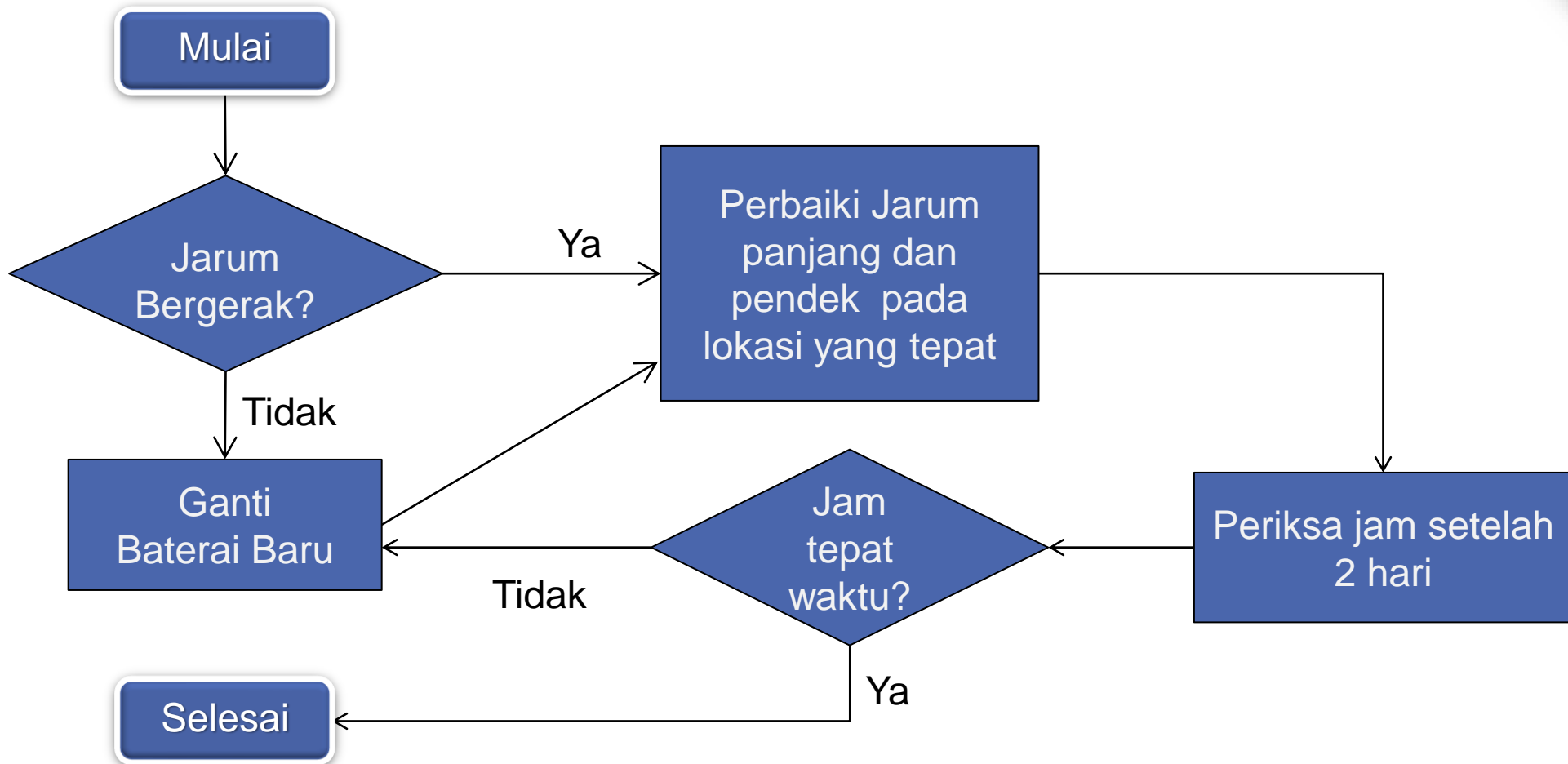


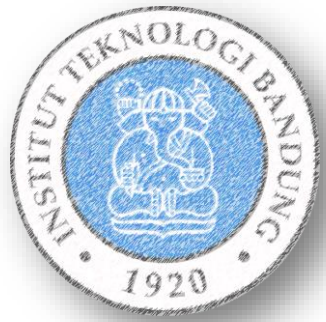
# Perbaiki Jam Dinding: Flowchart-2



Bagaimana jika ternyata setelah dua hari jam kembali tidak tepat?

# Perbaiki Jam Dinding: Flowchart-3





# Perbaikan Jam Dinding: Pseudocode

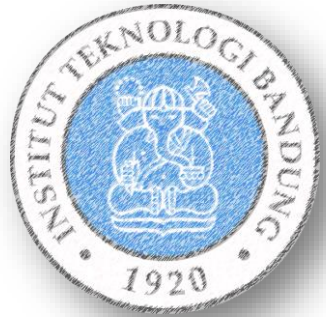
Setara dengan Flowchart-3

```
if jarum_bergerak? = tidak then  
    Ganti_Baterai_Baru  
{ Di titik ini jarum jam sudah pasti bergerak }  
repeat  
    Perbaiki_Letak_Jarum_Jam  
    Periksa_Jam_Setelah_2_Hari  
    if jarum_jam_tepat? = tidak then  
        Ganti_Baterai_Baru  
until (jarum_jam_tepat? = ya)
```

# Contoh-5

## Which photo do you want?

2014-JP-03



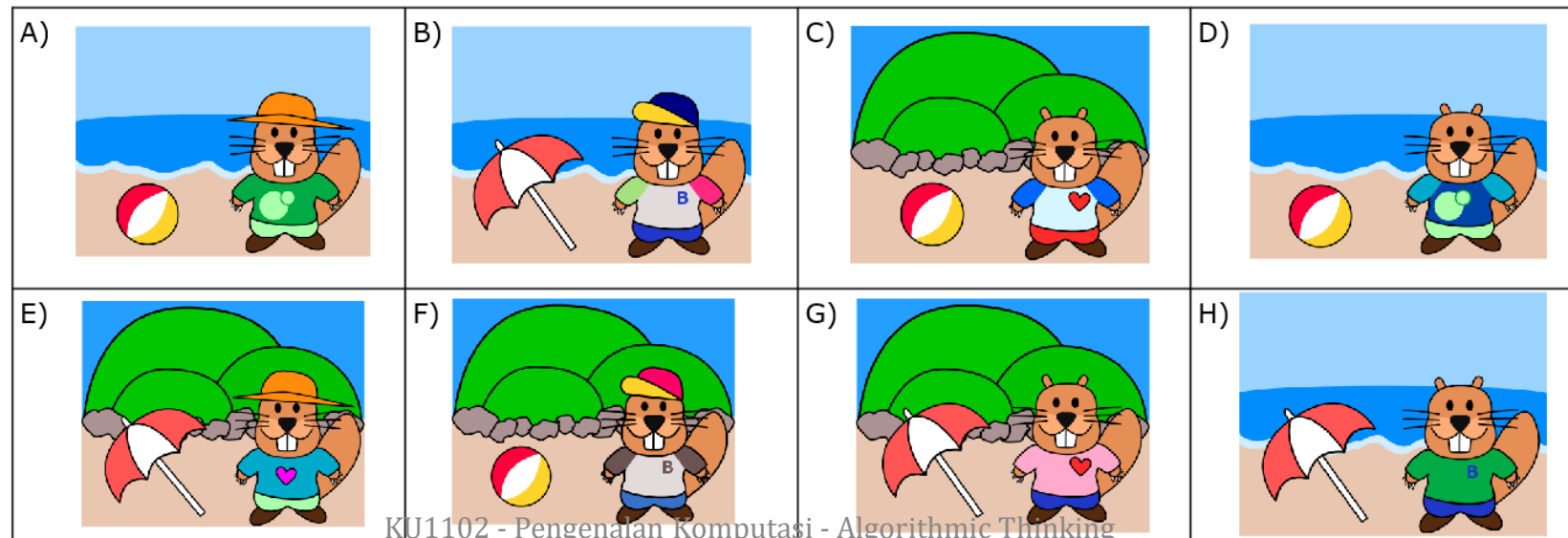
Johnny has 8 photos. He would like to give one of them to Bella. He asks her some questions to find out which photo she wants:

“Do you want a photo with a beach umbrella?” “Yes.”

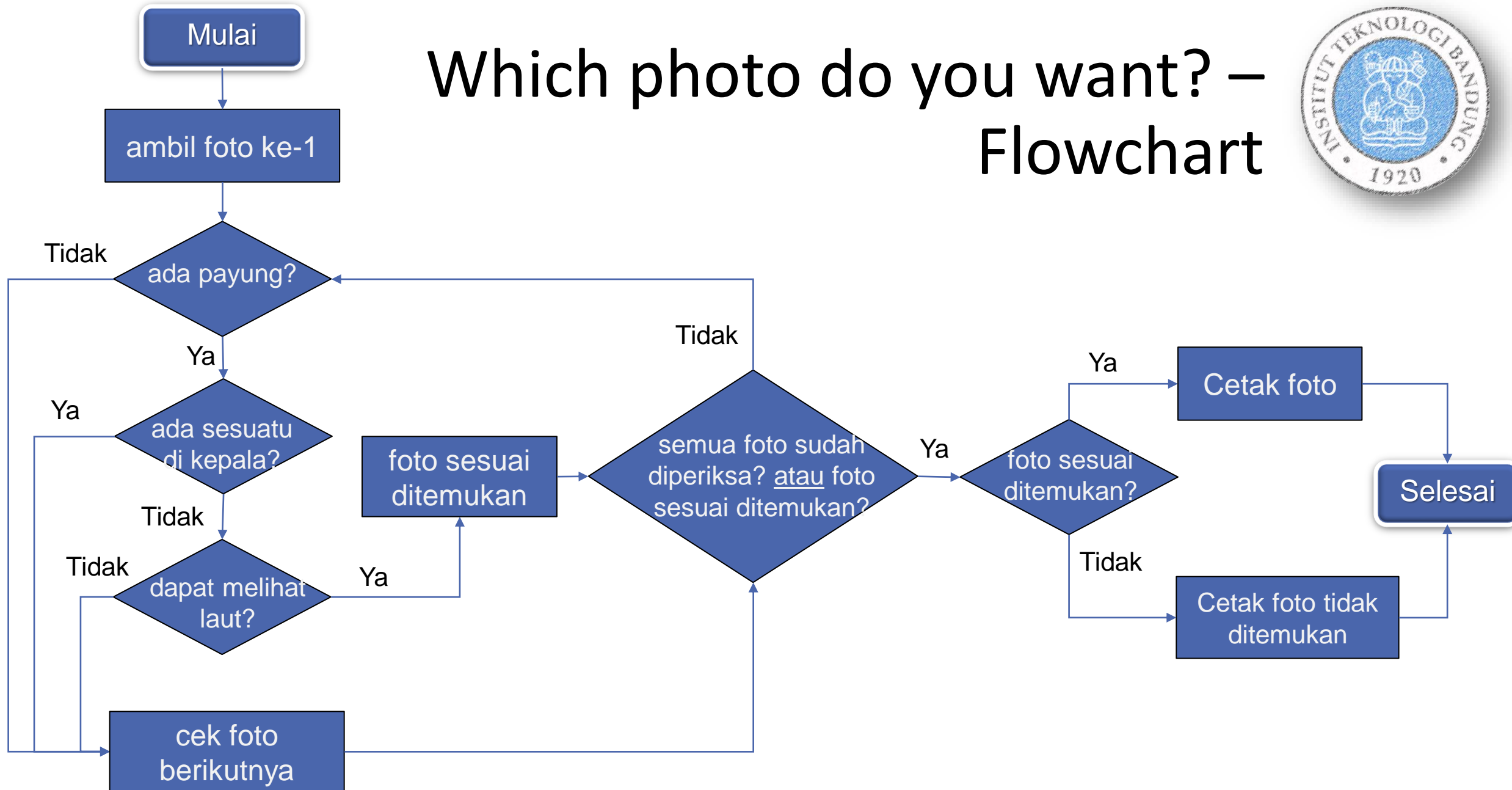
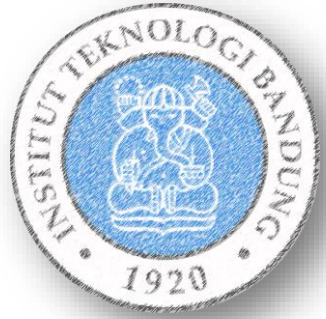
“Do you want a photo where I have something on my head?” “No.”

“Do you want a photo where you can see the sea?” “Yes.”

**Which photo should Johnny give to Bella? (Source: Bebras Challenge)**



# Which photo do you want? – Flowchart



# Which photo do you want? – Pseudocode

```
ambil_foto_ke_1
repeat
    if ada_payung? = ya then
        if ada_sesuatu_di_kepala? = tidak then
            if dapat_melihat_laut? = ya then
                foto_sesuai_ditemukan
            else
                cek_foto_berikutnya
        else
            cek_foto_berikutnya
    else
        cek_foto_berikutnya
until (semua_foto_sudah_diperiksa) or
      (foto_sesuai_ditemukan)

if (foto_sesuai_ditemukan) then
    cetak_foto
else
    cetak_foto_tidak_ditemukan
```

# Contoh-6: Jemputan Undangan

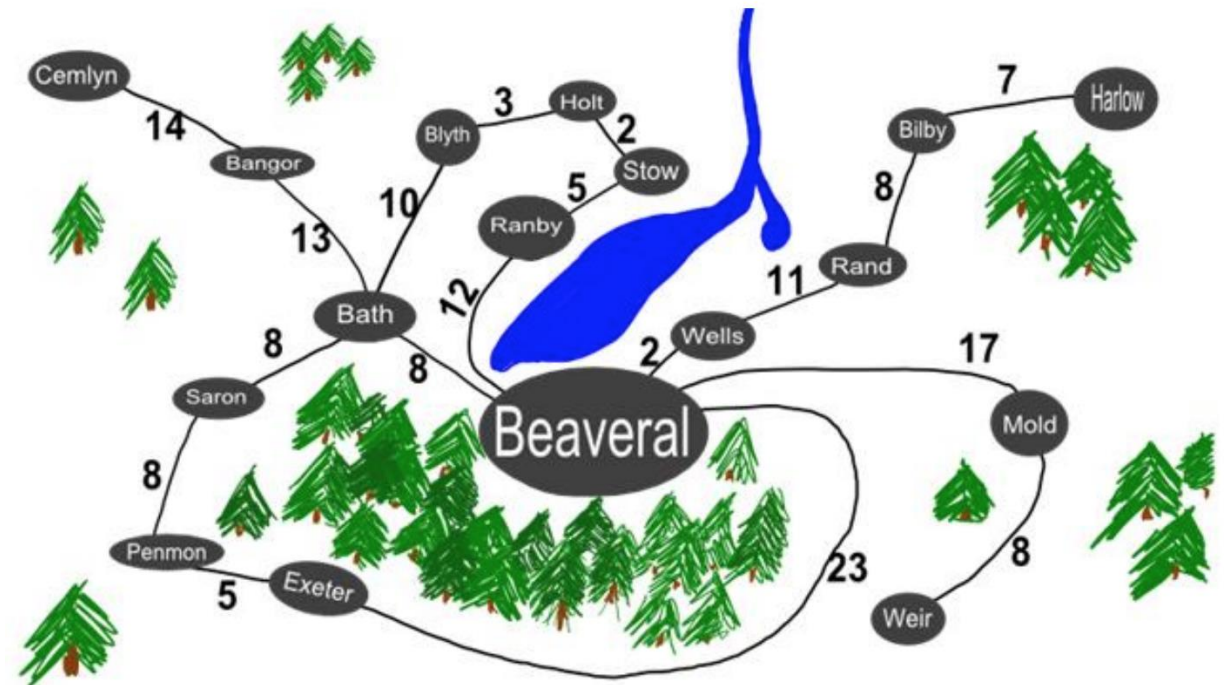
Patra tinggal di desa Beaveral. Ia ingin mengundang teman-temannya untuk merayakan ulangtahunnya. Namun ia hanya mengundang teman-teman yang tinggal tidak lebih dari 20 km dari rumahnya agar tak kemalaman pulang. Jarak rumah teman-temannya dimunculkan dalam angka pada peta sebagai berikut.

## **Tantangan:**

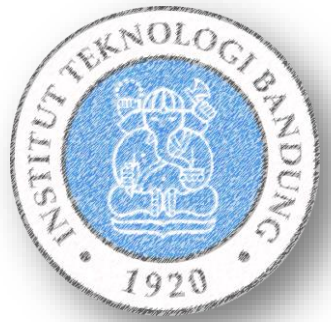
Teman-teman yang diundang, dan rumahnya berjarak lebih dari 17 akan disediakan jemputan. Tentukan teman-teman Patra yang diundang dan akan dijemput.

## **Tantangan tambahan:**

- Tuliskan langkah-langkah Anda mendapatkan solusi tersebut dalam bentuk *flowchart/pseudocode*
- Dapatkah Anda menggeneralisasi langkah-langkah algoritmik tsb sedemikian hingga dapat digunakan pada peta-peta yang lain?





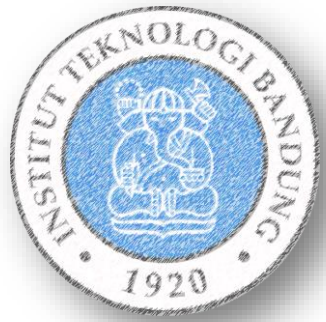


# Pengantar Bahasa Pemrograman



# Dari Algoritma Menjadi Program

- *Programmer* mengubah **algoritma** menjadi **kode program** komputer dengan menggunakan **bahasa pemrograman**
- Proses untuk menuliskan kode program berdasarkan algoritma disebut sebagai ***coding***
  - File hasil menuliskan kode program: ***source code*** (kode sumber)
- Setiap pernyataan dalam algoritma ditranslasikan secara detil ke dalam kode program
- ***Compiler/interpreter*** akan mentranslasi kode program dalam bahasa pemrograman tertentu menjadi bentuk yang dipahami oleh komputer



# Bahasa Pemrograman

- Setiap komputer memproses instruksi dalam **bahasa mesin** (*machine language*)
  - Kode-kode numerik yang digunakan untuk mengerjakan operasi-operasi dasar:
    - *Adding and subtracting numbers*
    - *Comparing numbers*
    - *Moving numbers*
    - *Repeating instructions*
- Programmer menggunakan **bahasa pemrograman tingkat tinggi** (high-level languages) untuk menuliskan kode program
  - Pascal, C/C++, Matlab, Python, Fortran, Basic, Java, dll.

# Compiler vs Interpreter

- **Compiler:** membaca seluruh kode program sekaligus dan menerjemahkannya menjadi kode yang dipahami mesin komputer

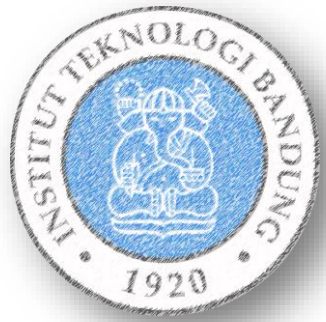


- Contoh: C/C++, Pascal, Fortran

- **Interpreter:** membaca baris kode satu per satu



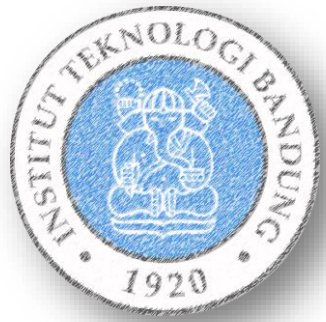
- Contoh: MATLAB, Python



# Paradigma Pemrograman

- **Paradigma pemrograman** adalah sudut pandang penyelesaian persoalan dengan program komputer
- Contoh paradigma pemrograman:
  - **Paradigma prosedural (imperatif)** → akan diajarkan di kuliah ini
  - Paradigma berorientasi objek
  - Paradigma deklaratif
  - Dll.
- **Paradigma prosedural (imperatif)**: Program didasari oleh **strukturisasi informasi** di dalam memori dan **manipulasi** dari informasi yang disimpan tersebut

**Program = Algoritma + Struktur Data**



# Bahasa Pemrograman Prosedural

- Ada **RIBUAN** bahasa pemrograman di dunia saat ini → termasuk bahasa pemrograman prosedural
- Tidak mungkin semua bahasa pemrograman dipelajari di kuliah
- Oleh karena itu yang diajarkan adalah “belajar pemrograman” → melalui **pola pikir komputasional** dan **paradigma pemrograman prosedural**
- Bahasa pemrograman yang diajarkan di Pengenalan Komputasi: C/C++, Pascal, MATLAB, Python, Fortran, “Pemrograman Visual”
- Di kelas ini akan diajarkan: ...