



UNIVERSITAS
GADJAH MADA

Meeting 1

Introduction to Computer and Programming

Fundamentals of Programming
TKU211131

DTETI FT UGM

Bimo Sunarfri Hantono
bhe@ugm.ac.id

LOCALLY ROOTED,
GLOBALLY RESPECTED

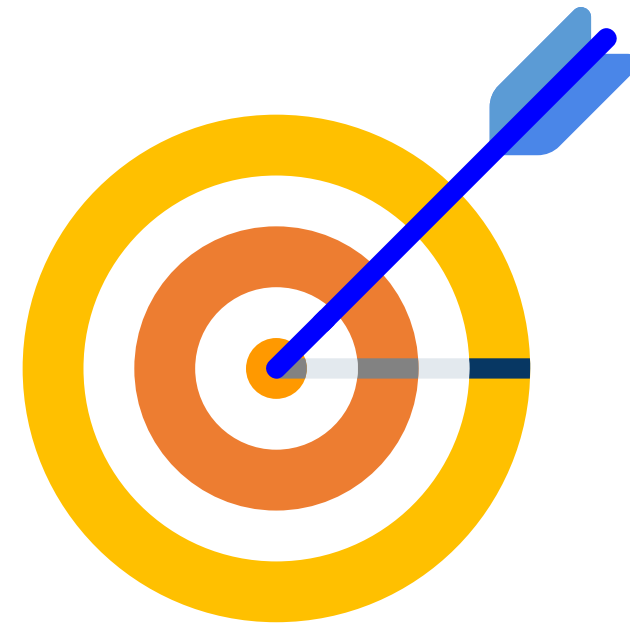
ugm.ac.id



<http://ugm.id/SurveiProgDasC>

Topics

- Introduction to the Course
- Overview of Computers Hardware
- Overview of Computers Software
- Intro to Programming





UNIVERSITAS
GADJAH MADA

Introduction to the Course

- **Dr. Bimo Sunarfri Hantono, S.T., M.Eng.**
- Dr.Eng. Silmi Fauziati S.T., M.T.
- Dr. Warsun Najib, S.T, M.Sc.

Required Course: **3 SKS**

- 150 minutes/week for lectures
- 150 minutes/week for structured assignments
- 150 minutes/week for independent study

Student Outcome:

- Basic knowledge and engineering principles
- Development of engineering solutions
- Modern Tools Utilization

Course Description:

This course will discuss about program development steps, ranging from defining problem, determining program input & output, and determining steps by utilizing operator and operands, data types, structure, programming control. This course also elaborates programming strategies and modularity.

Learning Outcome:

1. Students are able to explain the concept of programming
2. Students are able to develop procedural paradigm program
3. Students are able to utilize various data types and basic data structures to develop the program
4. Students are able to implement modular programming strategies and choose various programming strategies to get a good, effective, and efficient program

Components for Final Grade:

- Assignments (%)
- Quizzes (%)
- Midterm Exam (UTS) (%)
- Final Exam (UAS) (%)

1. Intro to programming in general
2. Overview of C/C++ language
3. Selection
4. Repetition
5. Combination of selection and repetition
6. Fundamental data types
7. Array and array processing
8. User-defined data types
9. Modularity
10. Recursion
11. Class and data abstraction
12. OOP and UML
13. Inheritance
14. Operator overloading

To Pass this Course

- Attend at least **75%** of lectures
 - a requirement to take the final exam.
- Participate in quizzes, complete assignments, and take exams.
- Pass each defined Learning Outcome (LO).
- All LOs will be evaluated. The final grade for each LO is calculated from several assessment components.
- Each LO must receive a minimum score of 50 (on a scale of 0-100).

Course Learning Materials

- Go to website eLok UGM
- Navigate to the “Course” menu
- Search for “DTETI-S1-2025-1-Pemrograman Dasar.”
- Look for the course taught by Bimo Sunarfri Hantono, Silmi Fauziati, and Warsun Najib
- Enter the course corresponding to your class.

**DTETI-2025-1-S1-
Pemrograman Dasar-A**



[Get access →](#)

**DTETI-2025-1-S1-
Pemrograman Dasar-B**



[Get access →](#)

**DTETI-2025-1-S1-
Pemrograman Dasar-C**



[Get access →](#)



UNIVERSITAS
GADJAH MADA

Overview of Computers Hardware

Pre-test

Differentiate between
these computers!



What is a Computer?

- A computer is an **electronic machine** that can follow instructions to process data into useful information.
- Works digitally: only understands **0 (OFF)** and **1 (ON)**.
- Used everywhere from **supercomputers** to **smartphones**.

When non-nerds try
to act nerdy:



Types of Computers (1)

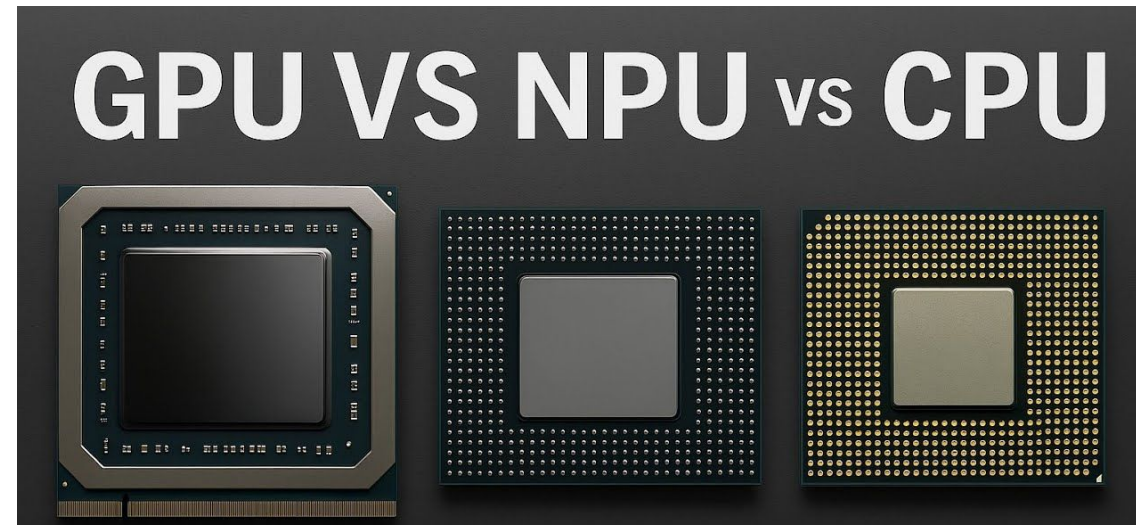
Type	Example	Common Use
Supercomputer	Frontier, Fugaku	Weather forecast, AI training
Mainframe	IBM z16	Banking, large data transactions



Types of Computers (2)

Type	Example	Common Use
Personal Computer	Laptop, Desktop	Work, study, gaming
Mobile Device	Smartphone, Tablet	Communication, apps
Wearables	Smartwatch, AR Glasses	Fitness, notifications





Processing Units:

- **CPU (Central Processing Unit):** General-purpose processor for running instructions.
- **GPU (Graphics Processing Unit):** Optimized for graphics rendering and parallel processing.
- **NPU (Neural Processing Unit):** Dedicated processor for accelerating AI and machine learning tasks.

Main Hardware Components (2)



Storage



Memory

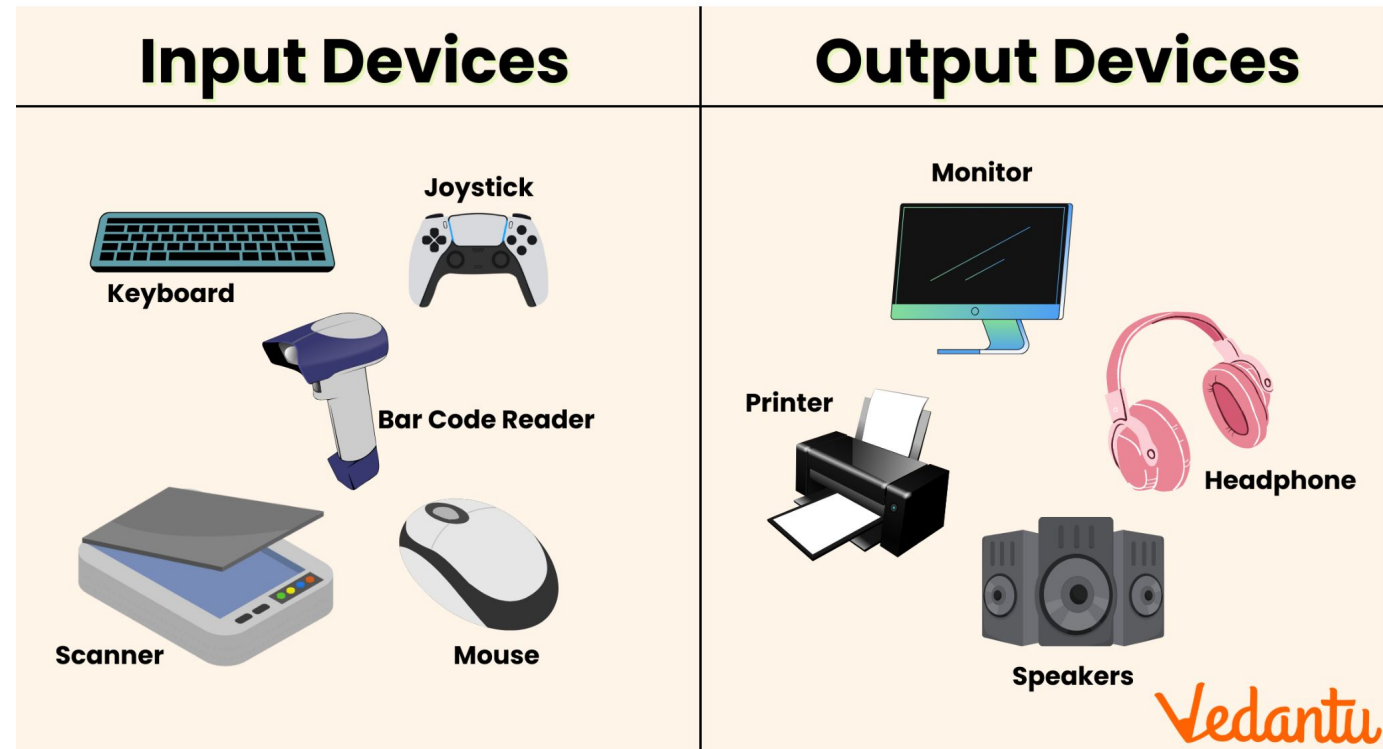
Memory and Storage:

- **RAM (Random Access Memory):** Temporary storage for active tasks and programs.
- **Storage:** Permanent data storage (SSD, HDD, cloud storage).

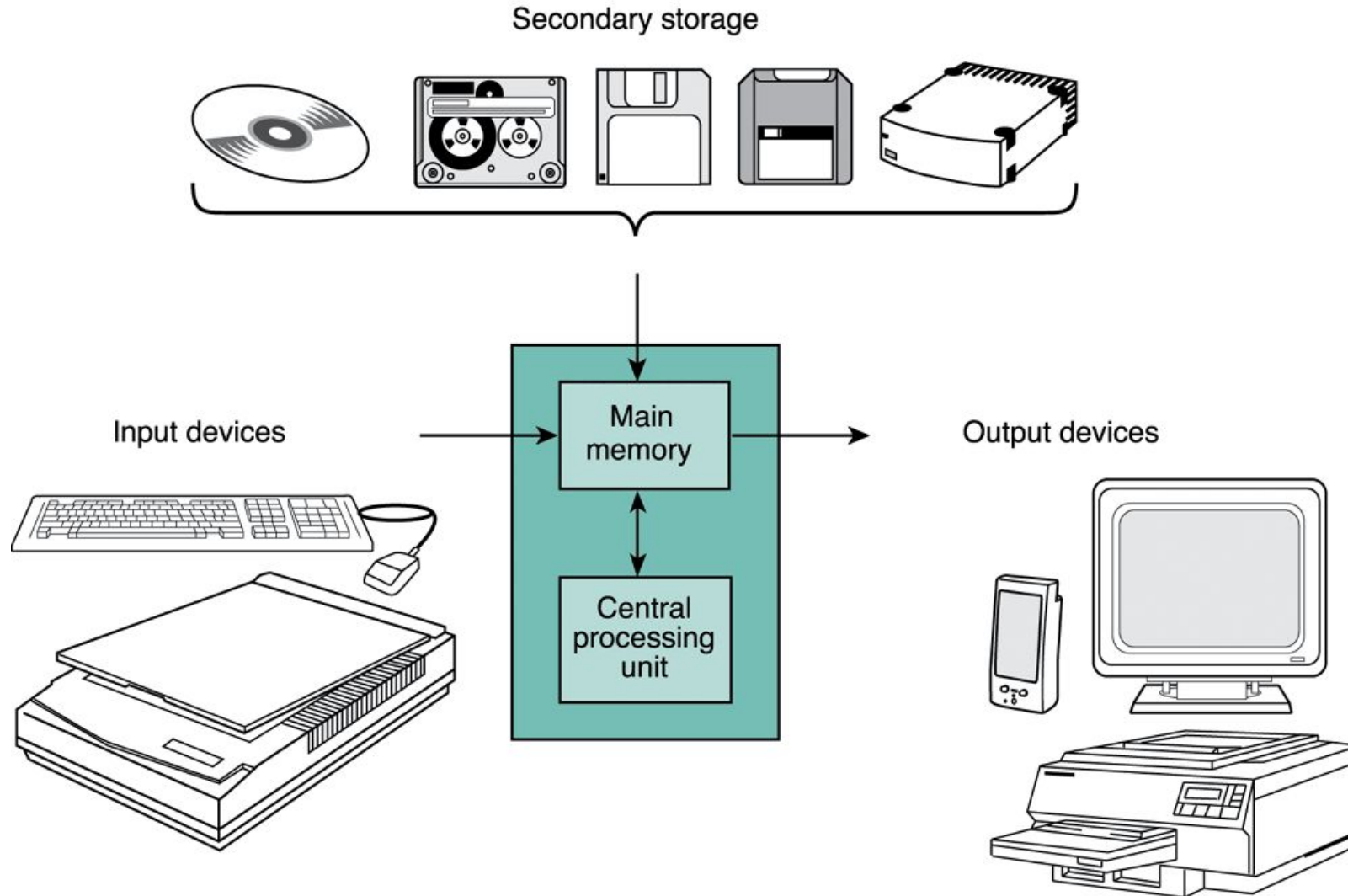
Main Hardware Components (3)

Input & Output Devices:

- Input Device: For **Inputting Data**:
 - Keyboard
 - Mouse
- Output Device: For **Displaying Results**:
 - Monitor
 - Printer



Interaction of Components in a Computer



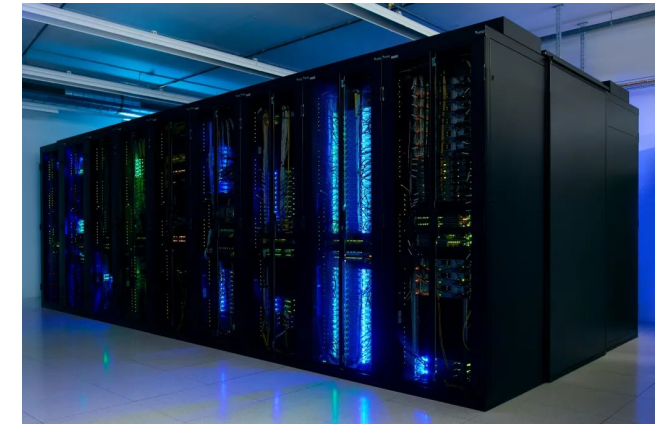
Modern Hardware Examples



Smartphones with AI chips.



IoT devices like smart home sensors.



Cloud servers running many apps at once.



UNIVERSITAS
GADJAH MADA

Overview of Computers Software

What is Software?

Software = instructions that tell hardware what to do.

Two main types:

1. **System Software** – runs the computer (Operating System).
2. **Application Software** – helps users do tasks.

Writing the code for
anti-virus software

Writing the virus to
sell the anti-virus
software

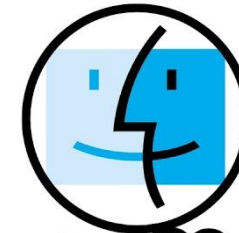


Operating System (OS)

An operating system is software that **controls the interaction** between hardware and the computer user and manages the allocation of computer resources. Loading the operating system into memory is called booting the computer.



Windows



macOS



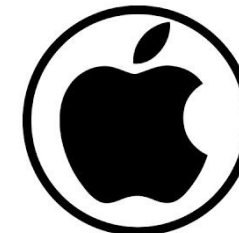
Linux



ChromeOS



Android



iOS



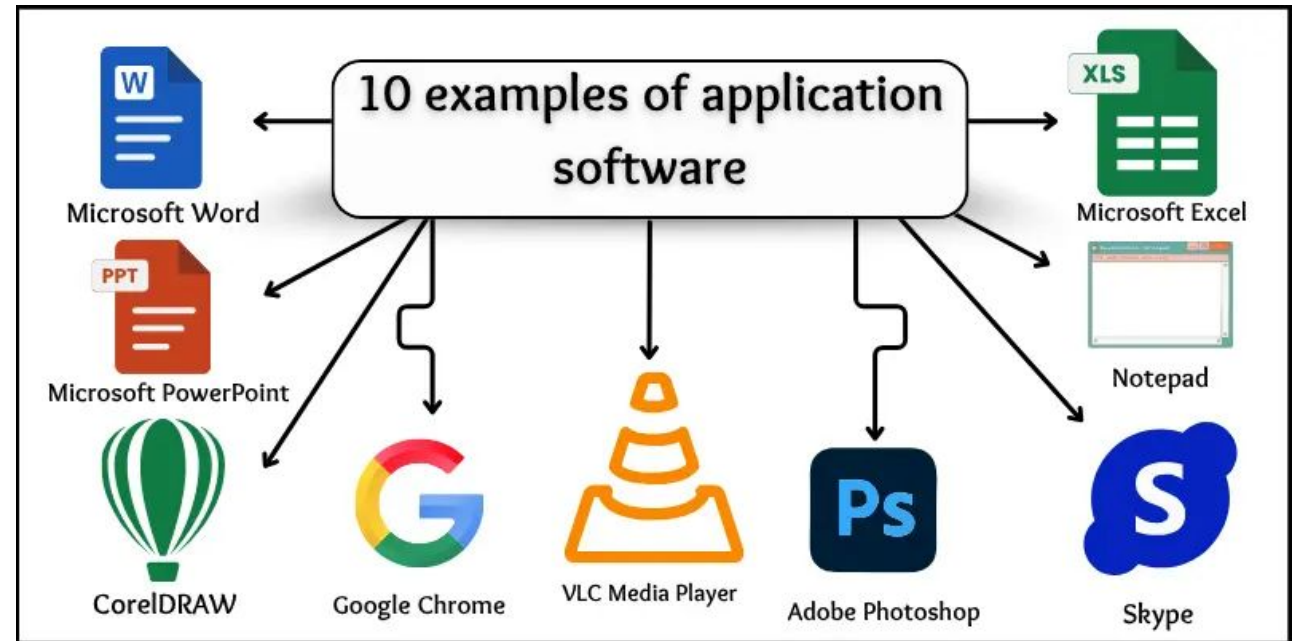
UNIX

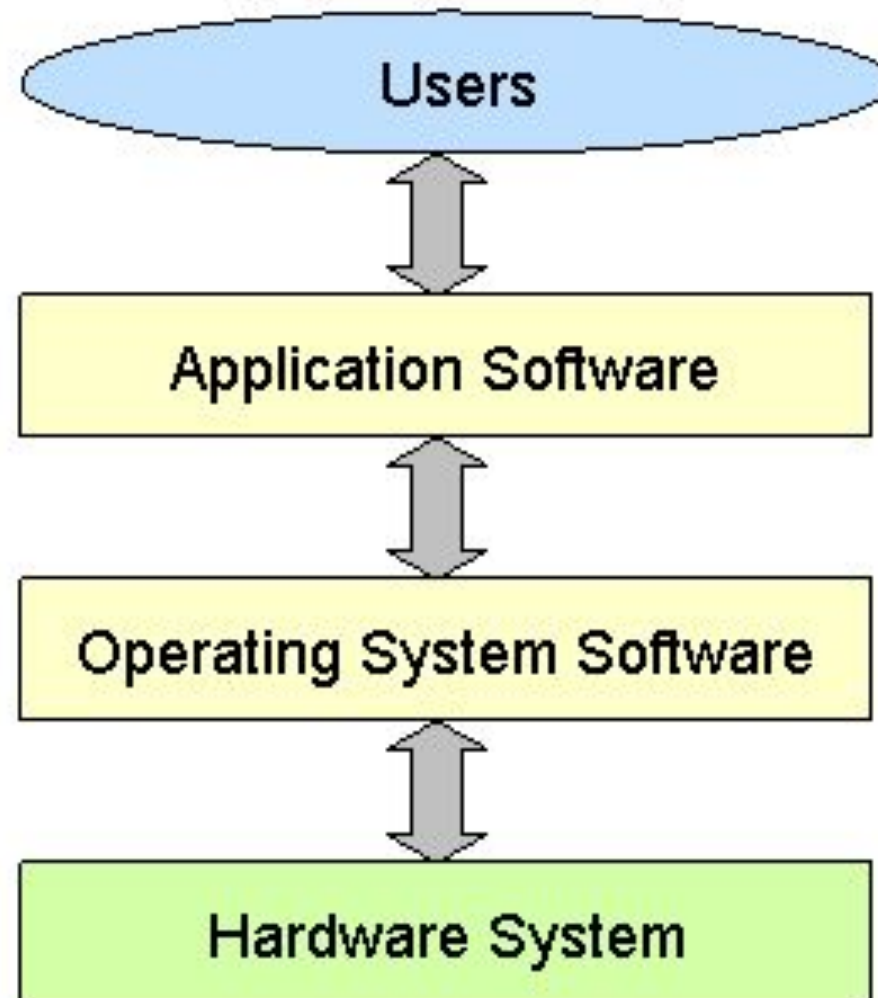


BSD

Application Software

- Application software is used for **specific tasks** such as word processing, accounting, or database management.
- Installing makes the application available on the computer by copying it to the computer's hard drive.



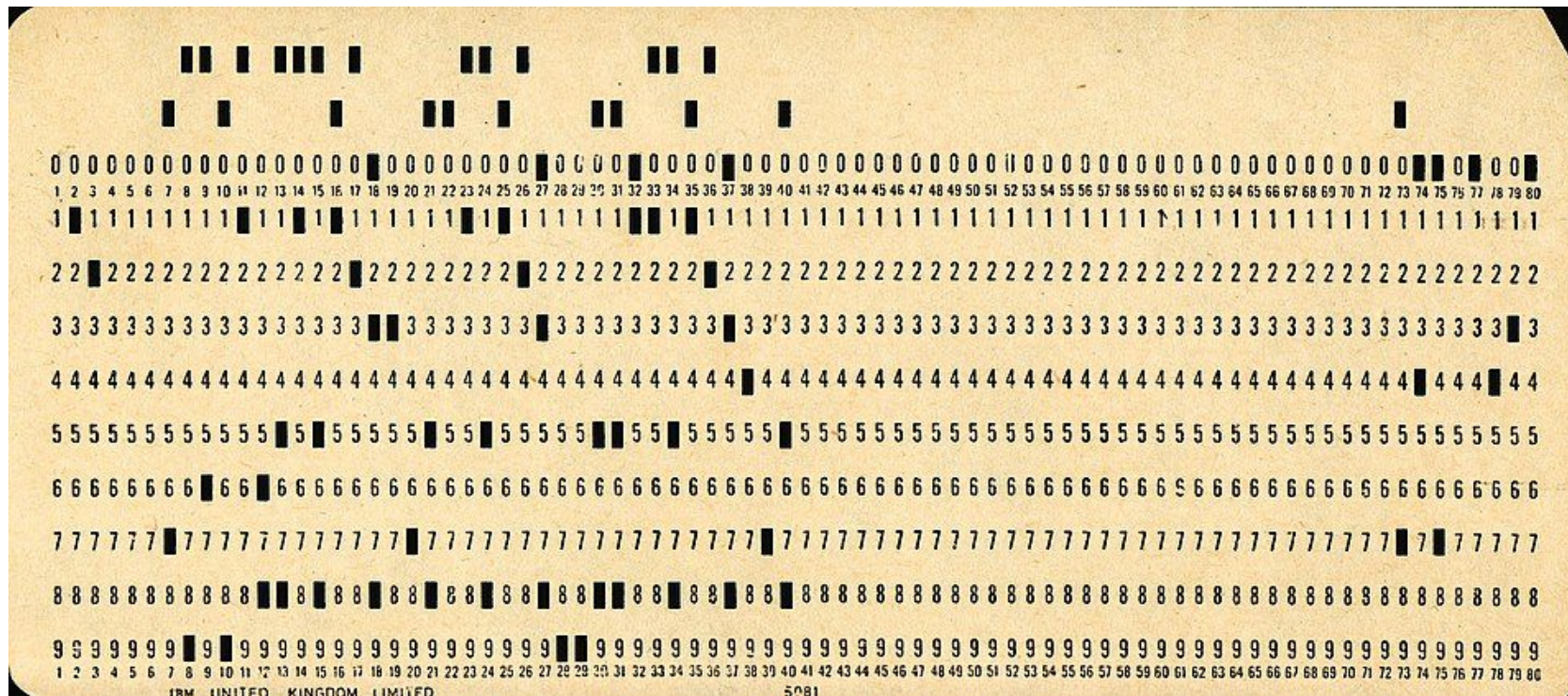




UNIVERSITAS
GADJAH MADA

Intro to Programming

Punched card: early stage of computer programming

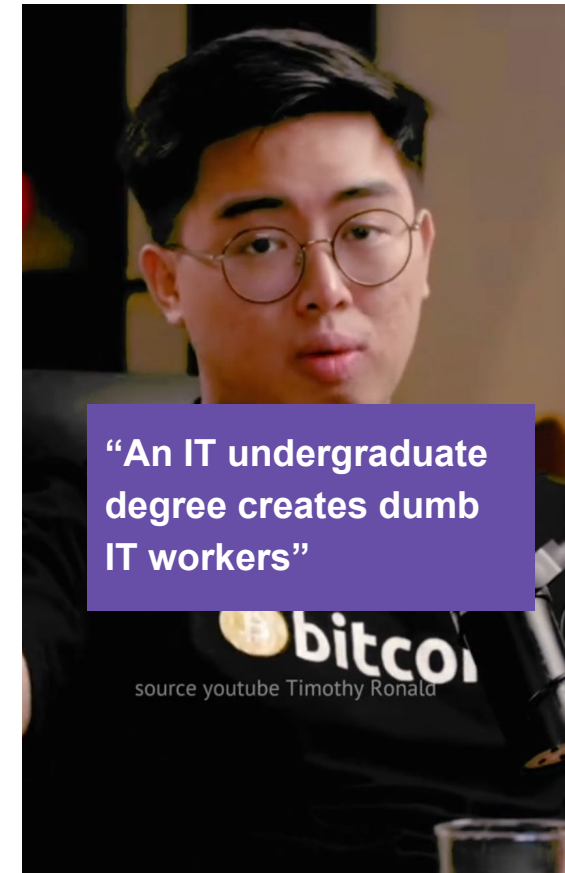


What is Programming?

Programming is the process of turning ideas into executable instructions.

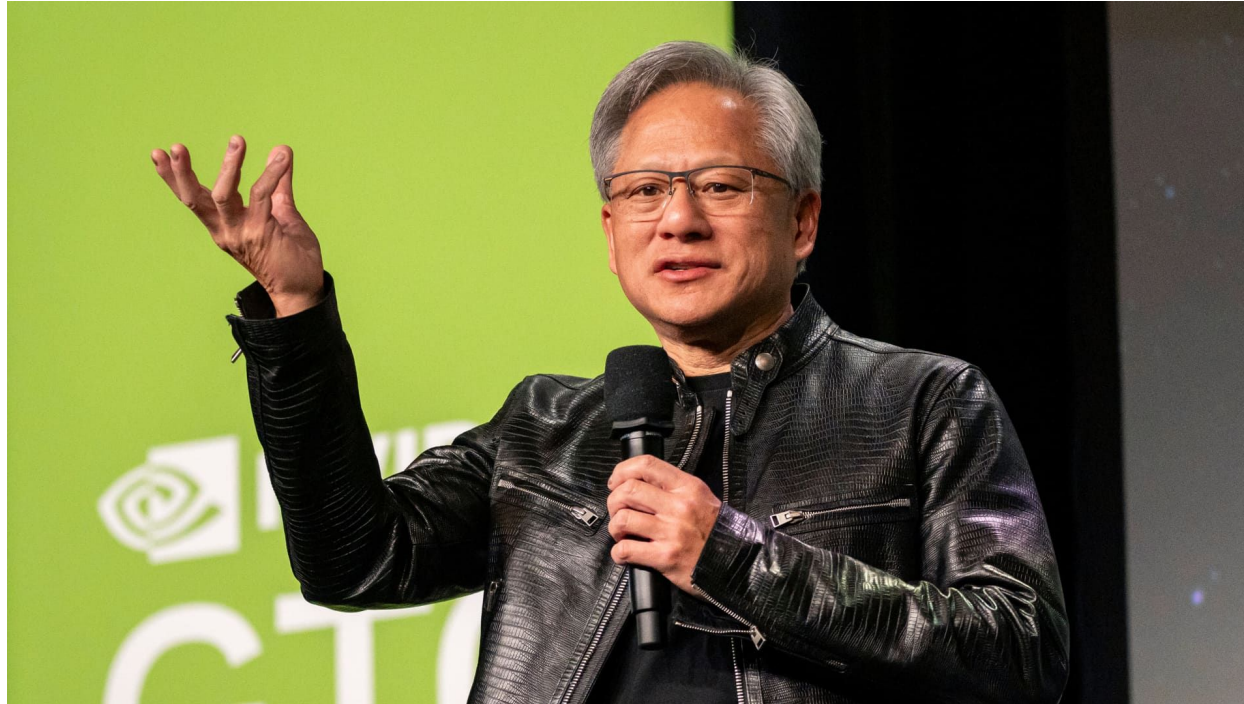
- Translate real-world problems into **data, rules, and steps**
- Choose representations (numbers, text, structures)
- Write **clear, unambiguous instructions** a computer can follow

Are IT fields irrelevant to today's problems?



https://www.tiktok.com/@timothyronaldarchive/video/7503462922249604358?is_from_webapp=1&sender_device=pc&web_id=7538609496621401607

Do We Still Need to Learn Coding?



*“We should stop telling kids to learn code...
Programming language is now human.”
— Jensen Huang, CEO of NVIDIA*

Why Learn Programming Today?

- Builds **logical & algorithmic thinking**
- Enables **automation** and problem-solving
- Improves your ability to **use and direct AI tools**
- A transferable skill in multiple domains

"Learning programming
so I can work in IT"

"Learning programming
so I can create small
scripts"

"Learning programming
so I can understand
programming memes"



Our Role in the AI Era

We are no longer just **coders**, we are:

- **Problem Designers** — define problems clearly
- **AI Orchestrators** — control and guide AI tools
- **Validators** — review and test AI outputs
- **Integrators** — connect and deploy solutions

Artists and Designers
when AI takes over
their jobs

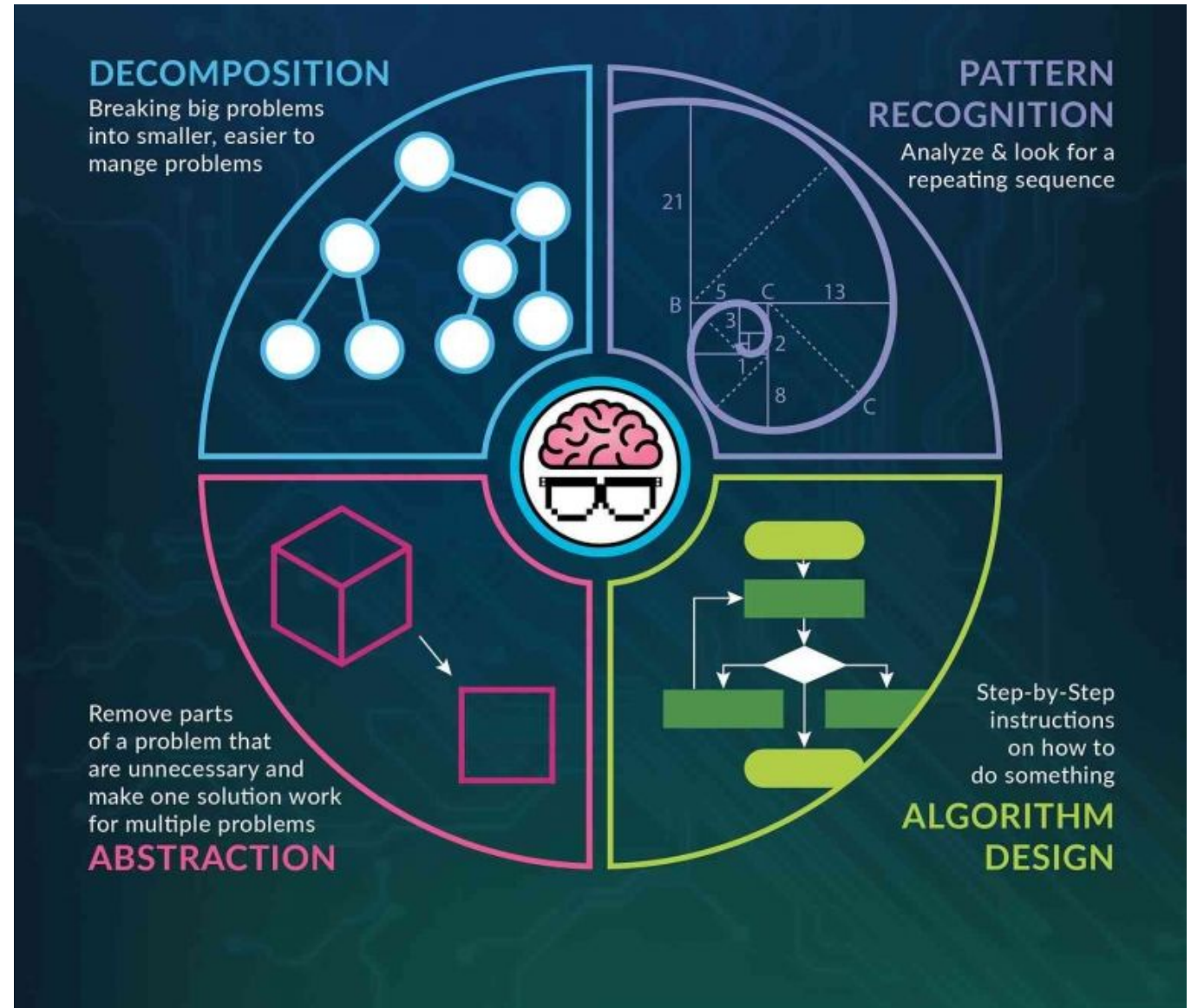


Programmers
when AI takes
over their jobs



Computational Thinking (CT)

- A problem-solving approach that uses systematic, step-by-step methods based on computation.
- **Note:** It's not only for computer-related problems.



Case: Car Breakdown with Low Fuel

Problem Definition

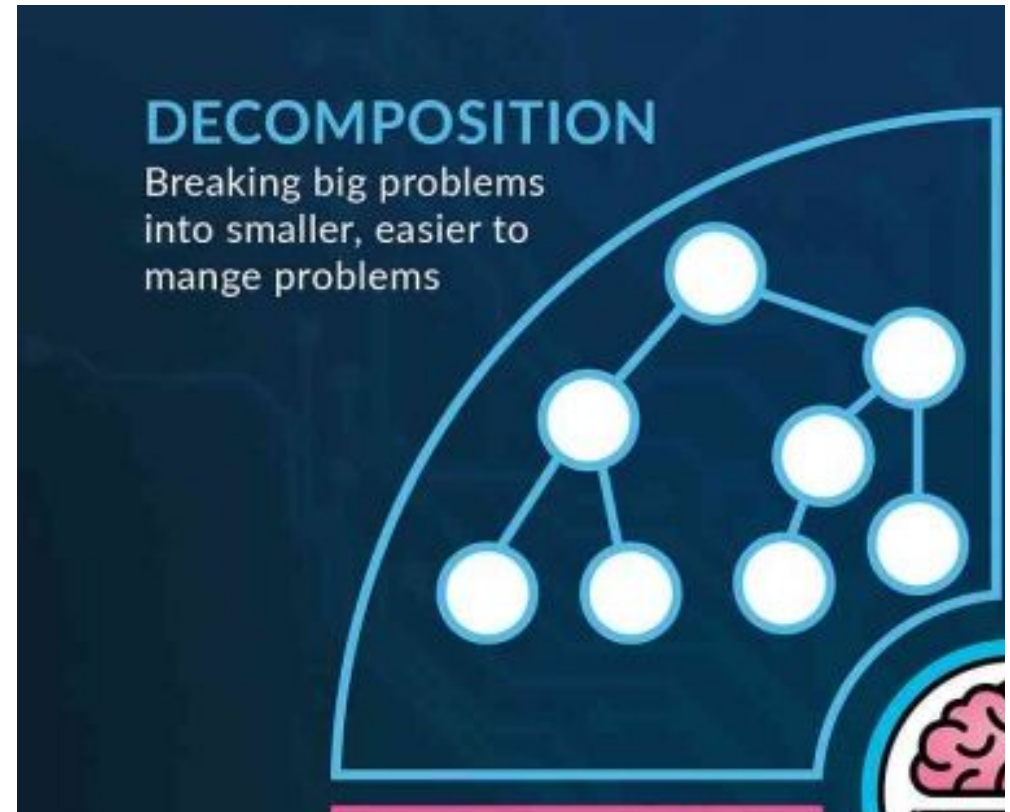
- **Problem:** Car stopped in the middle of the road, fuel almost empty.
- **Goal:** Get the car running again or find an alternative safe solution.
- **Constraints:** Limited time, roadside condition, tools in the car.



Step 1: Decomposition

Break the problem into smaller parts:

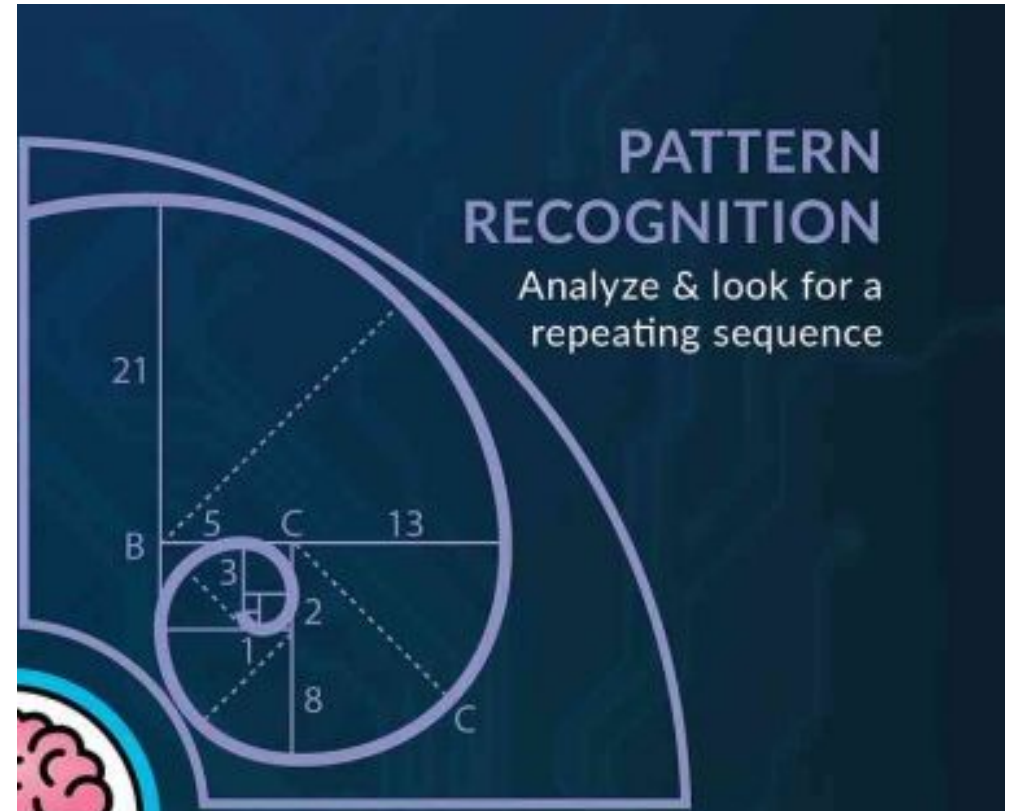
- Secure the car (hazard lights, safety triangle).
- Check fuel indicator.
- Check battery status.
- Decide which to fix first.
- Test and confirm.



Step 2: Pattern Recognition

Identify common causes:

- Empty fuel tank
- Dead battery
- Loose battery cables
- Overheating



Step 3: Abstraction

Focus on important details, ignore the rest:

- Relevant: Fuel level, battery status, spare battery availability.
- Irrelevant: Car color, music, decorations.



Step 4: Algorithm Design

Create a step-by-step plan:

1. Secure the car.
2. Check fuel and battery.
3. If fuel > 0 and spare battery available
→ Replace battery.
4. Try starting the engine.
 - a. If successful → Drive to nearest gas station.
 - b. If not → Seek assistance.

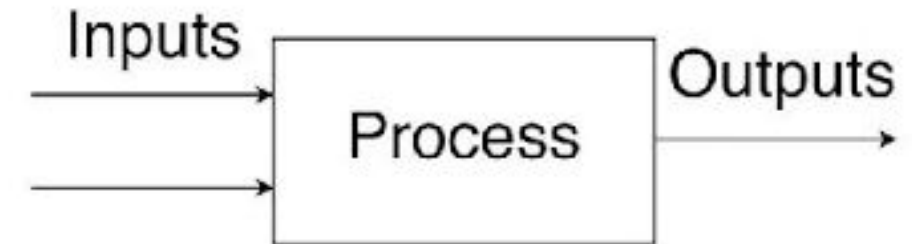


IPO Diagram (Input – Process – Output)

A tool to clearly define what goes **into**, what happens **inside**, and what comes **out** of a program.

Structure:

- **Input:** What data do we need?
- **Process:** What steps or calculations will we perform?
- **Output:** What results will we produce?



Example – Car Breakdown Problem






- **Input:** Fuel level, battery status, spare battery availability
- **Process:** Check conditions → Replace battery if possible → Attempt to start engine
- **Output:** Car running again or request for assistance

Definition: A simplified, plain-language description of the steps in an algorithm, not tied to any programming language.

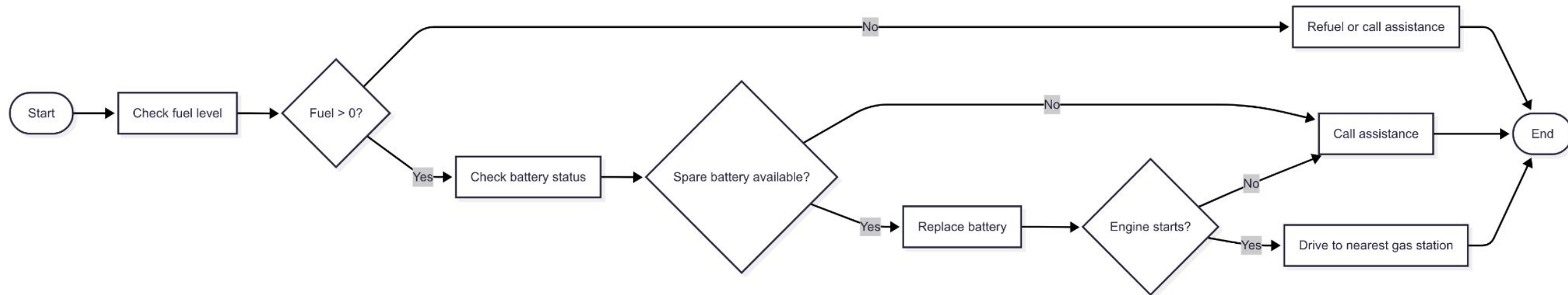
```
IF fuel > 0 AND spare battery available
    Replace battery
    IF engine starts
        Drive to gas station
    ELSE
        Call assistance
ELSE
    Refuel or call assistance
```

Algorithm Design: Flowchart (1)

Definition: A diagram that visually represents the steps and decision points in an algorithm using symbols.

Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision

Algorithm Design: Flowchart (2)





UNIVERSITAS
GADJAH MADA

Any Question?



UNIVERSITAS
GADJAH MADA

Thank You!
See you, next week,
stay safe!

