Basics

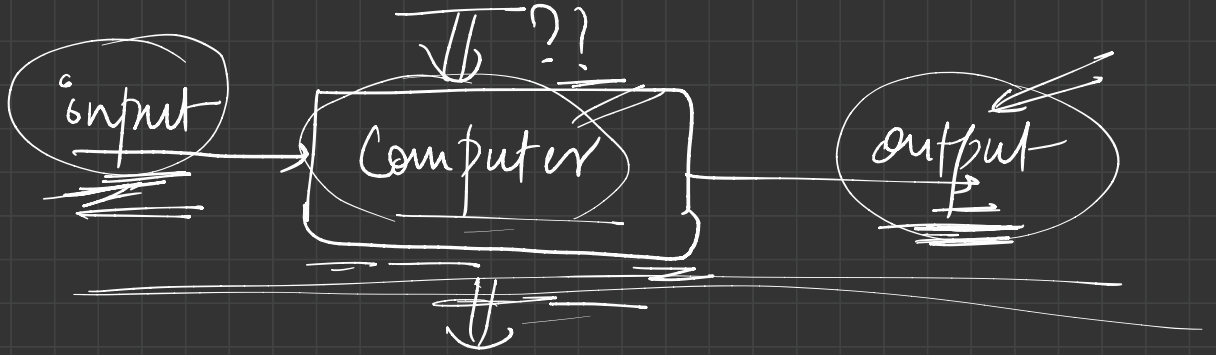
C++

{ Basics
Fundamentals

C++ another language?

C++?
Algorithms?
Programs?

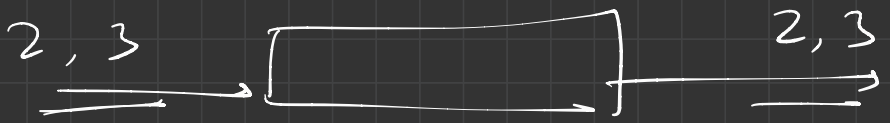
{ we are trying to communicate
with the computer? }



tools

thing going on here?

sensible



no work

to do something and to get some output we give some instructions

↓

Programs

How to add two numbers?

Step ① we take the first number

② we take the second number

③ we perform addition (maths)

④ and we tell the result.

→ algorithm to add two numbers

We need to provide the computer with some instructions for it to work / give output. (and input)

→ the set of instructions to get the output for some input ⇒ Algorithm

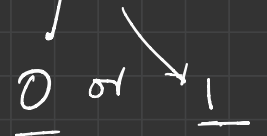
- must be clear
- must be precise
- must be correct

Execution / Run

Computer → we are telling how to work on input and get output

{giving a set of instructions}

Computer only understands binary



→ It is impossible to communicate / instruct with the computer in 0s and 1s.
binary

→ Computer understands Binary 0s and 1s

→ we understand → English / Hindi, etc.

How do we communicate?

Problem → Computer → Solution

Problem ⇒ give instruction in human readable | understandable

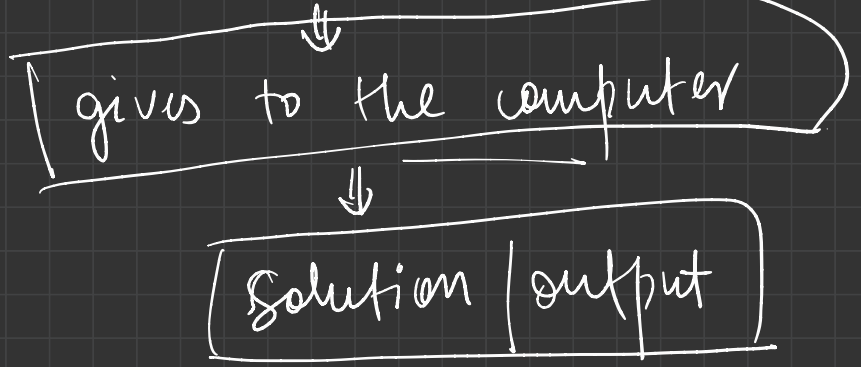
input

language ⇒ many

which one to use?

translator

translates to Binary (0s and 1s)



× Binary → ×

× English
× Hindi
} Spoken Languages

= C++ / Java / ---

advantages
→ easy to read write understand

disadvantages
→ computer doesn't understand

→ there are many

↓

We need as many translators

→ no proper set of rules

↓

different interpretations.

imprecise

→ It is not always possible to instruct the computer properly in a spoken language.

↓

too many ?

Binary

Spoken Language

→ We need some standard language to communicate with the computer.

⇓

Programming Languages

(C, C++, Java, Python, ———)

⇓

C++

A programming language is a set of rules that gives us a way to communicate with the computer.

↓

provide instructions

tell what to do.

- understandable by the humans
- should be able to properly translated to binary.

Syntax: grammar/set of rules of the programming language.

Programming Languages

- High level — more understandable by humans
- Low level — more understandable by computers

Python, Java, C++, C, Assembly, Machine Code

Python → almost like english

```
print("Hello")
```

High level

1. C/C++/Java/Python ← we use this
2. Assembly

low level —

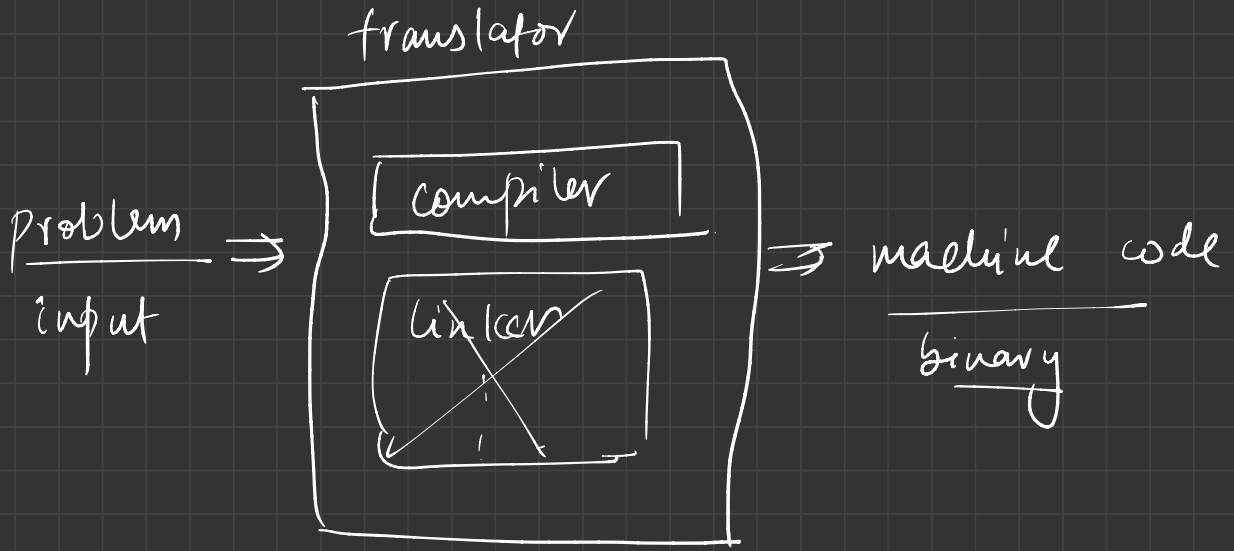
3. Machine code → machine readable code
4. Binary → Computers absolutely understand

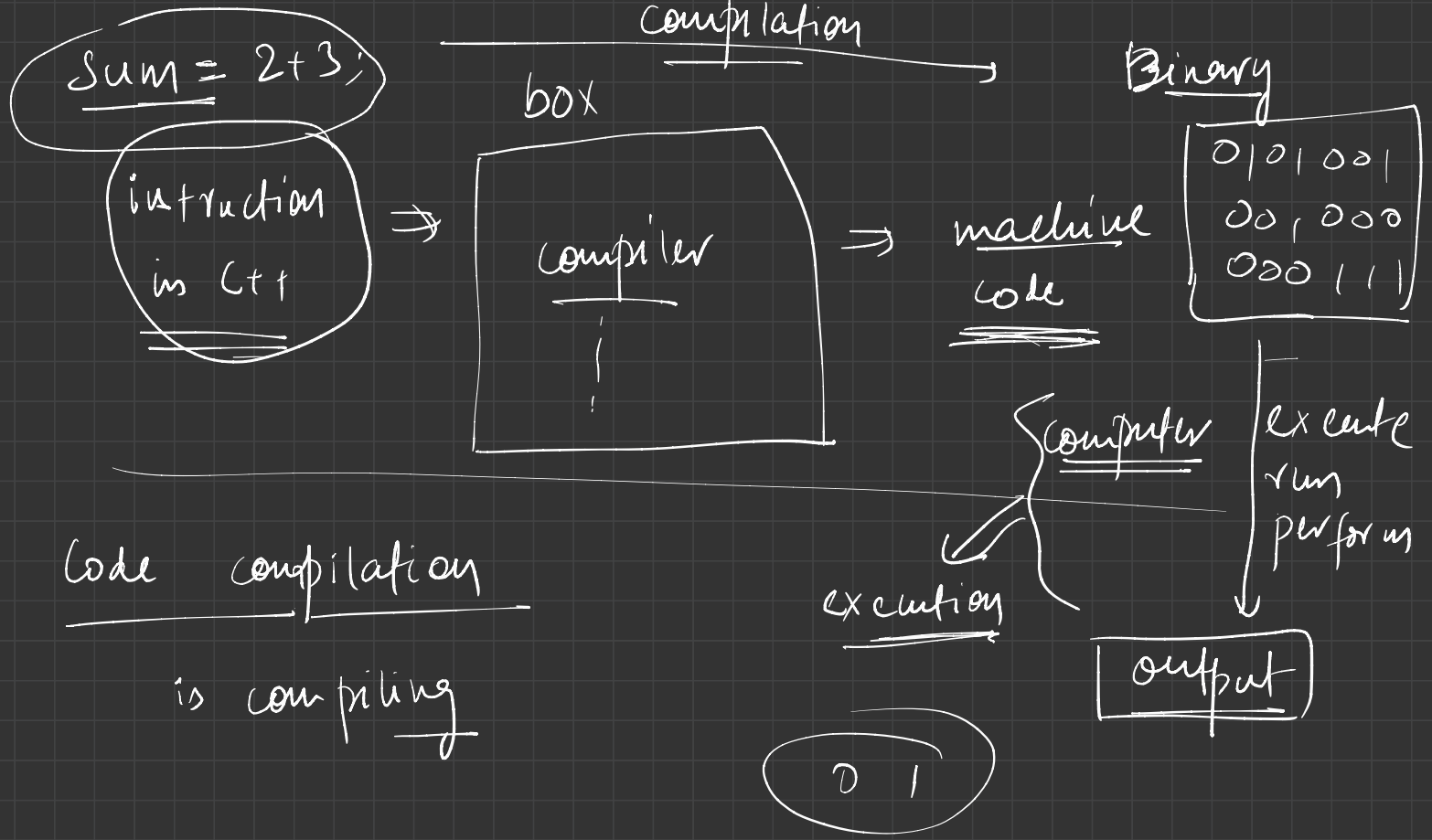
We only use High-level languages

⇓

C++ ← we will use

how does it work?

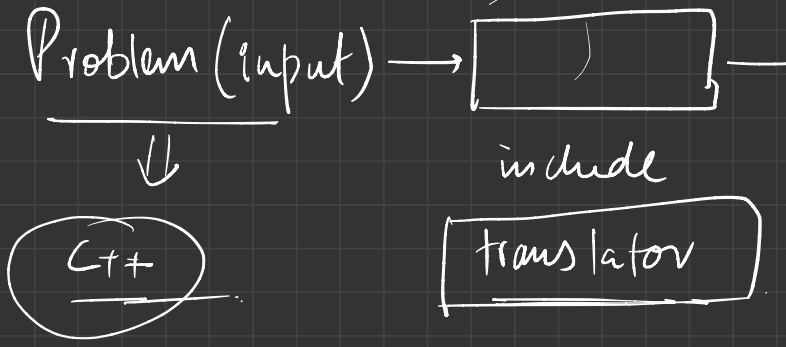




Code compilation

is compiling

a, b, c =



set of instructions ⇒ algorithm

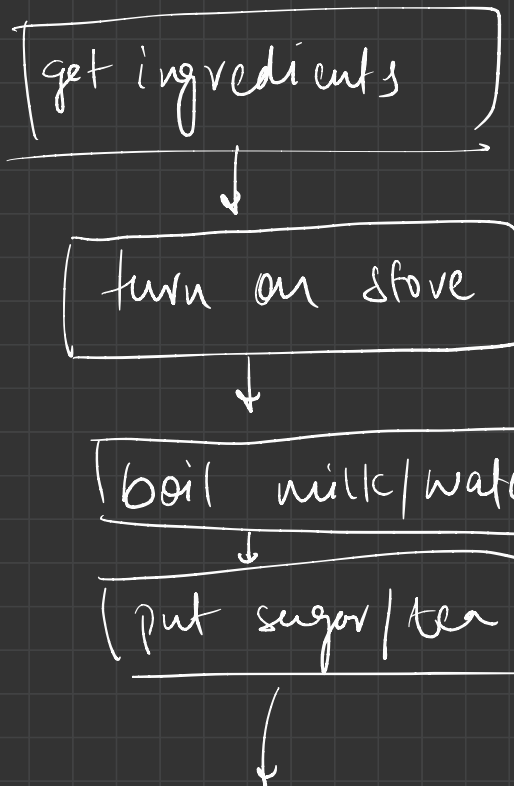
How to make tea? ⇒ can we say this the algorithm?

1. we gather the ingredients (tealeaves, sugar, water, ...)
2. we ignite / turn on the stove
3. we boil water / milk

(1)

④ we put sugar/tea leaves, etc.

⑤ we wait for 5 mins



flowchart

2

wait for 5 mins

→ The same algorithm can be expressed as

spoken language, flowchart, code (programming language), pseudo code.

spoken language → complex & difficult to translate

code → best for computers

Programming → Data
→ Instructions

Data → information
input value

→ we need a way to represent

→ we store data in variables

x, y

sum = 2 + 3 = 5

sum = a + b

variables

first no.

second no.

→ sum

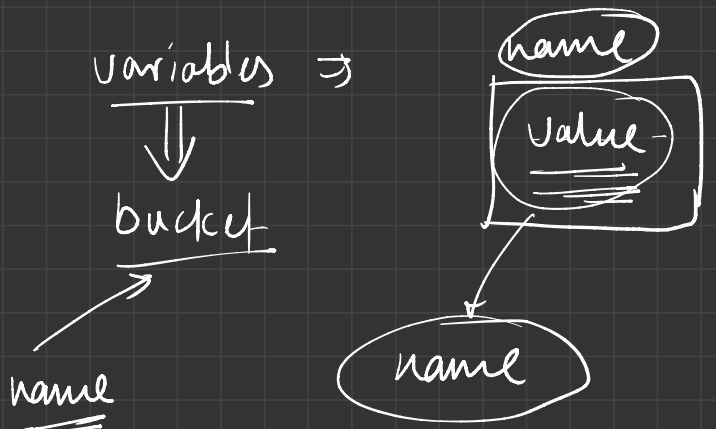
5

a

2

b

3



Sum = a + b

2 + 3 = 5

→ getting the value

→ setting the value

→ creating a variable

naming rules (specific to C++)

① it must start with a letter or _ (underscore)

② and must only contain letters, numbers, or _

0. a ✓ → can be a variable name

Which of the following is not a proper variable name?

1. sum ✓
2. _sum ✓
3. _0abc ✓
4. 1vbc ✗
5. Rat246 ?? ✓
6. rat_12345_abc ✓
7. pqr%abc ✗

23%abc ✗

{a-z, A-Z} ✓

Instruction? → an order

→ Instructing a computer

{ spoken lang.
prog^n. lang.
binary (0100 11) }

→ Instruction: consisting of only 0s and 1s (low level)

→ some commands (high level)

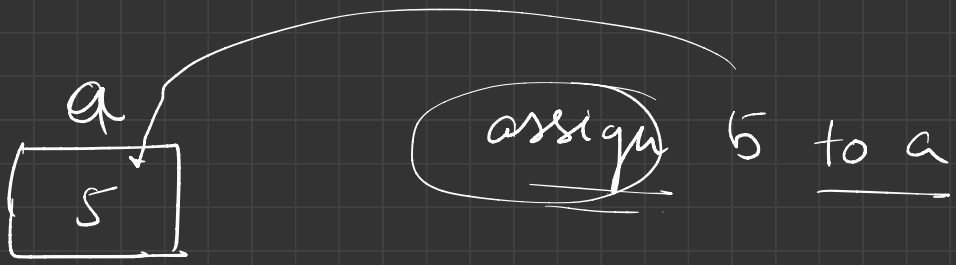
cout << "Hello World"; in C++

↓

output → Hello World

Instructions

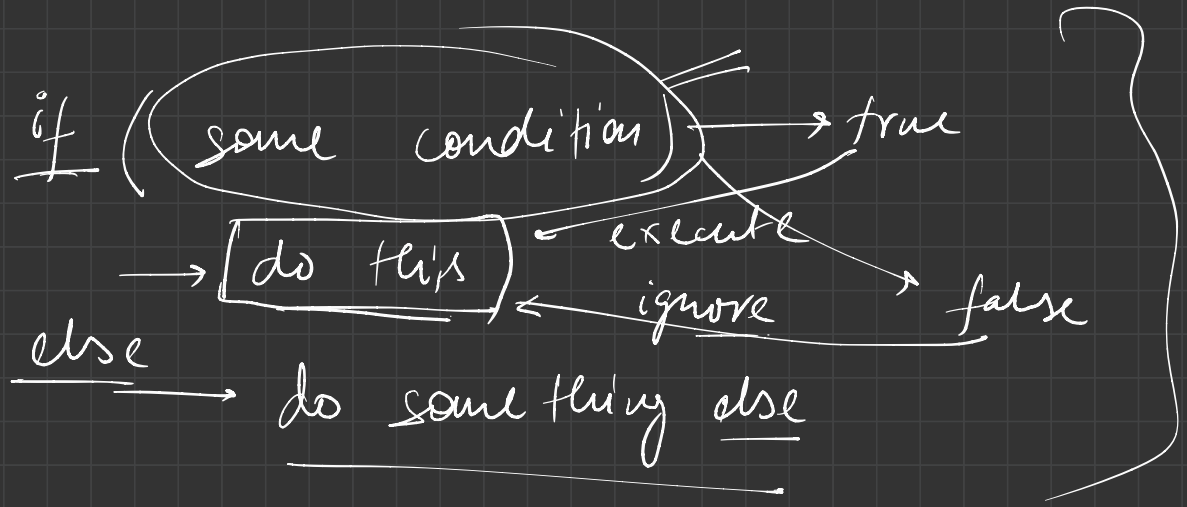
1. read / receive from a variable / taking an input
2. write / output / print
3. perform some operation (arithmetic ops).
   - ↳ add, sub, multiply, divide
4. assign a value to a variable (set)
5. do something based on some condition ⇒ Conditionals
6. repeat something ⇒ Loops



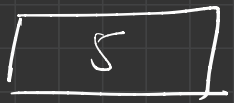
Conditionals (If-Else)

if (tomorrow is Sunday) then it is a Holiday,
else it is not a Holiday.

```
if (tomorrow is Sunday)
    → go to class
else
    → do not go to class
```



Doing something based on whether the given condition is true / false.



if (a is divisible by 2)

→ output (a is even) & ignore

{ else

→ output (a is odd) }

Loops

→ do something many times

→ output (Hello World)

Q. Output "Hello World" 5 times ?

→ output (Hello World)
→ output (Hello World)
→ output (HelloWorld)
→ output (HelloWorld)
→ output (Hello World)

Solve !!

1000

→ instead use a loop → true

→ while (some condition)

→ do something

for-loop
do-while
do-until

(var) repetitionsLeft = 5

while (repetitionsLeft > 0) {

output ("Hello World")

decrease (repetitionsLeft)

}

Output

1 2 3

3

Dry Run:

repetitionsLeft = 3

repetitionsLeft > 0 → true

3 > 0 ✓

→ Write "I will not misbehave" 100 times.

I will not misbehave
I will not misbehave
⋮

⇒ Computer

output ("I will not misbehave")
output
output
⋮
100 times

→ While - loop

→ while (some condition)

→ do something X

if( )

→ ( )

else

→ ( )

keep counting from 1 onwards

1, 2, 3, ——, 101

num × 2

(101)

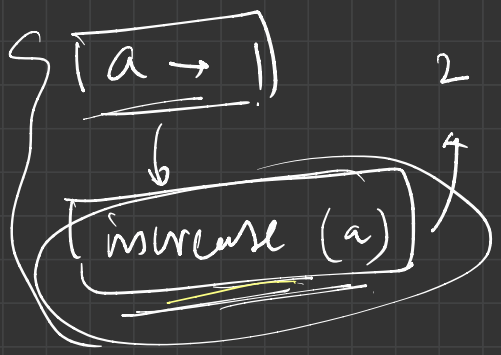
101 <= 100

while (count is <= 100)

→ output ("I will not misbehave")

Hello World

this is executed 100 times but written only once



2 → increase (a) → 3 → increase (a) → 4

⇒ will it behave same as the counting person above?

loop?

a → 1 2 3 ... 100 101

while (a <= 100) {

→ output ("I ---- ")

→ increase (a)

}

count "

print "I will not misbehave"

100 times

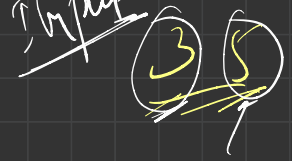
a = a + 1

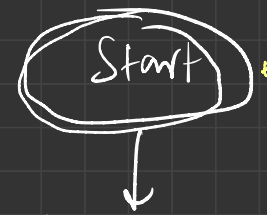
Flow chart → diagrammatic representation of

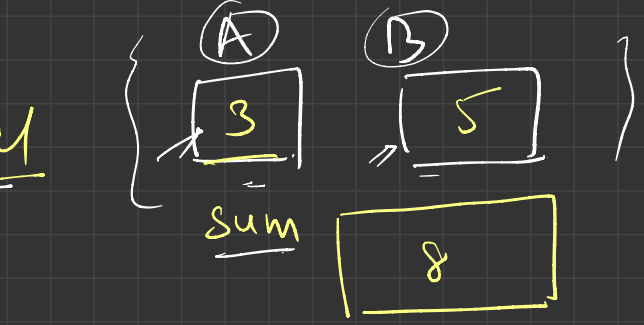
Pseudocode → set of instructions

- read
- write
- assign
- perform operation
- conditionals
- loops

Q. Compare two numbers ?
and sum

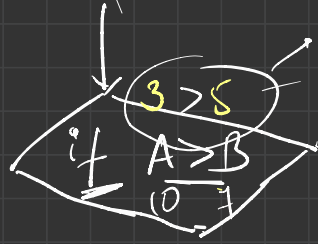






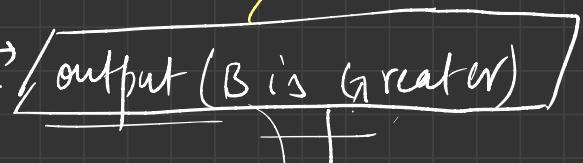
read/write instructions

process

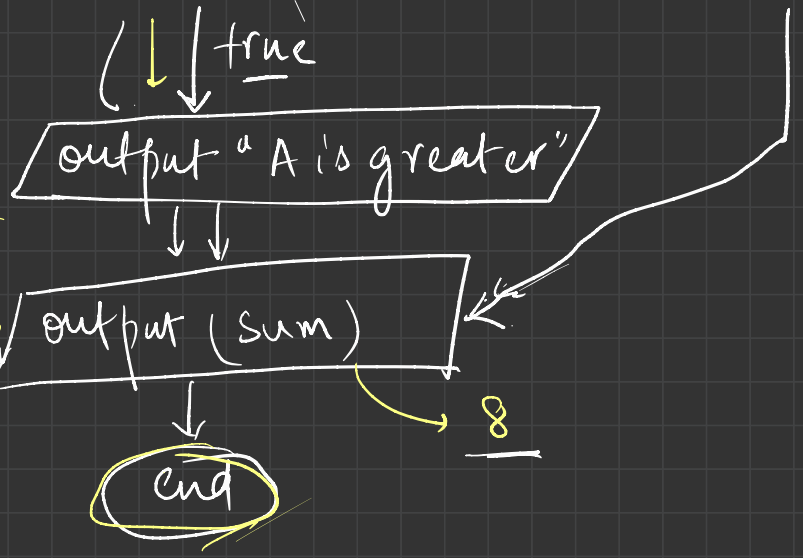


Conditionals
loops

false



B is greater



Programming Lang

Pseudo Code ⇒ an abstract way to represent instructions / flowchart

1. start
2. read A
3. read B
4. sum ← A+B
5. if (A>B)
   output ("A is greater")
   else
   output ("B is greater")
6. output (sum)

(7) end

how does it gets the solution / output

⇓

following inst: (algorithm)