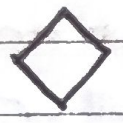
#9_ Flowcharts

17/05/2025

- Flowcharts help visually map out logic to solve a problem.

- **Flowchart Components**




→ START / STOP

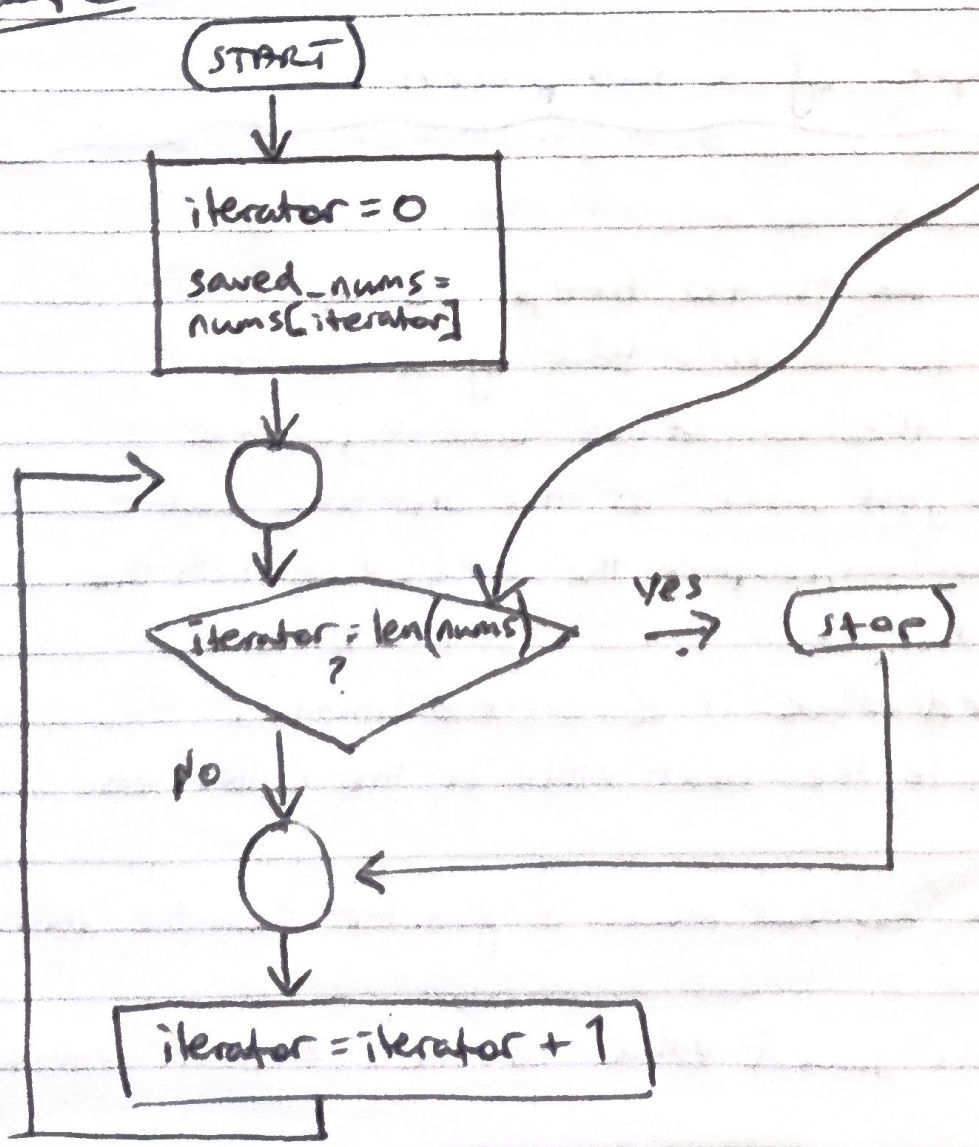
⇒ Processing Step

⇒ Input / Output

⇒ Decision

⇒ Connector

example



note:

this component should only have two branches, anymore more branches, use separate ◇ for each branch.

17/05/2025

- looping manually rather than using specific constructs allows you to better understand the logic and to think like a computer.

- thinking like a computer helps you debug logical errors

A LARGER PROBLEM

- Sub-processes are processes within a process that require much logic
  -> while the user wants to keep going:
    - ask the user for a collection of numbers
    - extract the largest one from that collection
    - ask the user if they want to input another collection.
  -> return the saved list of numbers

BRIEF OVERVIEW

this is an example of a sub-process.

FULL PSEUDOCODE:

-> while the user wants to keep going:
  - ask the user for a collection of numbers
  - iterate through the collection one-by-one
    - save the first value as the starting value
    - for each iteration, compare the saved value with the current value:
    - if the current value is ≤ saved value
      - move to the next value in the collection

* otherwise, if the current value is greater than the saved value
  - reassign the saved value as the current value

17/05/2025

- after iterating through the collection, save the largest value into the list.
- ask the user if they want to input another collection

→ return the saved list of numbers.

- Difficult to trust accuracy of long pseudocode. This is why it is important to extract a logical grouping into a sub-process to tackle each piece separately

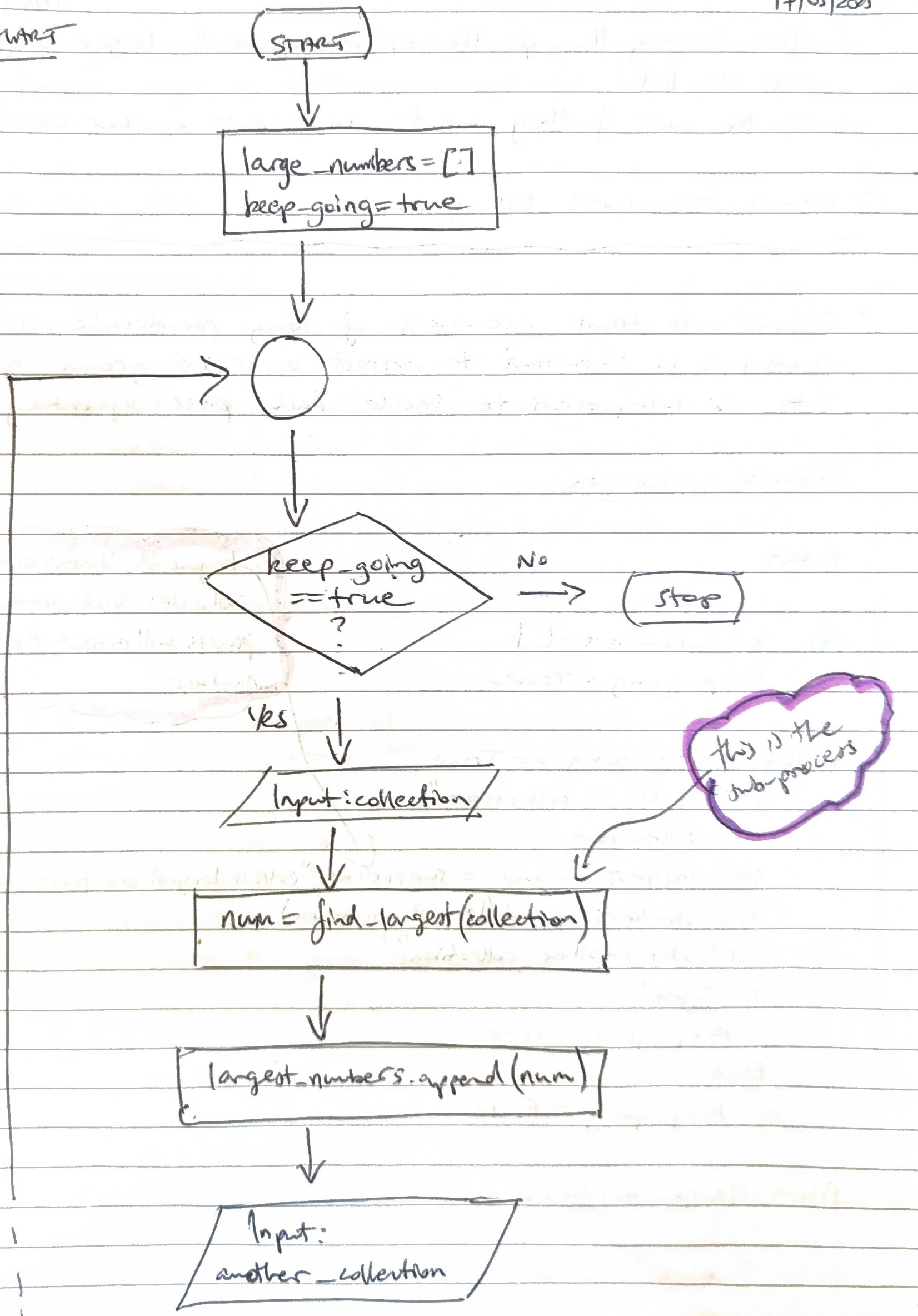
FORMAL PSEUDOCODE

```
START

SET large_numbers = [ ]
SET keep_going = True

WHILE keep_going == True
    GET "enter a collection"
    SET collection
    SET largest_number = SUBPROCESS "extract largest one from collection"
    large_numbers.append(largest_number)
    GET "enter another collection?"
    IF "yes"
      keep_going = True
    ELSE
      keep_going = False

PRINT large_numbers
```

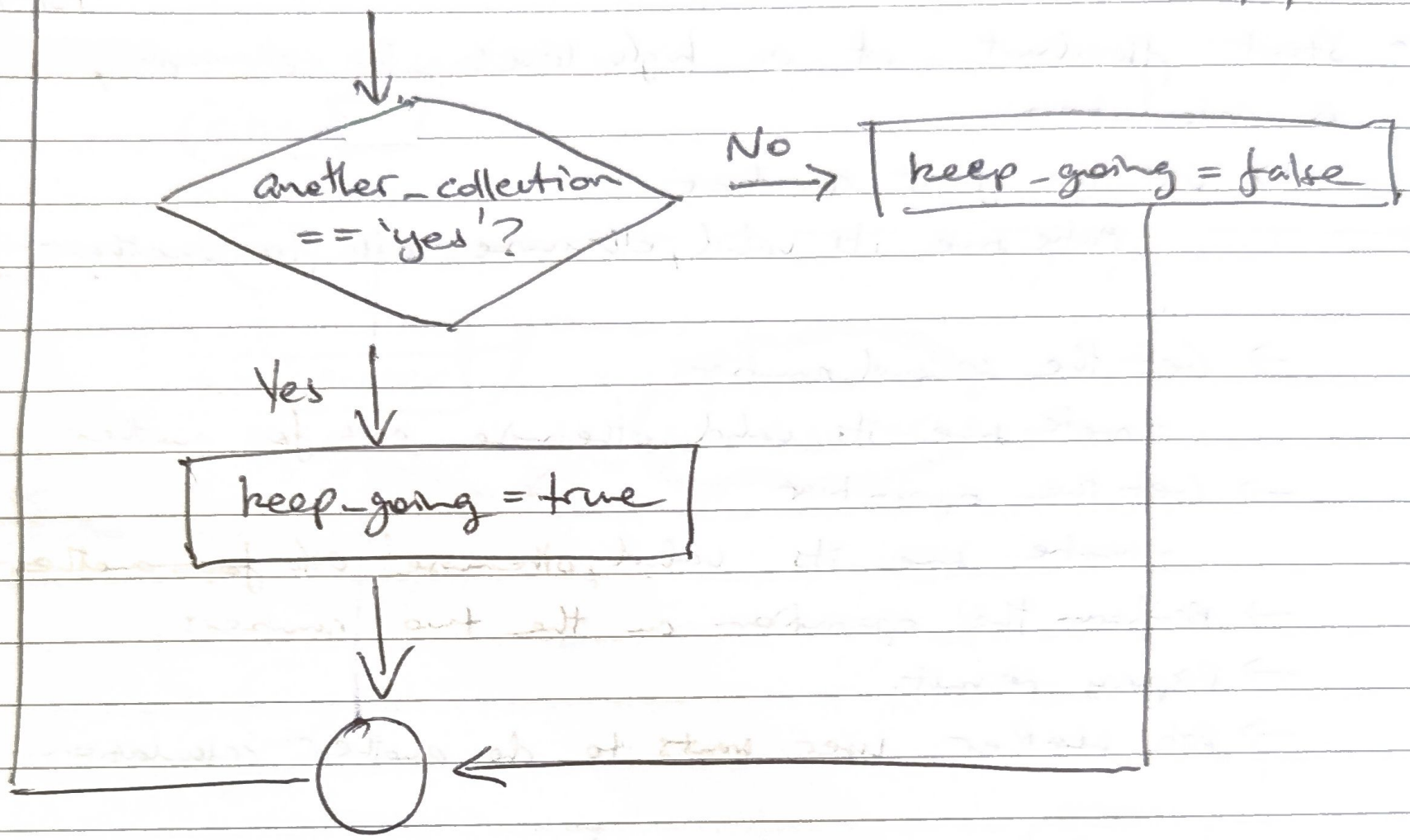
keyword 'SUBPROCESS' indicates that some other process will extract the largest number.

17/05/2025

FLOWCHART



17/05/2025



- Moving logic to sub-processes we use (declarative) (syntax) rather than (imperative)

(syntax) → the rules on how code should be written

(imperative) → how to do it.

(declarative) → what you want

We trust that the (sub-process) will do its job

Can work on imperative logic and ignore the rest of the program when we are ready.

18/05/2025

- Start flowchart at a high level, For example, a calculator:
  - → Get the first number
    - Make sure its valid, otherwise ask for another
  - → Get the second number
    - make sure its valid, otherwise ask for another
  - → Get the operator
    - make sure its valid, otherwise ask again.
  - → Perform the operation on the two numbers
  - → Display result.
  - → Ask whether user wants to do another calculation.

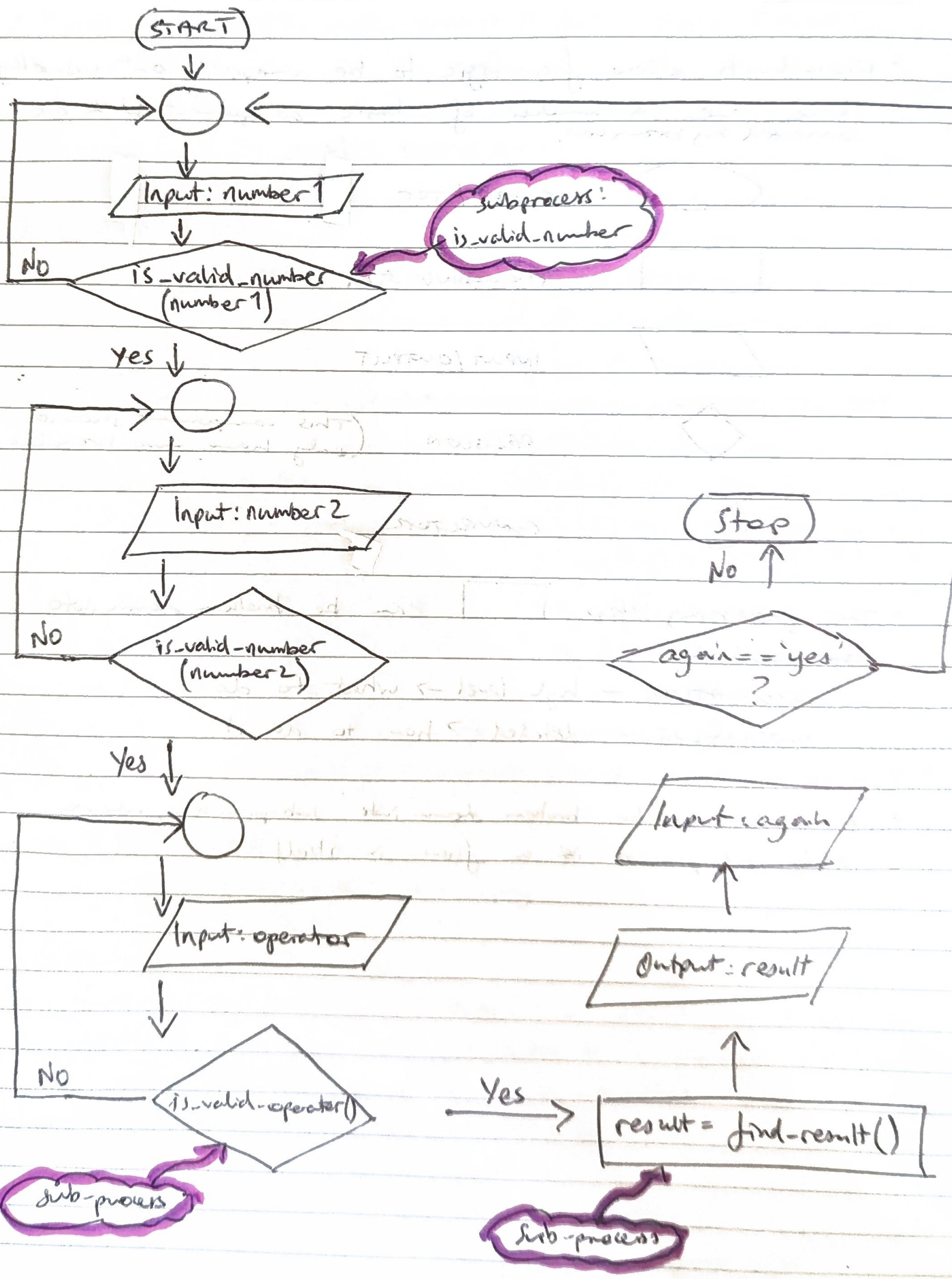
The above is a good starting level prior to drilling down to <u>imperative</u> pseudocode and outlining specifics.

IMPORTANT

LS INTERVIEW CODING ASSESSMENTS EXPECT THE LOGIC TO BE LAID OUT PRIOR TO CODING.

18/05/2025

FLOWCHARTING THE CALCULATOR



18/05/2025

Summary:

- Flowcharts allow for logic to be mapped out visually.
- There are a number of basic components which are connected by branches.

START / STOP

PROCESSING STEP

INPUT / OUTPUT

DECISION (This component should only have two branches)

CONNECTOR

- The processing step  can be broken down into two methods:

  DECLARATIVE — high level → what to do

  IMPERATIVE — detailed → how to do it.

- Processes can be broken down into sub-processes, where each sub-process is a flow in itself.