2023

Pseudocode

Logic Algorithm

Tri Hadiah Muliawati

Politeknik Elektronika Negeri Surabaya
Department of Informatics and Computer Engineering,

Objectives:

1. Students able to understand the importance of pseudocode as an assisting tool to turn algorithms into code
2. Students able to turn algorithm and flowchart into pseudocode

Outline:

1. Fundamental Pseudocode
2. Exercise

Fundamental Pseudocode

Politeknik Elektronika Negeri Surabaya
Department of Informatics and Computer Engineering,

Let’s Throwback to Algorithm

Algorithm is a **way of specifying a multi-step task**, and is especially useful when we wish **to explain to a third party (be it human or machine) how to carry out steps with extreme precision**.

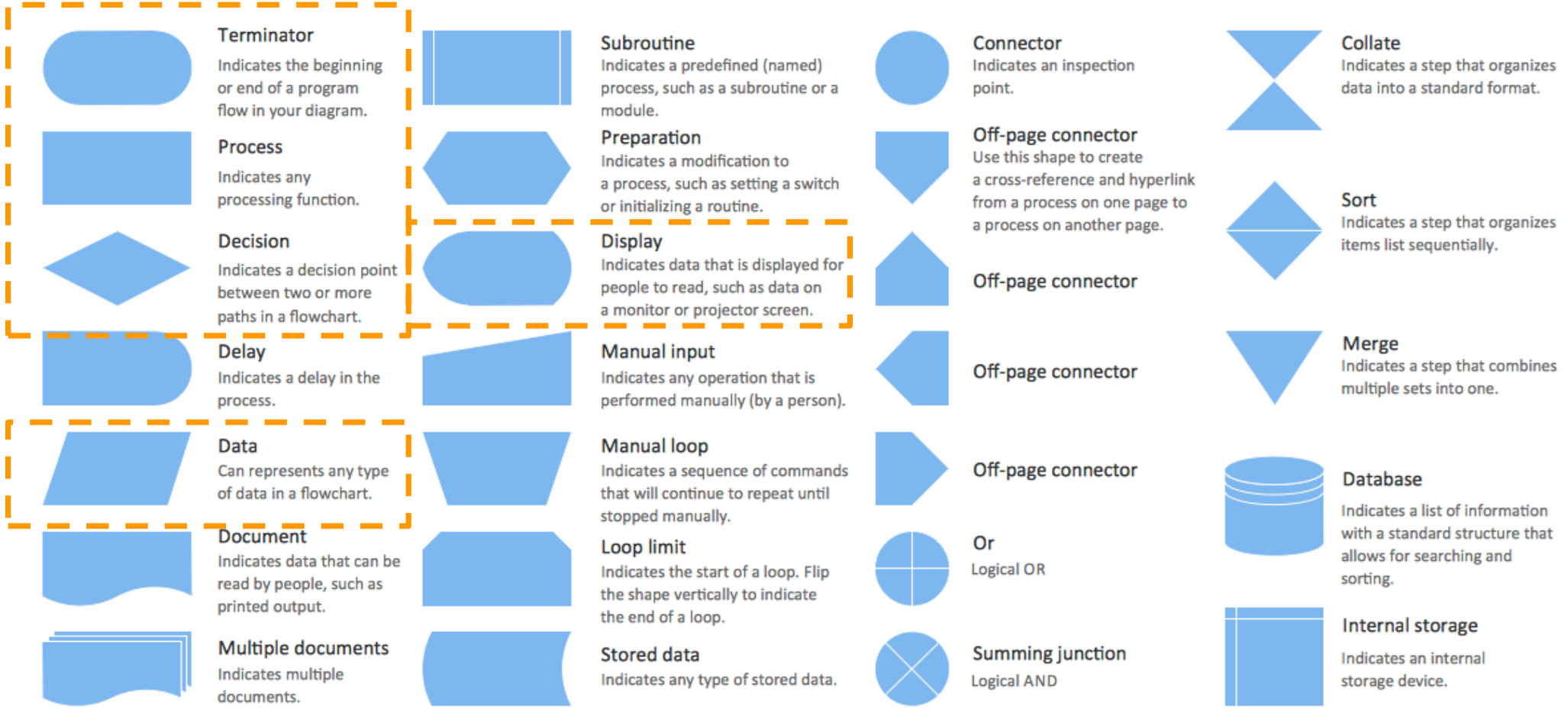
The properties of an algorithm are:

1. Finiteness
2. Definiteness
3. Input
4. Output
5. Effectiveness

Let’s Throwback to Flowchart

- Flowchart can be seen as **graphical representations** of algorithms.
- Flowchart is **more intuitive and make it easy to understand** the algorithm (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.

Let’s Throwback to Flowchart



Algorithm vs Flowchart

Objective: Calculate the area of a rectangle

Algorithm:

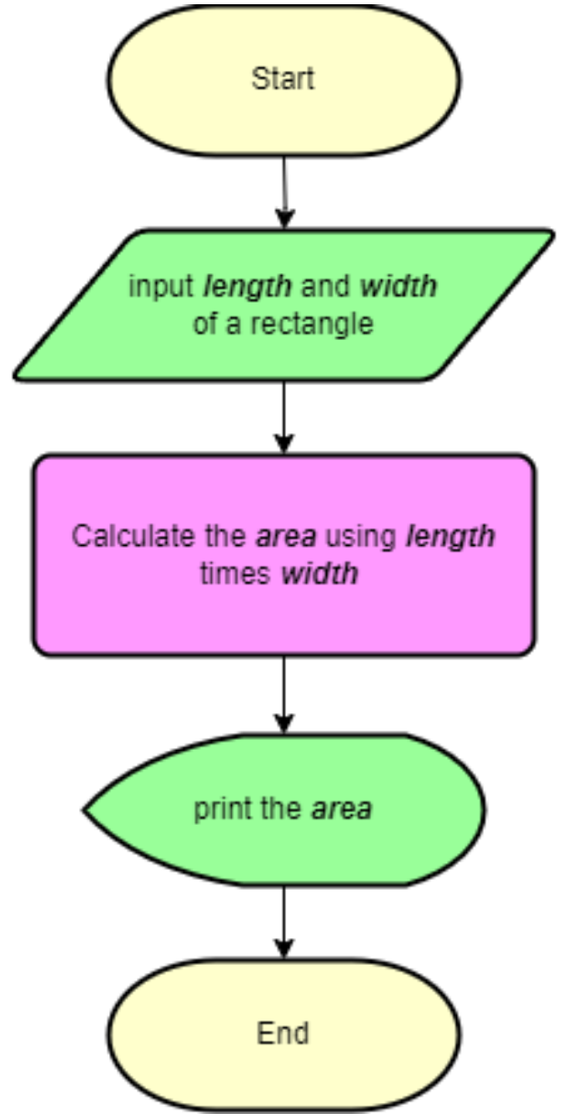
- Input the length of a rectangle
- Input the width of a rectangle
- Calculate the area using the length times the width
- Print the area

Algorithm vs Flowchart

Objective: Calculate the area of a rectangle

Algorithm:

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



Algorithm vs Flowchart

Objective: Check whether a student passes “Pemrograman 1” subject or not. He / She will pass if the final score is more than or equal to 56.

Algorithm:

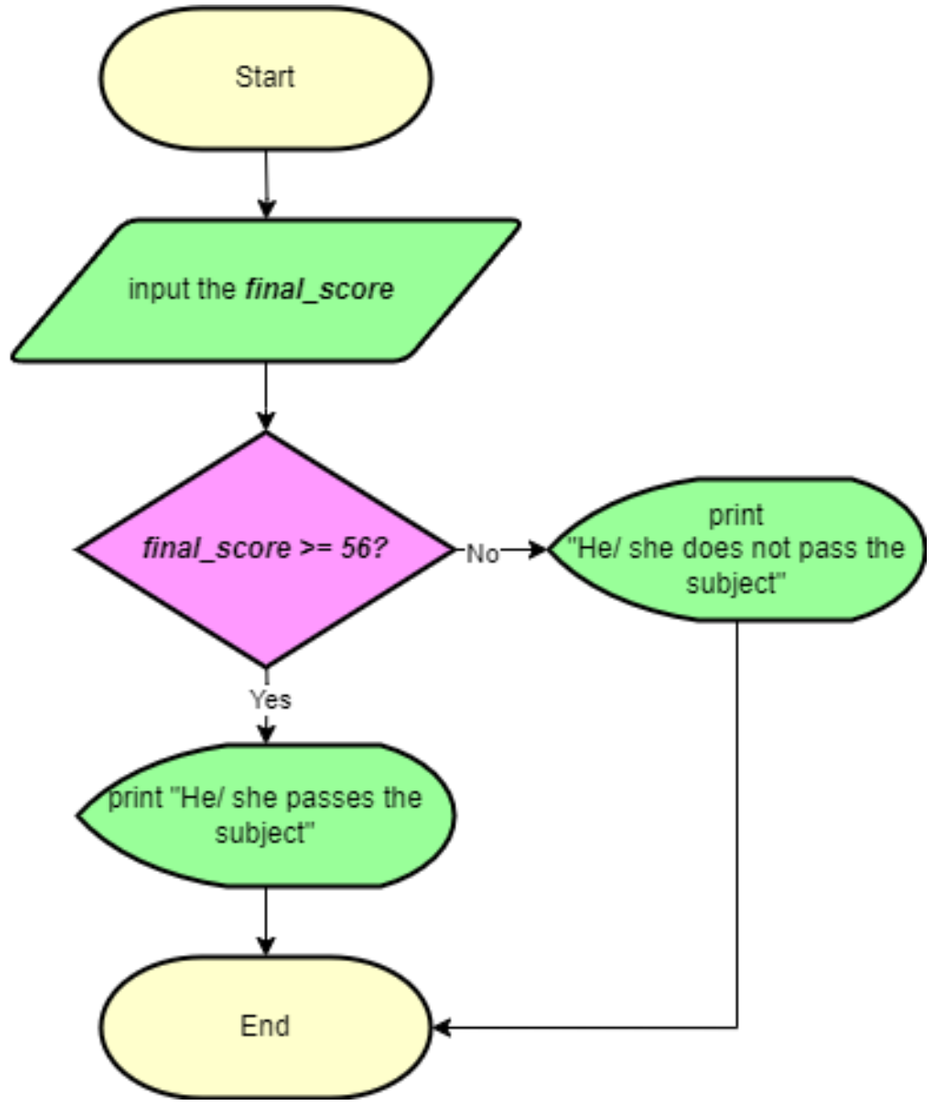
- Input the final score
- Check whether the final score is more than or equal to 56
- If yes, print he/she passes the subject
- If no, print he/she does not pass the subject

Algorithm vs Flowchart

Objective: Check whether a student passes “Pemrograman 1” subject or not. He / She will pass if the final score is more than or equal to 56.

Algorithm:

- Input the final score
- Check whether the final score is more than or equal to 56
- If yes, print he/she passes the subject
- If no, print he/she does not pass the subject



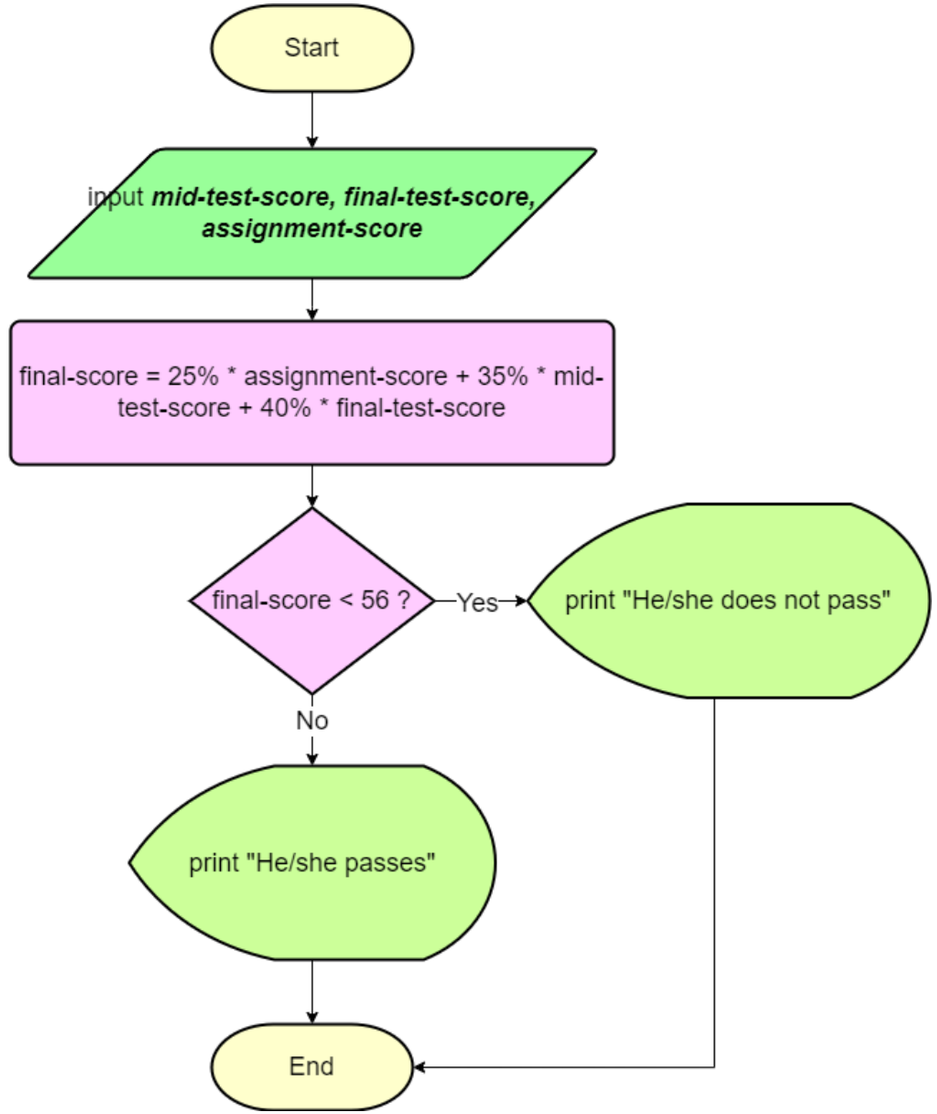
Exercise

- Write an algorithm to check whether a student passes “Pemrograman 1” subject. He / She will pass if the final score is more than or equal to 56. Final score is calculated using 25% of assignment score, 35% of mid-test score, and 40% of final-test score.
- Draw the flowchart

Answer

Algorithm:

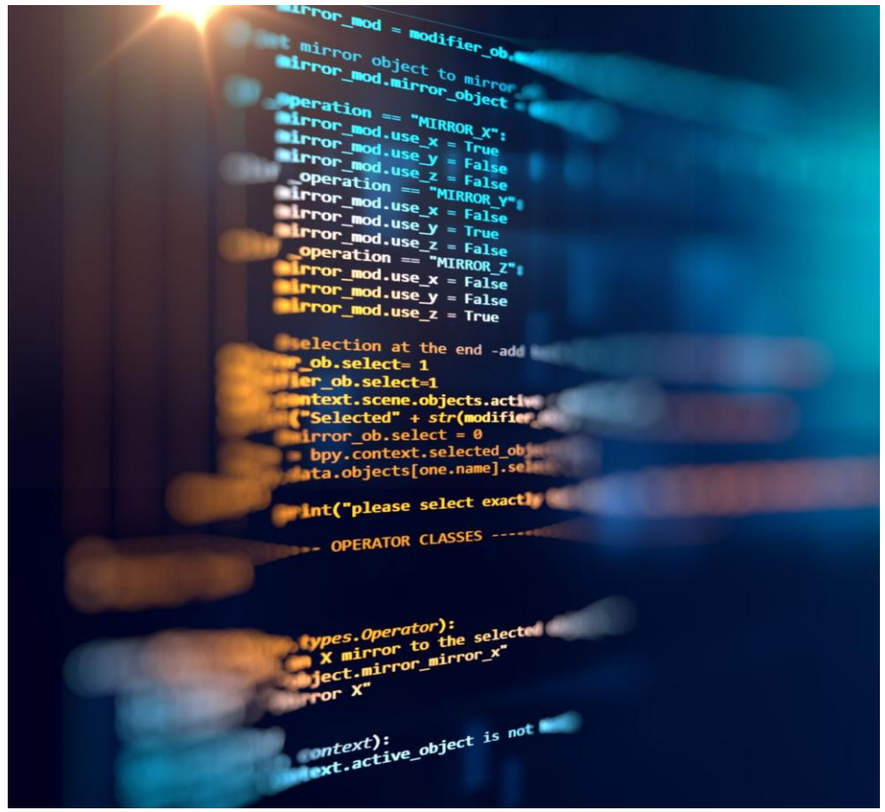
- Input the assignment, mid-test, and final-test score
- Calculate the final score using 25% of assignment score + 35% of mid-test score + 40% of final-test score
- Check whether the final score is more than or equal to 56
- If yes, print “He/she passes the subject”
- If no, print “He/she does not pass the subject”



Flowchart can help to visualize the algorithm, thus we can understand it better. However, it is **not easy to turn a flowchart into a practical code**.
Therefore, we can use **pseudocode** as an assisting tool **to turn algorithm and/or flowchart to code**.

What is Pseudocode?

- Pseudocode is a fake code that is an explanation of how to **solve a problem**
- Pseudocode uses a language that **almost** like programming language
- Apart from that, pseudocode usually uses language that is **easy to understand** and also **more concise** than algorithms
- In pseudocode, there is **no standard syntax or grammar rules**
- Therefore, we can apply this pseudocode in various programming languages

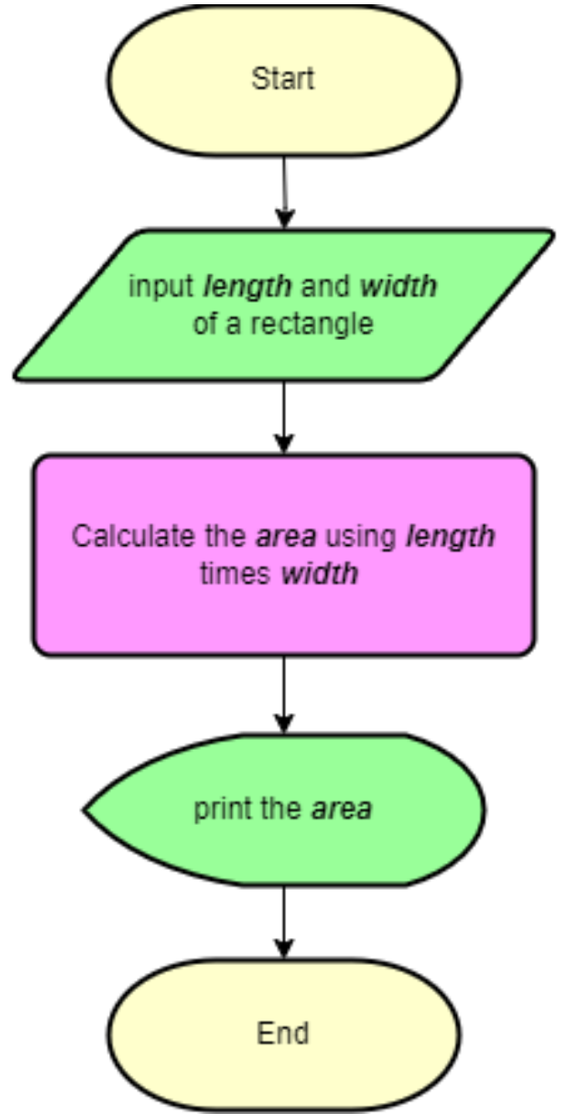


Flowchart vs Pseudocode

Objective: Calculate the area of a rectangle:

Pseudocode:

- INPUT *length*
- INPUT *width*
- *area* ← *length * width*
- PRINT *area*



Algorithm vs Pseudocode

Objective: Calculate the area of a rectangle

Algorithm:

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

Pseudocode:

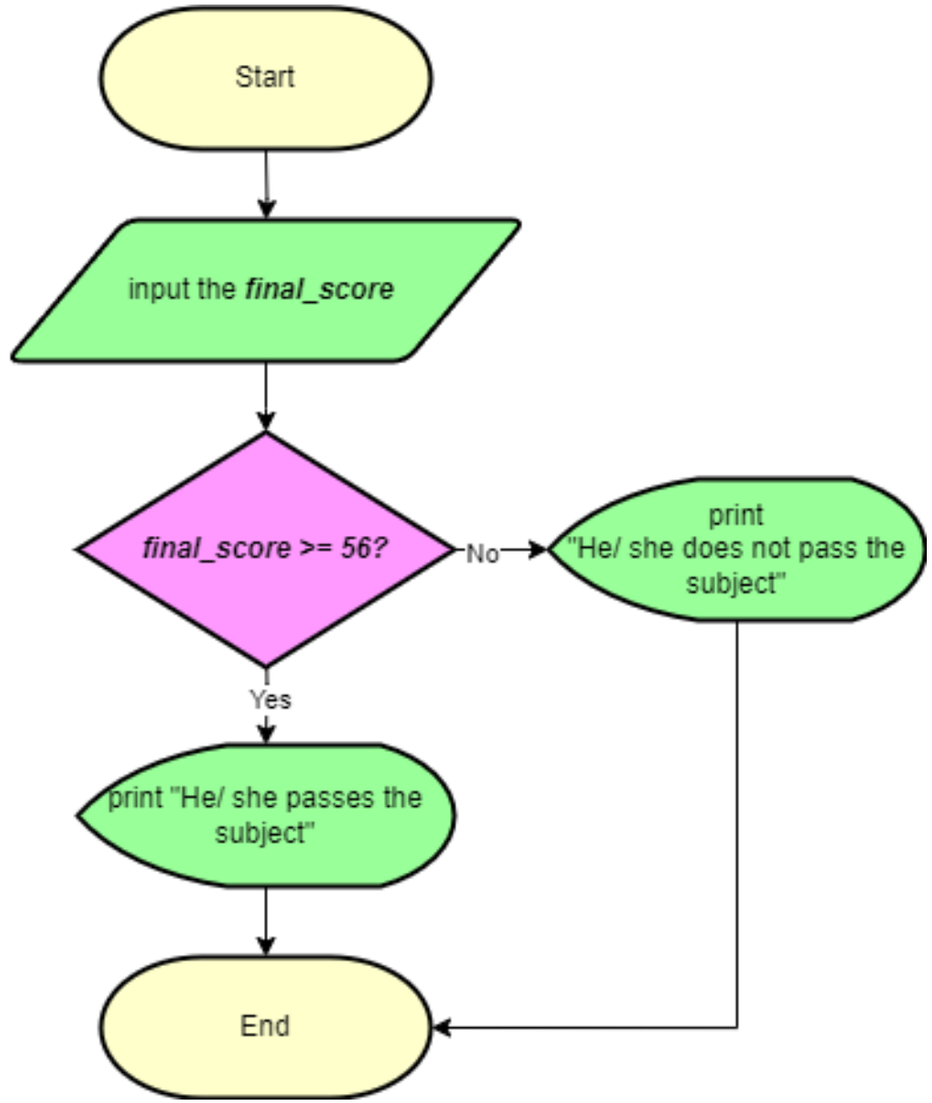
- INPUT *length*
- INPUT *width*
- *area* ← *length * width*
- PRINT *area*

Algorithm vs Flowchart

Objective: Check whether a student pass “Pemrograman 1” subject or not. He / She will pass if the final score is more than or equal to 56.

Pseudocode:

- INPUT the *final_score*
- IF *final_score* >= 56 THEN
- PRINT “he/she passes the subject”
- ELSE
- PRINT “he/she does not pass the subject”
- ENDIF



Algorithm vs Pseudocode

Objective: Check whether a student pass “Pemrograman 1” subject or not. He / She will pass if the final score is more than or equal to 56.

Algorithm:

- Input the final score
- Check whether the final score is more than or equal to 56
- If yes, print he/she passes the subject
- If no, print he/she does not pass the subject

Pseudocode:

- INPUT the *final_score*
- IF *final_score* >= 56 THEN
- PRINT “he/she passes the subject”
- ELSE
- PRINT “he/she does not pass the subject”
- ENDIF

Pseudocode Functionality

Variables

Names of places to store values

quotient, decimalNumber, newBase

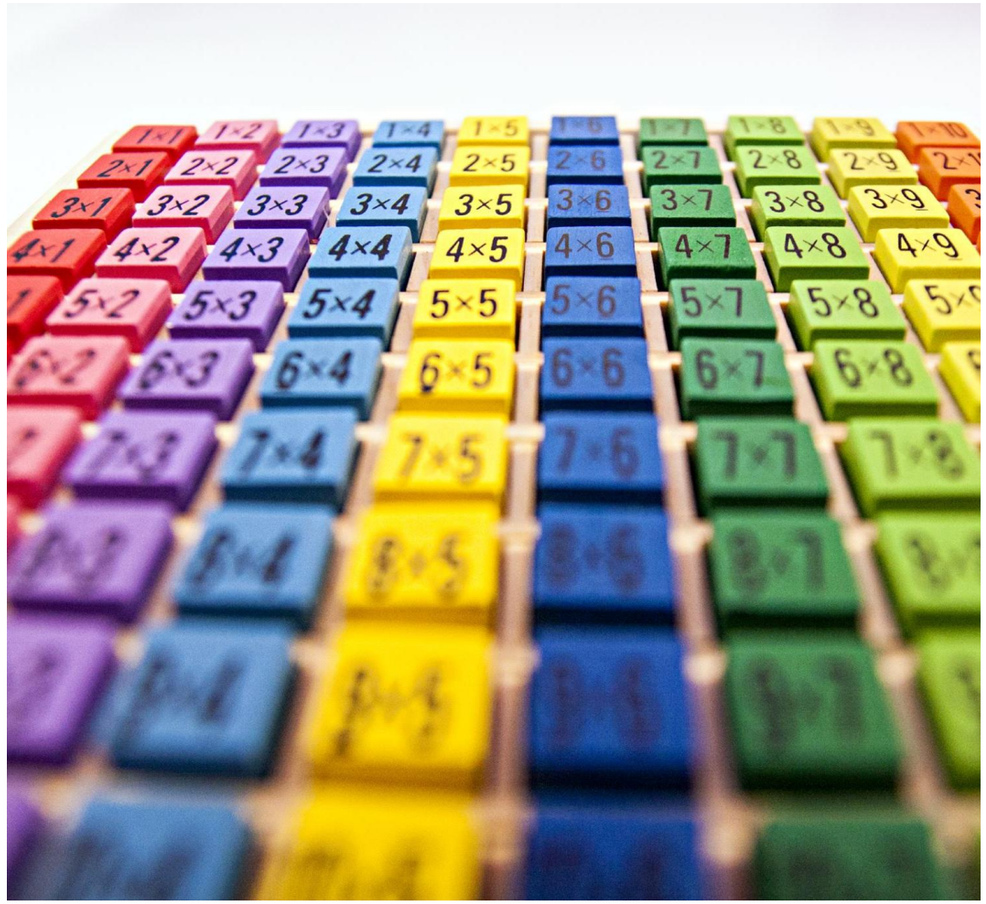
Assignment

Storing the value of an expression into a variable

quotient ← 64

quotient ← 6 * 10 + 4

SET *quotient* to 189



Pseudocode Functionality

Output

Printing a value on an output device

WRITE, PRINT

Input

Getting values from the outside word and storing them into variables

GET, READ, INPUT

Pseudocode Functionality

Sequence

A sequence consists of one or more instructions.

Each instruction is executed sequentially according to the order in which it was written.

Example: Exchange of two integers

1. DECLARE *A*, *B*, and *C* as integers
2. INPUT the values of *A* and *B*
3. $C \leftarrow A$
4. $A \leftarrow B$
5. $B \leftarrow C$

1	A	B	C
2	10	5	-
3	-	5	10
4	5	-	10
5	5	10	-

Pseudocode Functionality

Selection / Decision

Making a choice to execute or skip a statement (or group of statements)

Example: Check if a certain number is a negative number or not

1. READ *number*
2. IF *number* < 0 THEN
3. WRITE *number* + " is a negative number."
4. ELSE
5. WRITE *number* + " is more than or equal to zero."
6. ENDIF

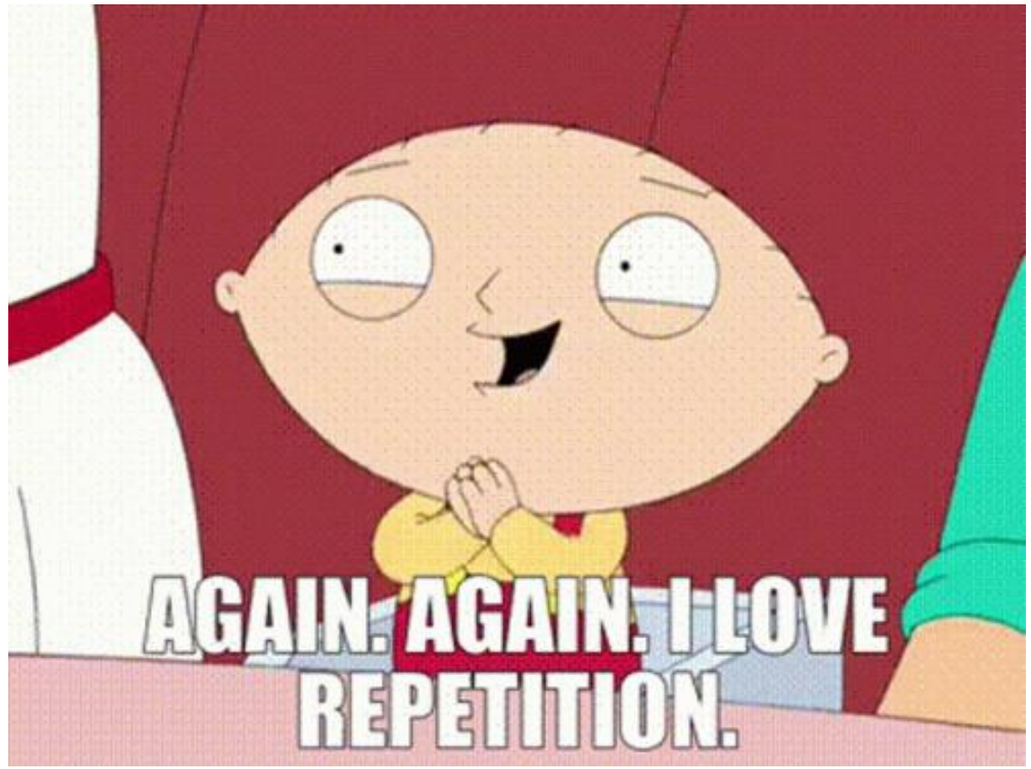
Pseudocode Functionality

Repetition

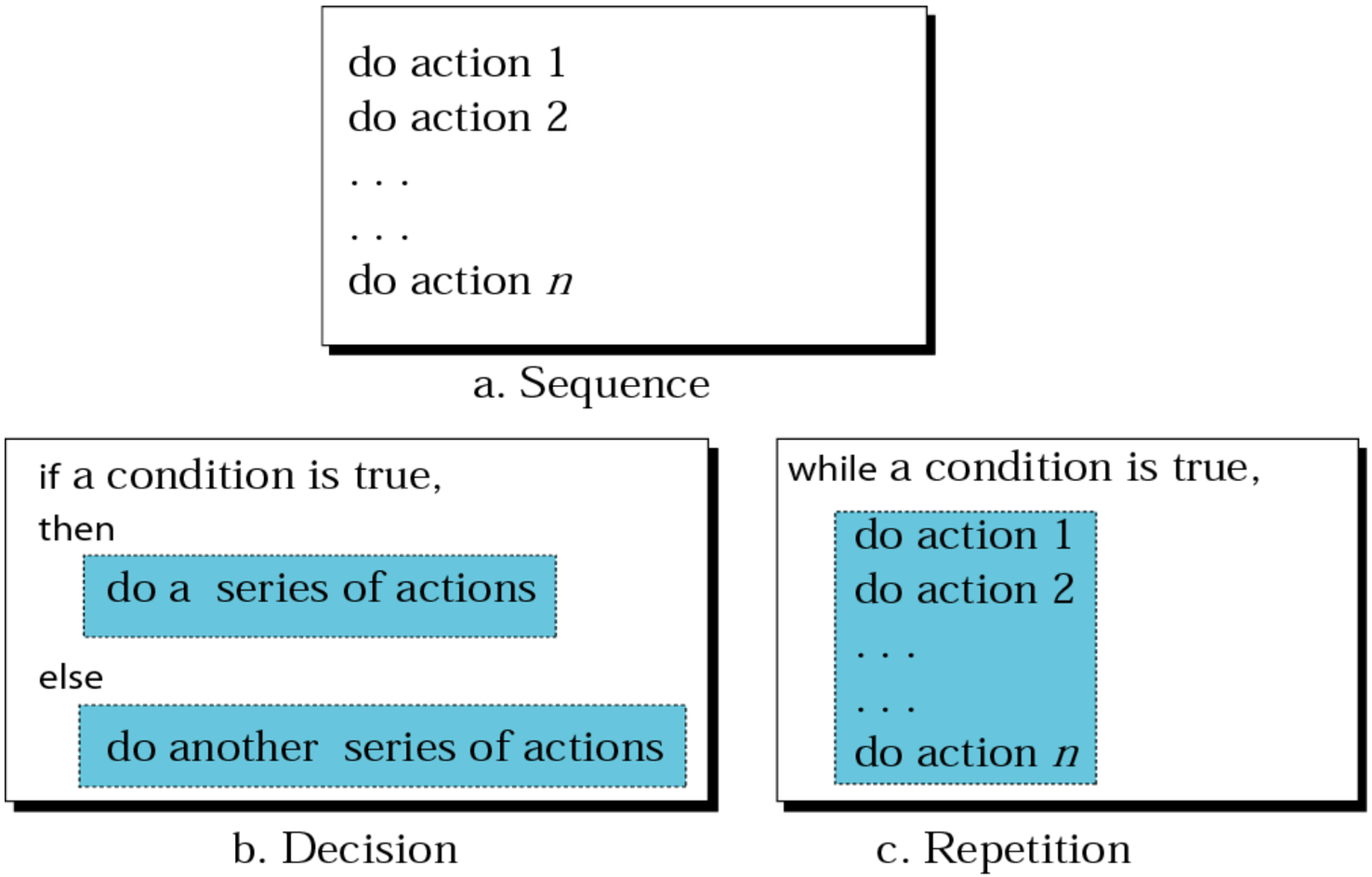
Repeating a series of statements

Example: Input and print 10 integer number sequentially

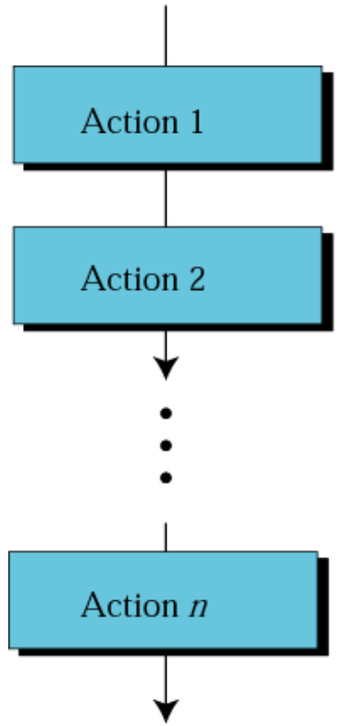
1. SET *count* to 1
2. WHILE (*count* < 10)
3. WRITE "Enter an integer number"
4. READ *aNumber*
5. WRITE "You entered " + *aNumber*
6. *count* ← *count* + 1



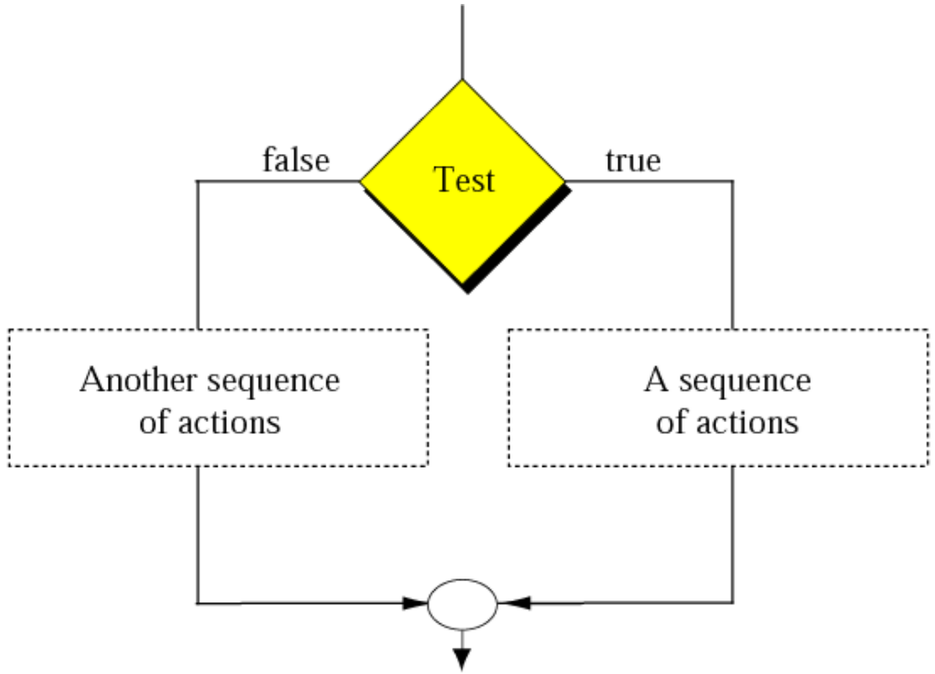
Three Constructs



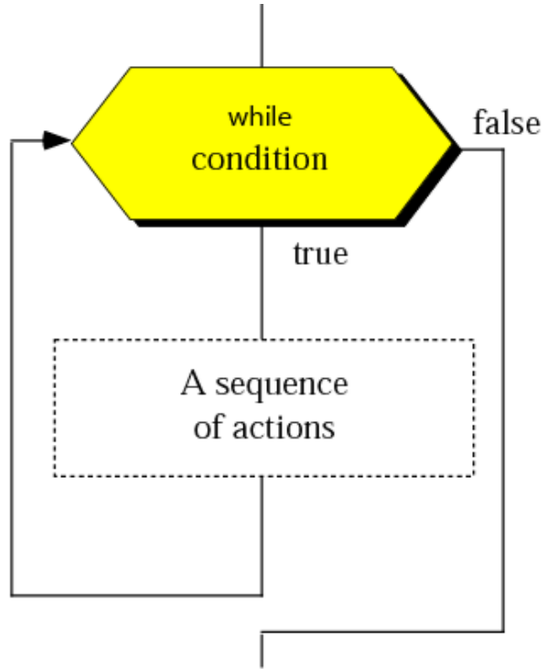
Three Constructs (Flowchart)



a. Sequence



b. Decision



c. Repetition

Three Constructs (Pseudocode)

```
action 1
action 2
   .
   .
   .
action n
```

a. Sequence

```
if (condition)
    then
        action
        action
        . . .
    else
        action
        action
        . . .
End if
```

b. Decision

```
while (condition)
        action
        action
        . . .
End while
```

c. Repetition

Which one to Choose between Pseudocode or Flowchart?

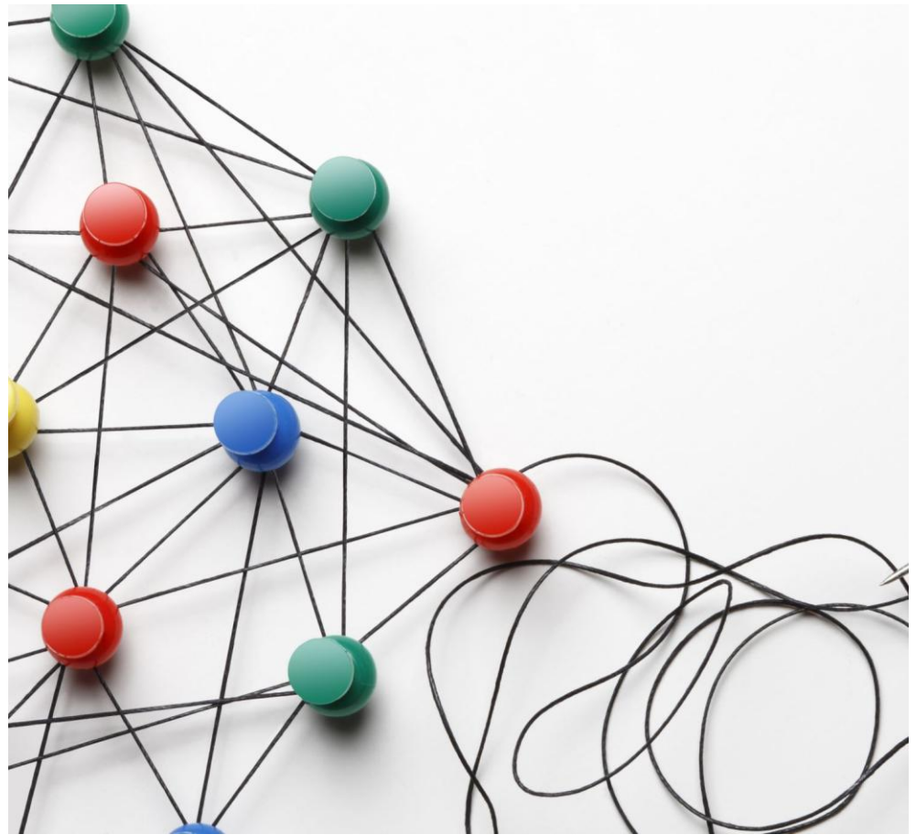
Flowchart

Symbols convey different types of actions

Pseudocode

A cross between code and plain English

Use the one that you think is easier to assist you when constructing code



Summary

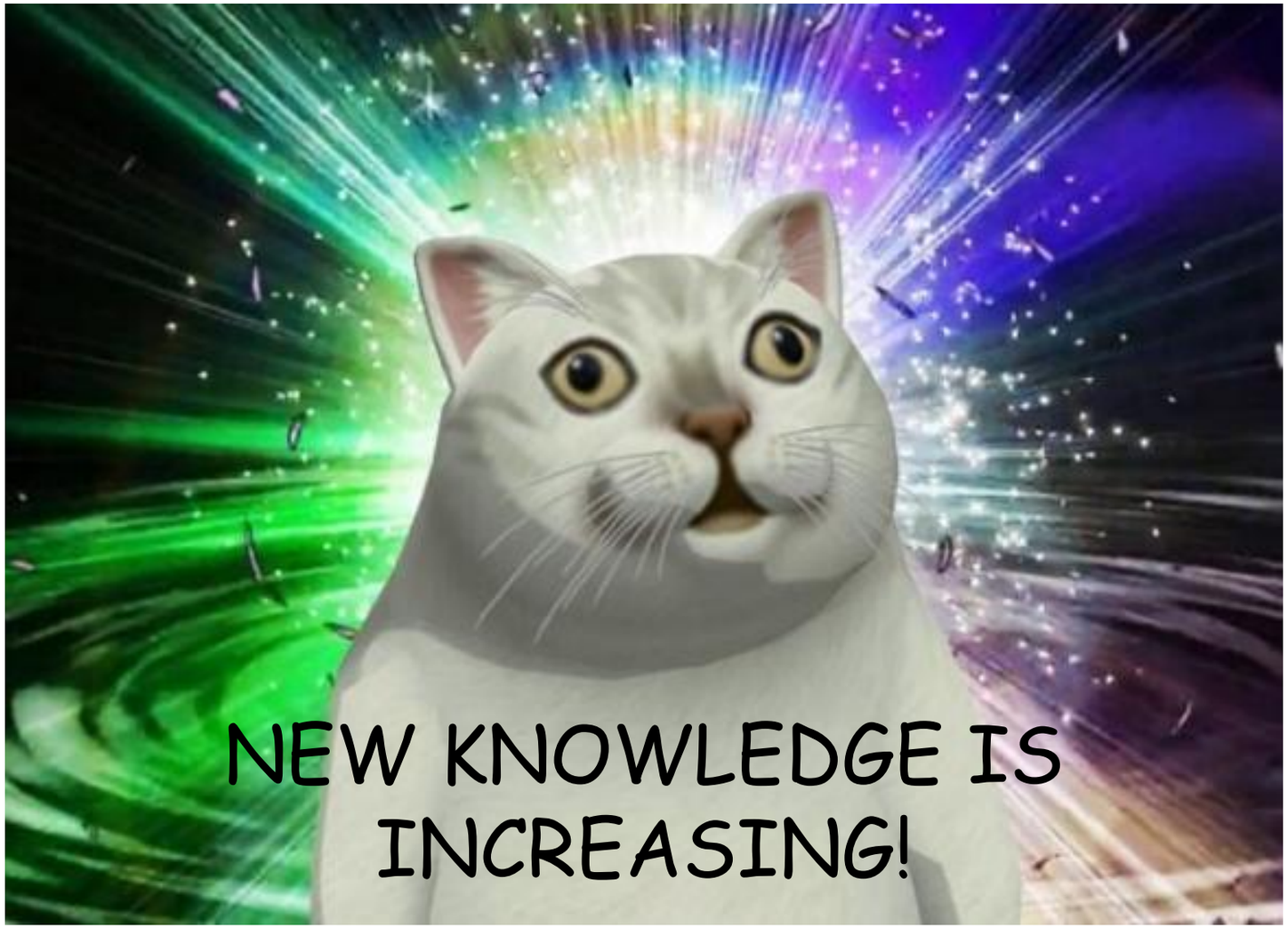
- Pseudocode is a cross between code and plain English that is an explanation of how to solve a problem
- Pseudocode uses a language that almost like programming language.
- There are Three Constructs : Sequence, Decision, and Repetition

Task

An automatic washing machine has three wash modes: Fast, Normal, and Intensive. The user selects the mode based on the level of dirtiness of the clothes. Each mode has a different wash duration:

- Fast: 15 minutes
- Normal: 30 minutes
- Intensive: 45 minutes

Create pseudocode that accepts user input for the selected mode and displays the appropriate wash duration.


NEW KNOWLEDGE IS
INCREASING!

Reference

1. Karl, Beecher. "Computational Thinking: A Beginner's Guide to Problem-Solving and Programming." Swindon, UK: BCS, The Chartered Institute for IT (2017).
2. Dale, Nell B., and John Lewis. Computer science illuminated. Jones & Bartlett Learning, 2007.

Next week

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

THANK YOU!