2023

Algorithmic Thinking and Flowchart

Logic Algorithm

Tri Hadiah Muliawati

Politeknik Elektronika Negeri Surabaya
Department of Informatics and Computer Engineering,

Objectives

1. Students able to understand The fundamental of Algorithmic Thinking
2. Students able to create flowchart to visualize the algorithm

Outline

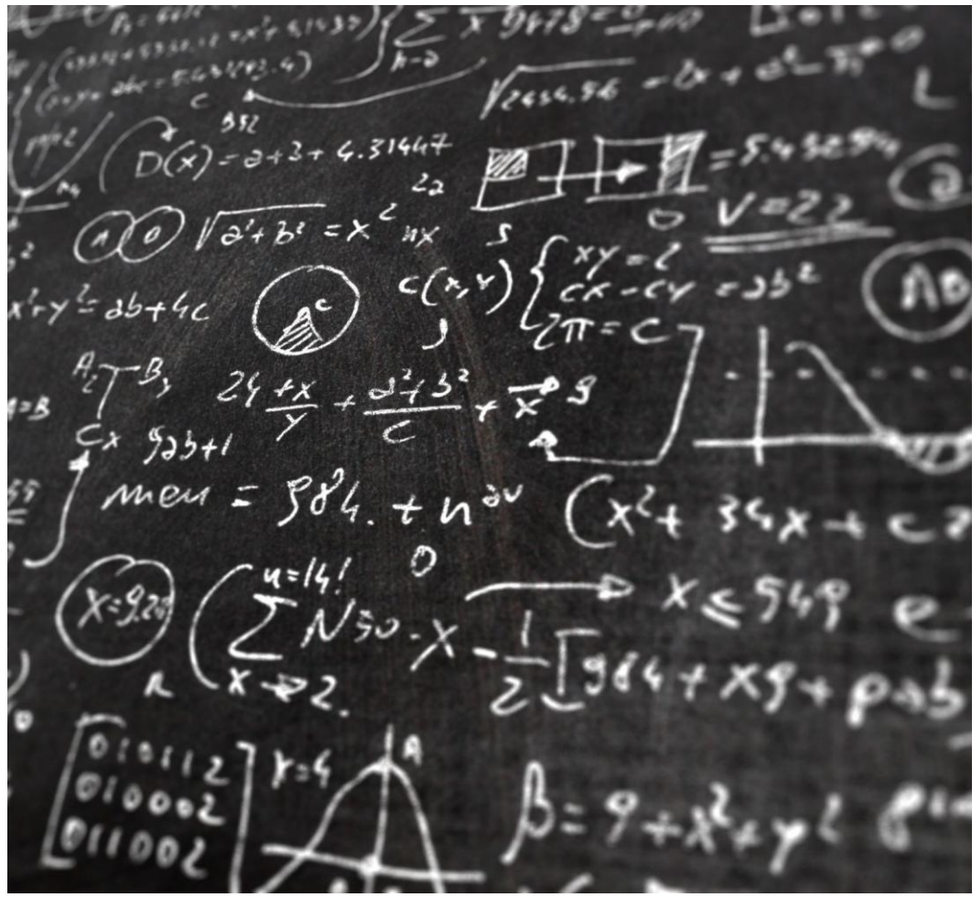
1. Fundamental Algorithmic Thinking
2. Flowchart

Fundamental Algorithmic Thinking

Politeknik Elektronika Negeri Surabaya
Department of Informatics and Computer Engineering,

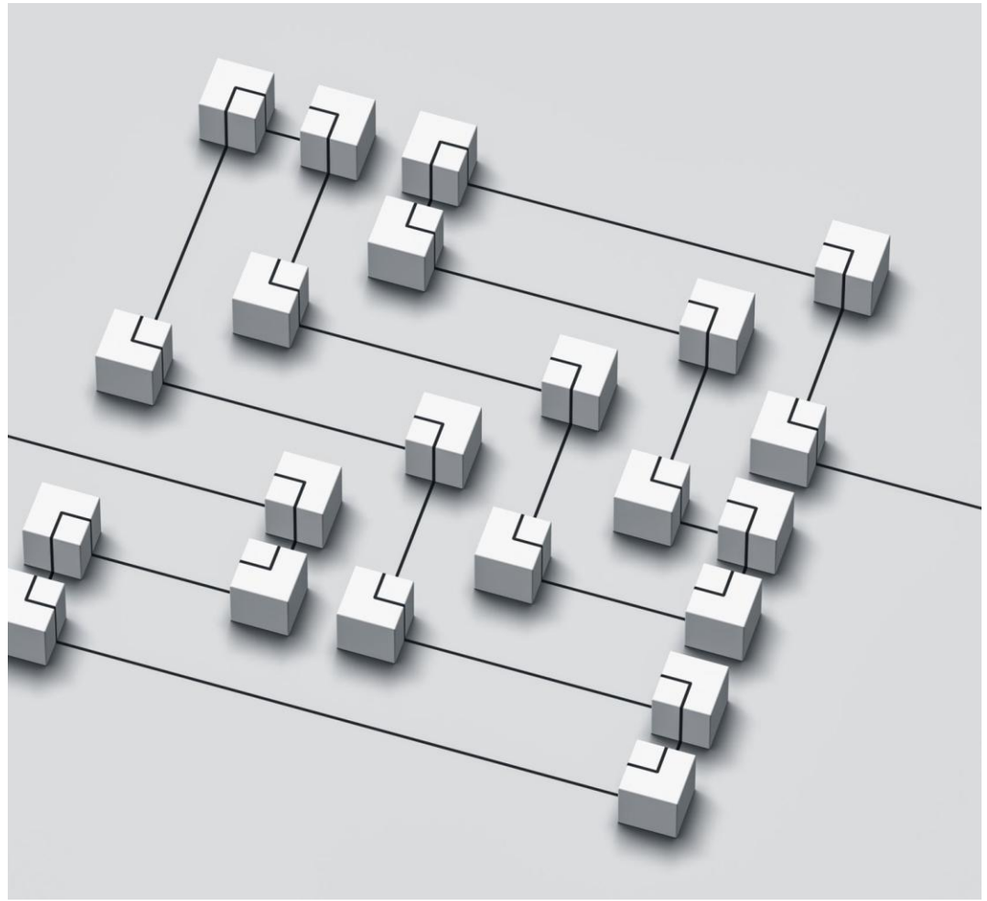
What is Algorithmic Thinking?

- Let's define Algorithm first: "An Algorithm is a **method to solve a problem** that **consists of exactly defined instructions**"[1]
- Algorithmic thinking is a competency that allows us to **understand and construct algorithms to solve a problem**.



What is Algorithmic Thinking?

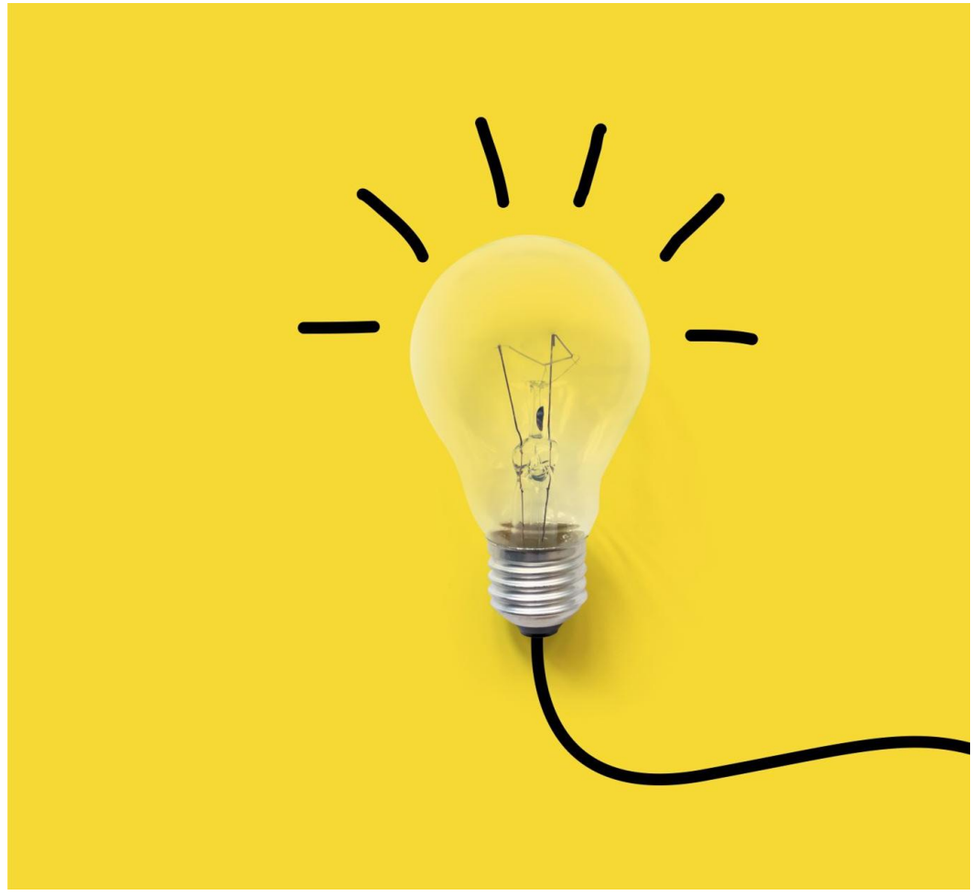
- Algorithmic thinking is a **key skill in the field of computing** that can be developed independently of any programming language.
- Algorithmic thinking **is not limited to the field of computing** and can be applied in other disciplines and daily life.



Algorithmic Thinking in Computing

But if we want to build functioning systems based on rules, then **logic alone isn't sufficient**. We need something that can i**ntegrate all these rules and execute actions** based on the outcomes of evaluating them.

That something is algorithms, and they are the power behind all real-world computational systems.

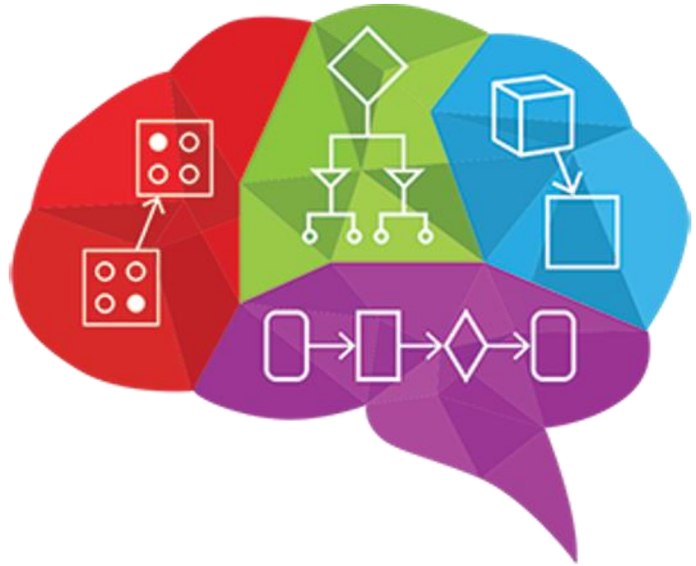


Facts Algorithmic Thinking in Computing

- **A computer will do exactly as it is told**. If it is told to do something impossible, it will crash.
- **A computer has no innate intelligence of its own**. It will not do anything that it has not been instructed to do.
- **A computer has no common sense**. It will not try to interpret your instructions in different ways. It will not make assumptions or fill in the obvious blanks in an incomplete algorithm.

What Actually is Algorithmic Thinking?

Algorithm is a sequence of clearly defined steps that describe a process to follow a finite set of unambiguous instructions with clear start and end points.



Algorithmic Thinking Includes

Being able to precisely define the problem

The ability to analyze a given problem

Identify the fundamental steps required to solve a given problem

Thinking about all scenarios and cases of the given problem

Defining a set of detailed instructions to solve the problem (designing the algorithm)

Examining the developed algorithm and improving its efficiency

Importance of Algorithmic Thinking

There is an algorithm behind any computer program.

Algorithmic thinking is the core skill required for programming.

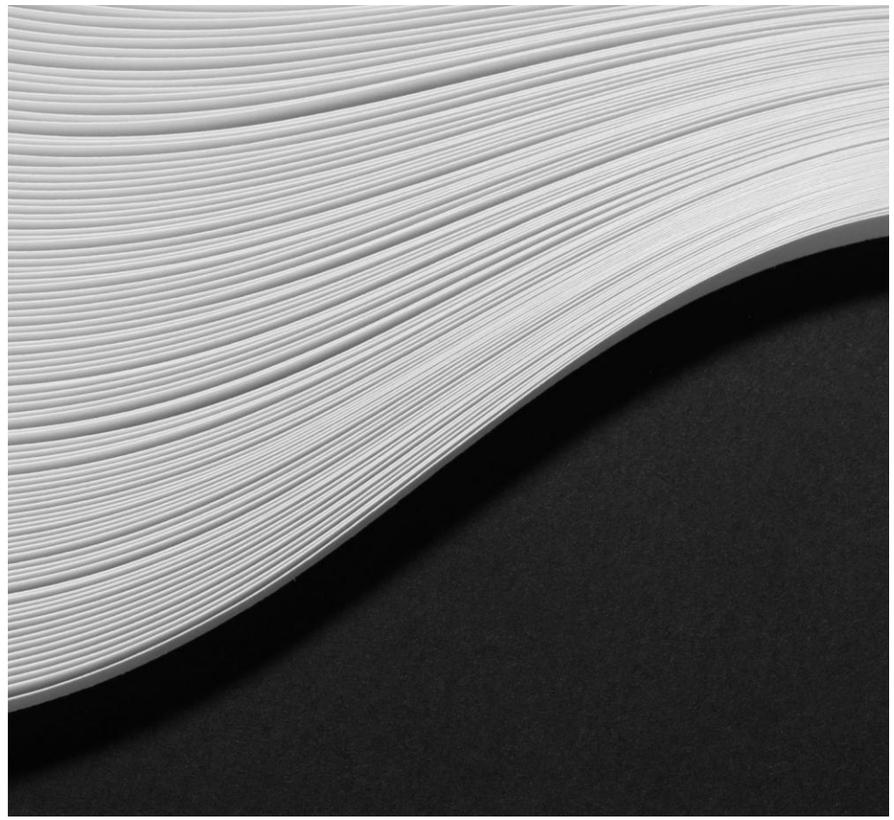
To be able to develop robust programs we need to invest time in developing efficient algorithms before coding.

We need to train our brains to understand the algorithmic logic and improve its algorithmic thinking skill to be able to intuitively develop algorithms.

Algorithmic Thinking Example

For sorting students' papers alphabetically based on their last name, the basic algorithm would be like this:

- Skip the first name
- Take the first character of the last name
- Put it in the list alphabetically
- If another paper exists with the same first letter of the last name, then compare the first letter of the first name and sort accordingly.
- Continue until no paper is left unsorted.



Exercise: Algorithmic Thinking

Brush the teeth

Cross the road

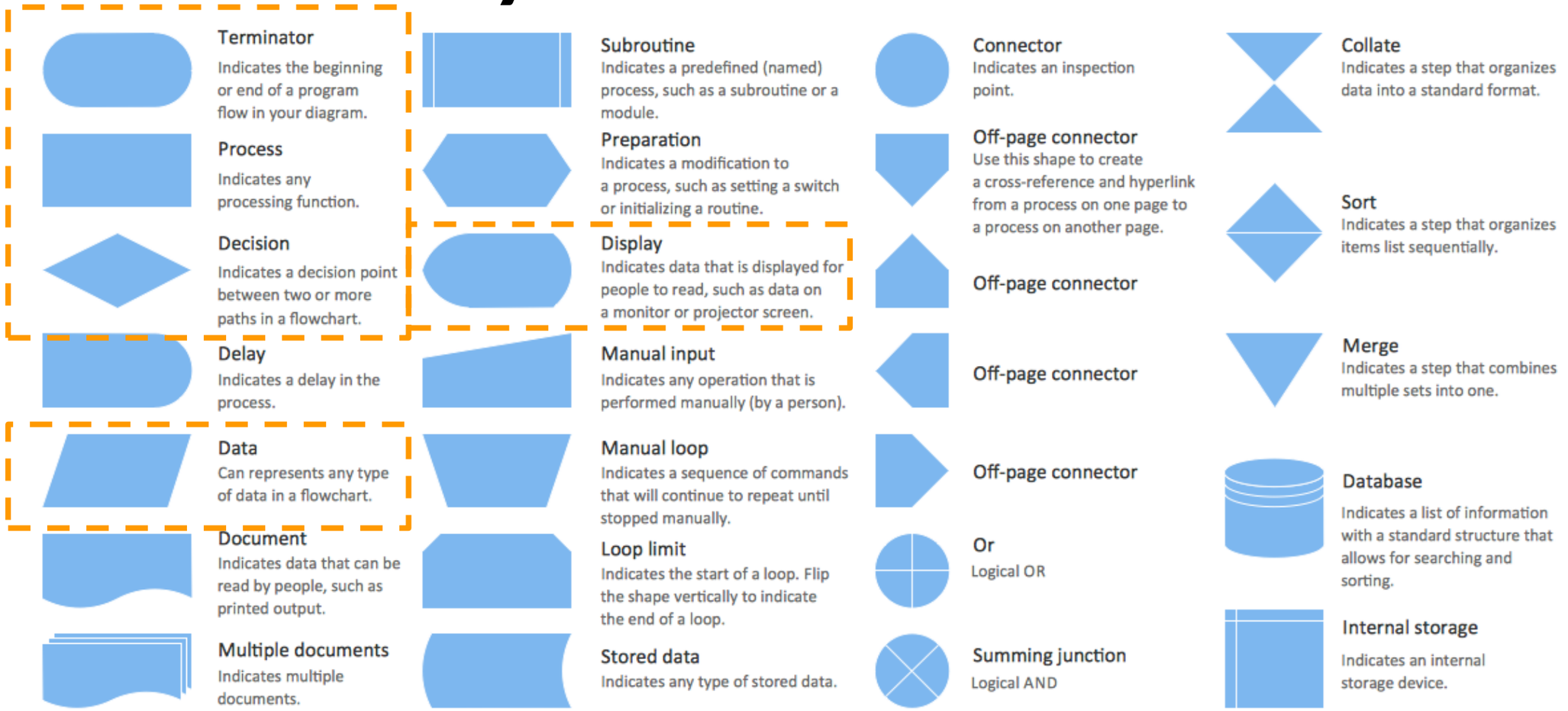
Make a lemon juice

Flowchart

Flowcharts

- Flowcharts can be seen as graphical representations of algorithms.
- Flowcharts are more intuitive and make it easy to understand the algorithms. (depending on the complexity of the problem however, sometimes algorithms may better represent the steps to solve the problem)
- We use standard symbols and shapes in a Flowchart to show the sequential steps of an algorithm.

Flowcharts Symbols



Terminator
Indicates the beginning or end of a program flow in your diagram.

Process
Indicates any processing function.

Decision
Indicates a decision point between two or more paths in a flowchart.

Delay
Indicates a delay in the process.

Data
Can represents any type of data in a flowchart.

Document
Indicates data that can be read by people, such as printed output.

Multiple documents
Indicates multiple documents.

Subroutine
Indicates a predefined (named) process, such as a subroutine or a module.

Preparation
Indicates a modification to a process, such as setting a switch or initializing a routine.

Display
Indicates data that is displayed for people to read, such as data on a monitor or projector screen.

Manual input
Indicates any operation that is performed manually (by a person).

Manual loop
Indicates a sequence of commands that will continue to repeat until stopped manually.

Loop limit
Indicates the start of a loop. Flip the shape vertically to indicate the end of a loop.

Stored data
Indicates any type of stored data.

Connector
Indicates an inspection point.

Off-page connector
Use this shape to create a cross-reference and hyperlink from a process on one page to a process on another page.

Off-page connector

Off-page connector

Off-page connector

Or
Logical OR

Summing junction
Logical AND

Collate
Indicates a step that organizes data into a standard format.

Sort
Indicates a step that organizes items list sequentially.

Merge
Indicates a step that combines multiple sets into one.

Database
Indicates a list of information with a standard structure that allows for searching and sorting.

Internal storage
Indicates an internal storage device.

Let’s Try : Sample Algorithm

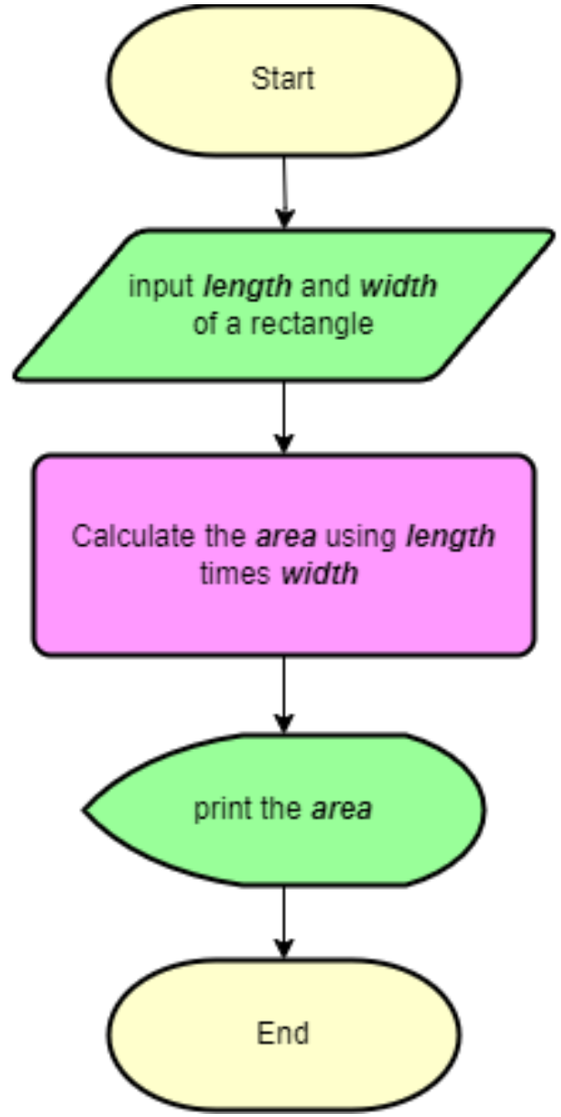
Write a code to calculate the area of a rectangle.

- Input the length of a rectangle
- Input the width of a rectangle
- Calculate the area using the length times the width
- Print the area

Let’s Try : Flowchart

Write a code to calculate the area of a rectangle.

- Input the length of a rectangle
- Input the width of a rectangle
- Calculate the area using the length times the width
- Print the area



Let’s Try : Sample Algorithm

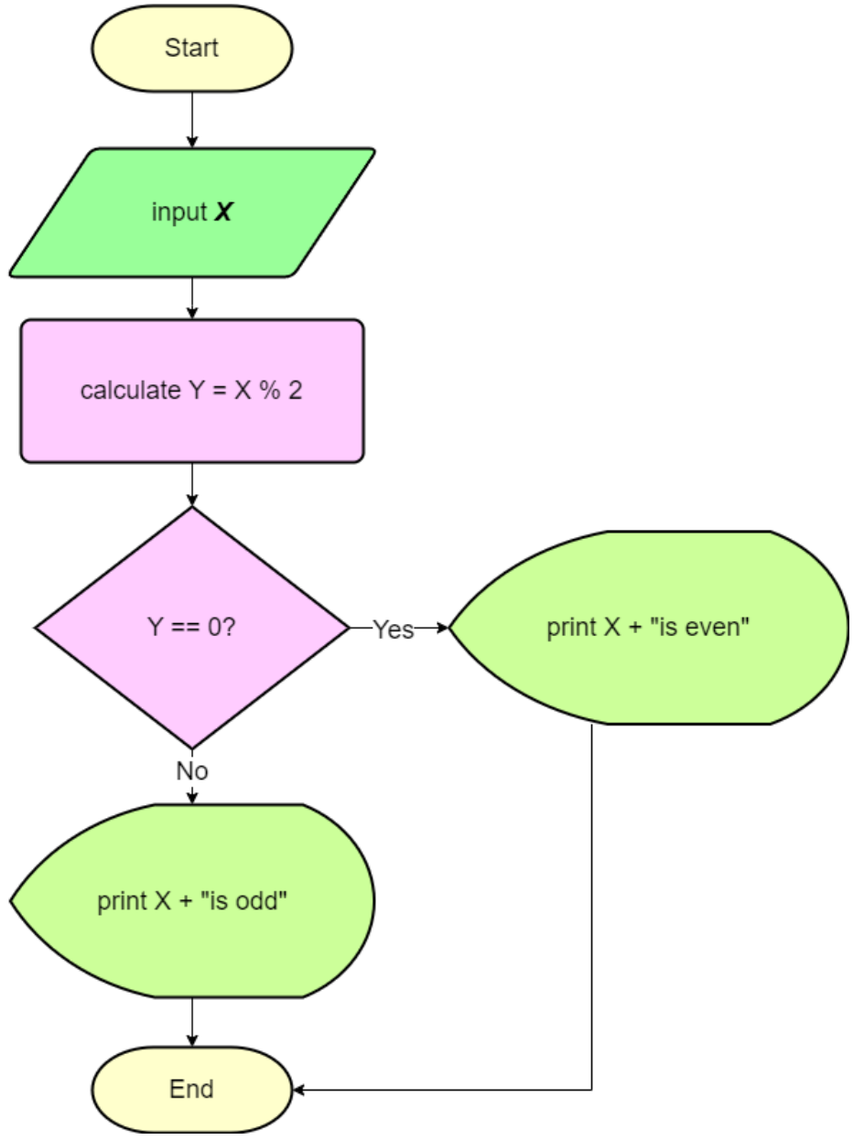
Write a code to determine if an input integer number is odd or even.

- Set variables X with a number.
- Divide X by 2 and store the remainder in variable Y.
- If Y equals to 0, print X + “is even.”
- Else, If Y equals 1, print X + “is odd”.

Let’s Try : Flowchart

Write a code to determine if an input integer number is odd or even.

- Set variables X with a number.
- Divide X by 2 and store the remainder in variable Y.
- If Y equals to 0, print X + “is even.”
- Else, If Y equals 1, print X + “is odd”.



Exercise: Flowchart

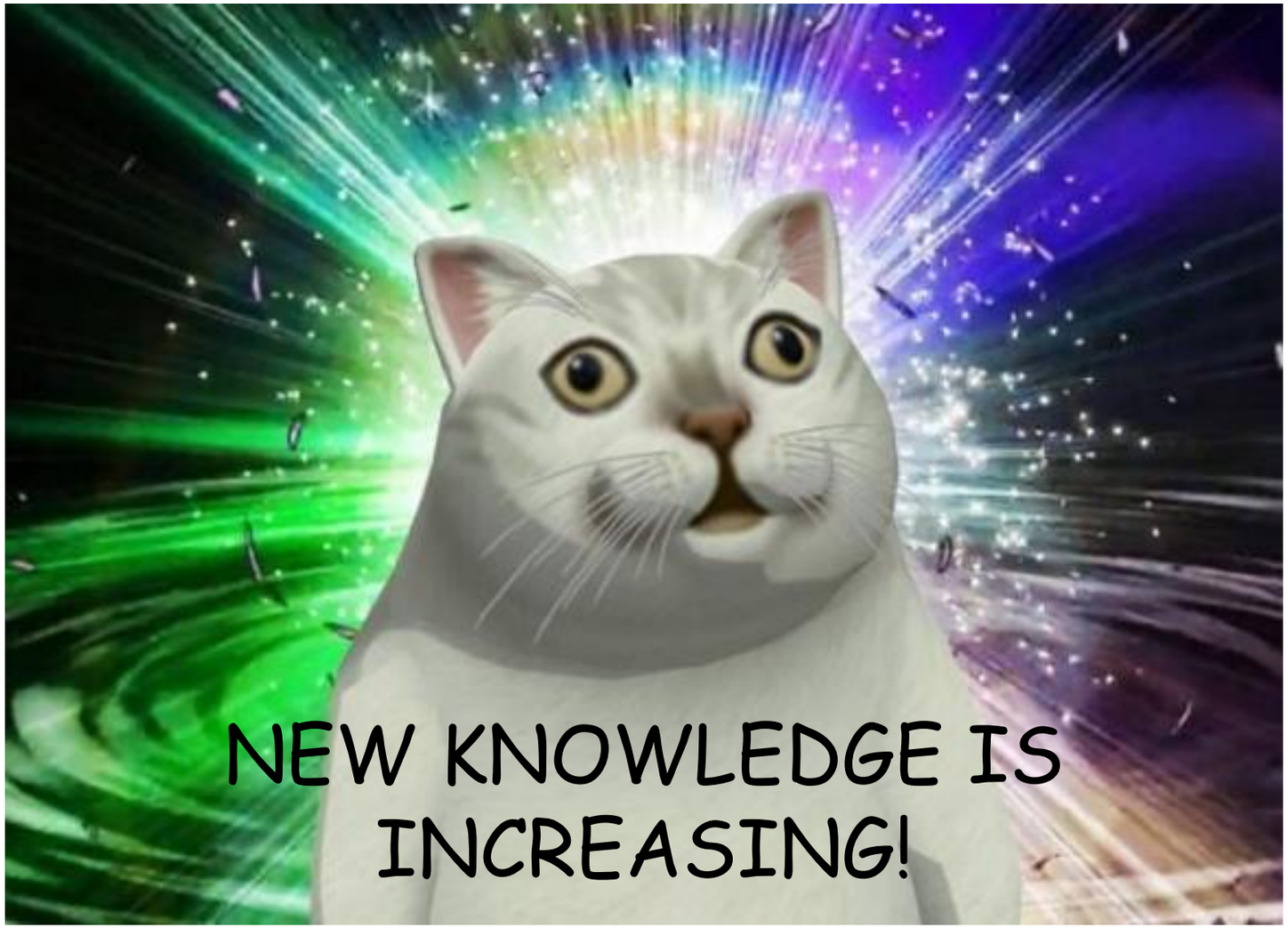
Brush the teeth

Cross the road

Make a lemon juice

Summary

- Algorithm is a sequence of clearly defined steps that describe a process to follow a finite set of unambiguous instructions with clear start and end points.
- Flowcharts can be seen as graphical representations of algorithms.

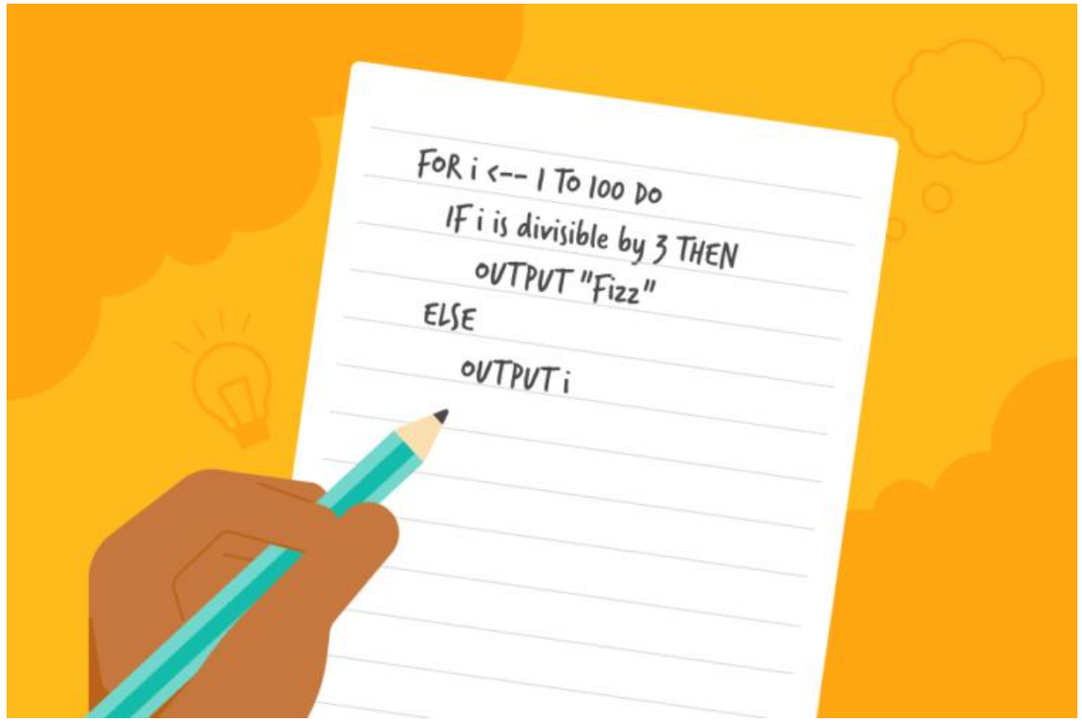

NEW KNOWLEDGE IS
INCREASING!

Referensi

Karl, Beecher. "Computational Thinking: A Beginner's Guide to Problem-Solving and Programming." Swindon, UK: BCS, The Chartered Institute for IT (2017).

Next week

- We will learn about **Pseudocode**. Please read references about it.
- **It is strongly encouraged to do self study and self-paced practicum.**



THANK YOU!