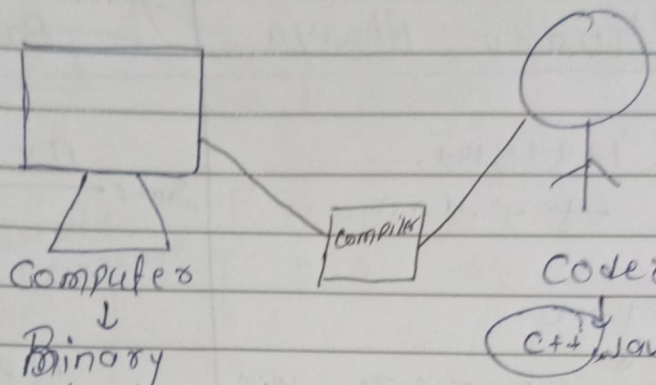
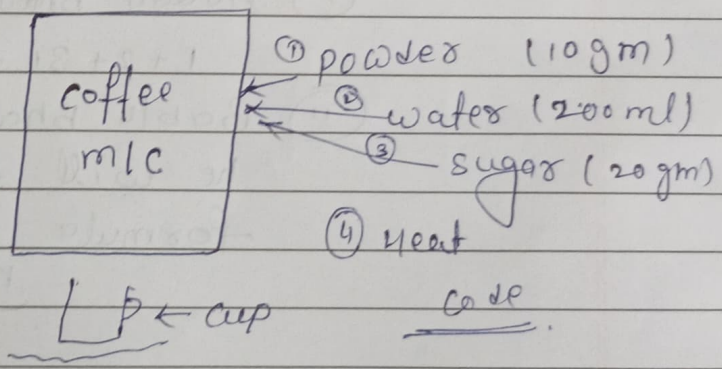
Flowcharts and PseudoCode



For given problem -

(i) understand problem
(ii) given value
(iii) Approach develop
(iv) code
(v) Error / debug
(vi) other solution.

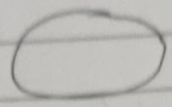
Instruction (pseudocode)

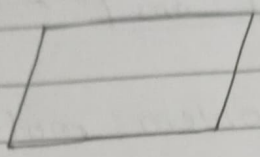


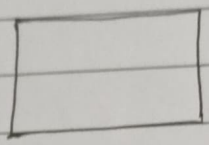
Suppose we have to make a cup of coffee. then we go to coffee m/c and code for making coffee is already written in the backend in the m/c. we don't need to write again and again for the same coffee making instruction.

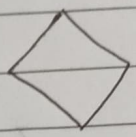
In the same way, for any problem we write (save) solution once and can reuse the same code as much as we need.

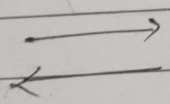
→ Flowchart Terminologies

1.  → Terminal (start / end)

2.  → Parallelogram (i/p ; o/p)

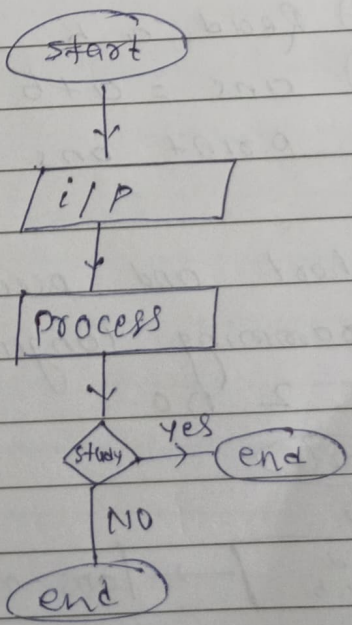
3.  → Rectangle (process)

4.  → Diamond (Decision)

5.  → Arrow. (Flow of program).

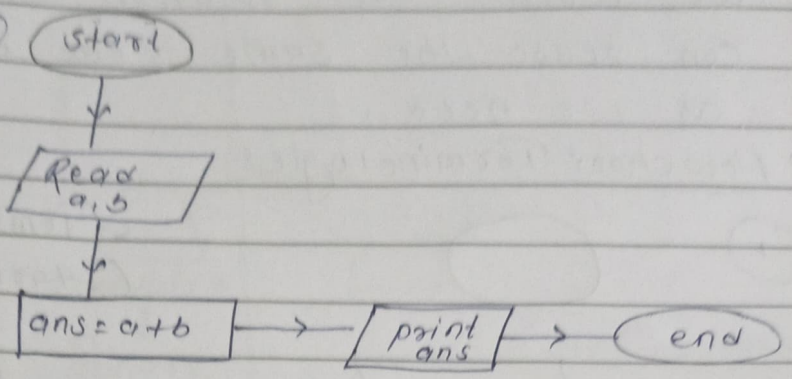
e.g

Life



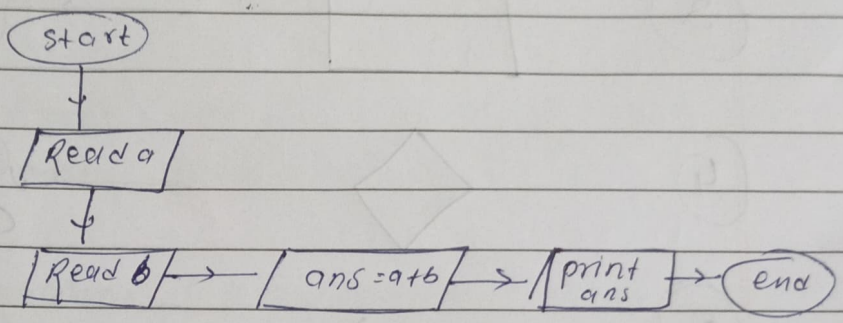
1. Sum of 2 no.

a.



→ for one problem multiple flowchart is possible.

b.

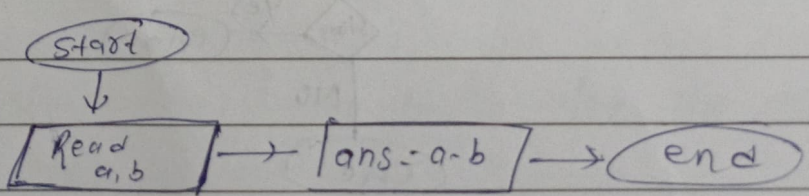


Pseude code

1. Read a, b
2. ans = a+b
3. print ans

flowchart and pseude code are language (programming language) independent.

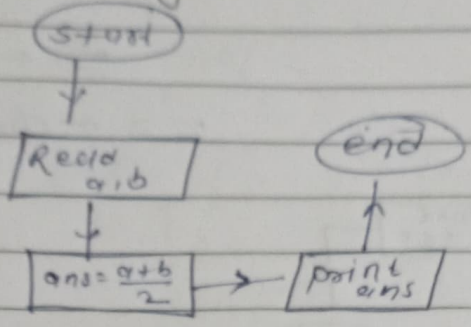
2. Sub 2 no.



H.W (i) Product of 2 no. } Flowchart
(ii) Divide 2 no. } pseudocode.

(3) Average of 2 no.

avg. $= \frac{a+b}{2}$

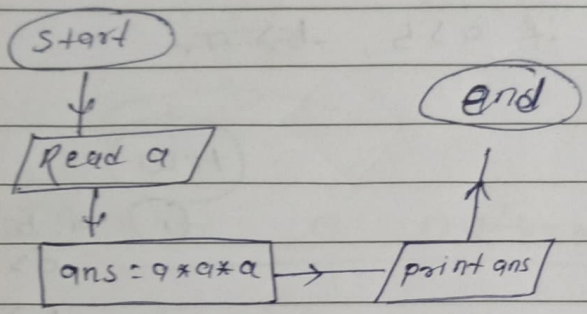


Pseudo code

(1) Read a,b
(2) ans = $\frac{a+b}{2}$
(3) print ans.

(4) Find cube of a no.

$2^3 = 8$
$3^3 = 27$



Pseudo code.

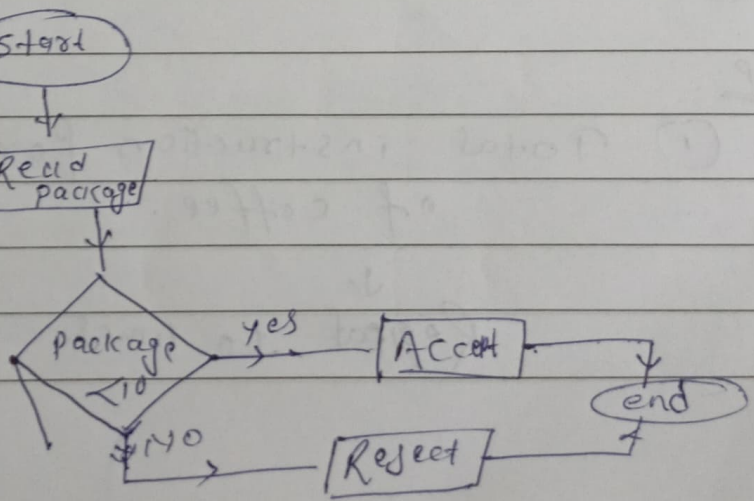
(i) Read a no.
(ii) ans = no. x no. x no
(iii) print ans.

H.W

(1) Find square of a no.

Decision making.

(5) MNC company Cracle
package >10 ✓



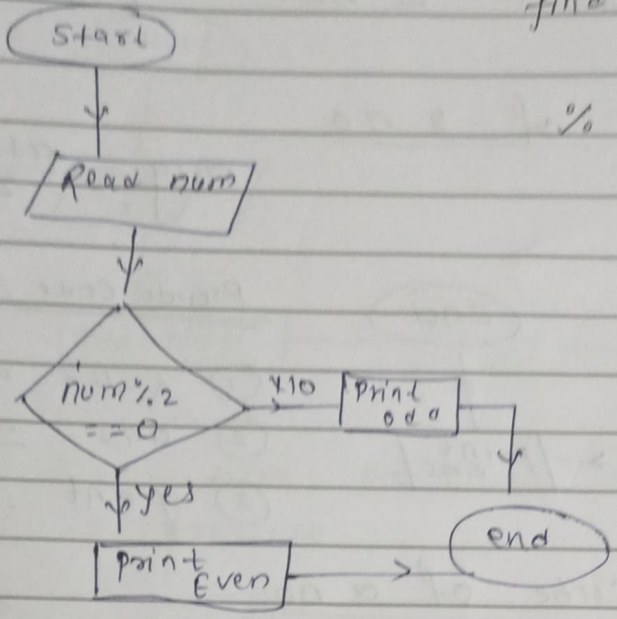
(6) Check no. is Even or odd.

find remainder
→ 0 even
→ 1 odd

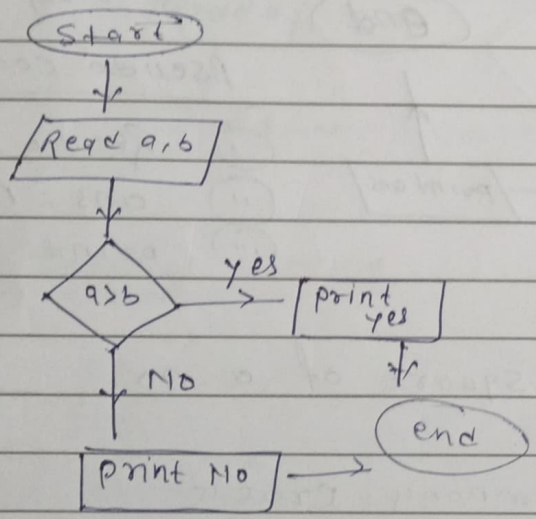
% ← modulus.

= ← Assigment

== ← checks



(7) a, b if a > b, ~~b > a~~



H·W

(1) a, b
a > b
b > a
a == b

H·W

(8) (11) num is +ve, -ve or zero.

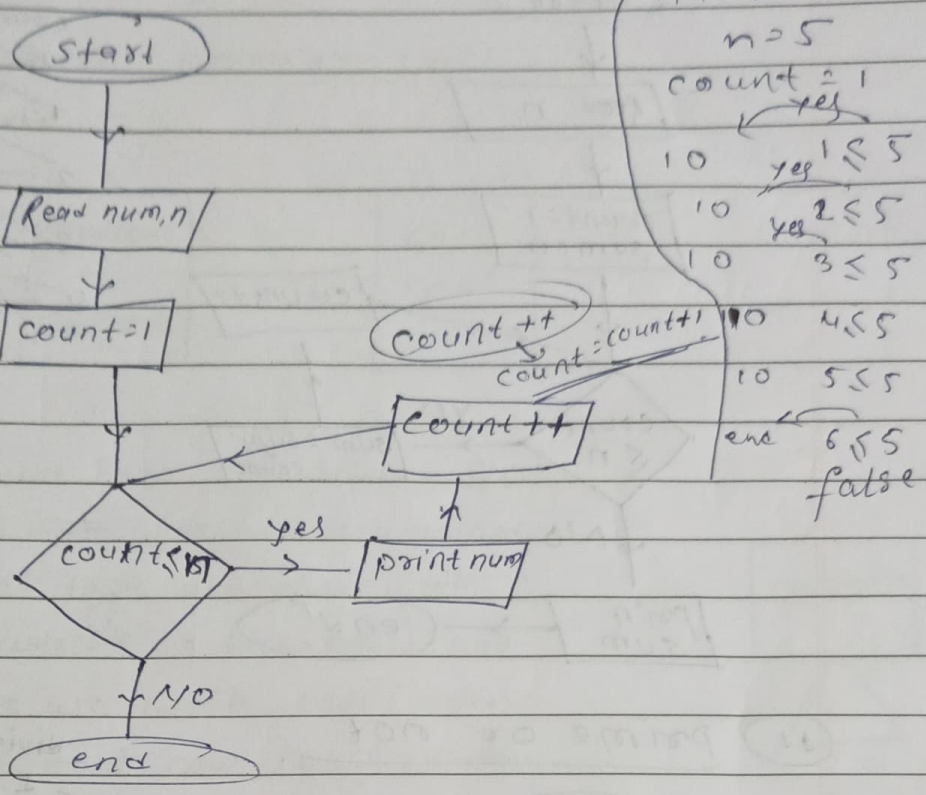
Loop.

(1) Total instruction for 1 cup of coffee.

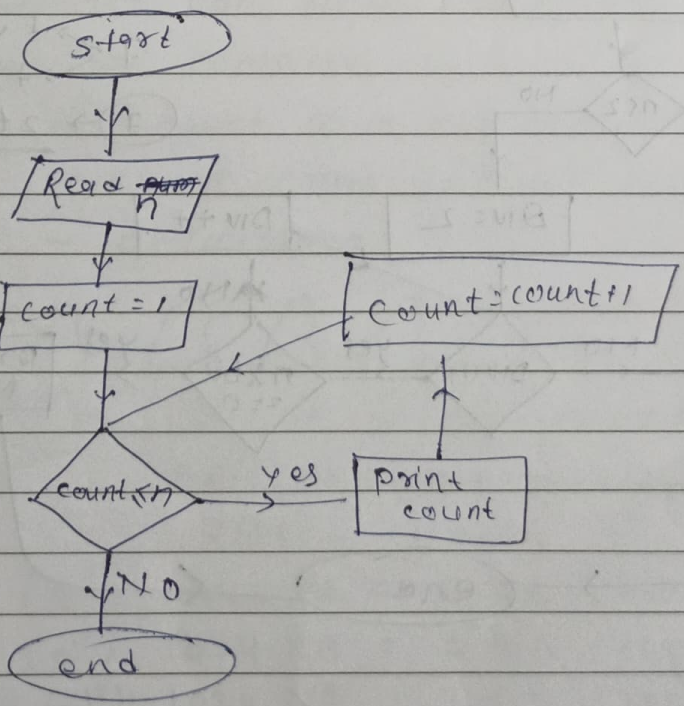
↓

Repeat 10 times.

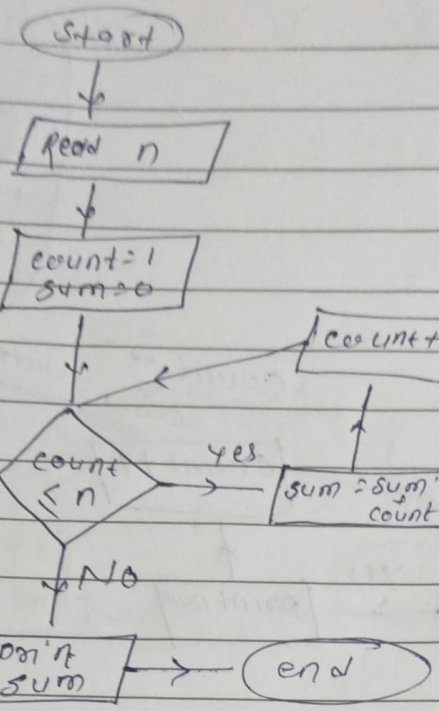
(8) Print num n times.



(9) Print n natural no.



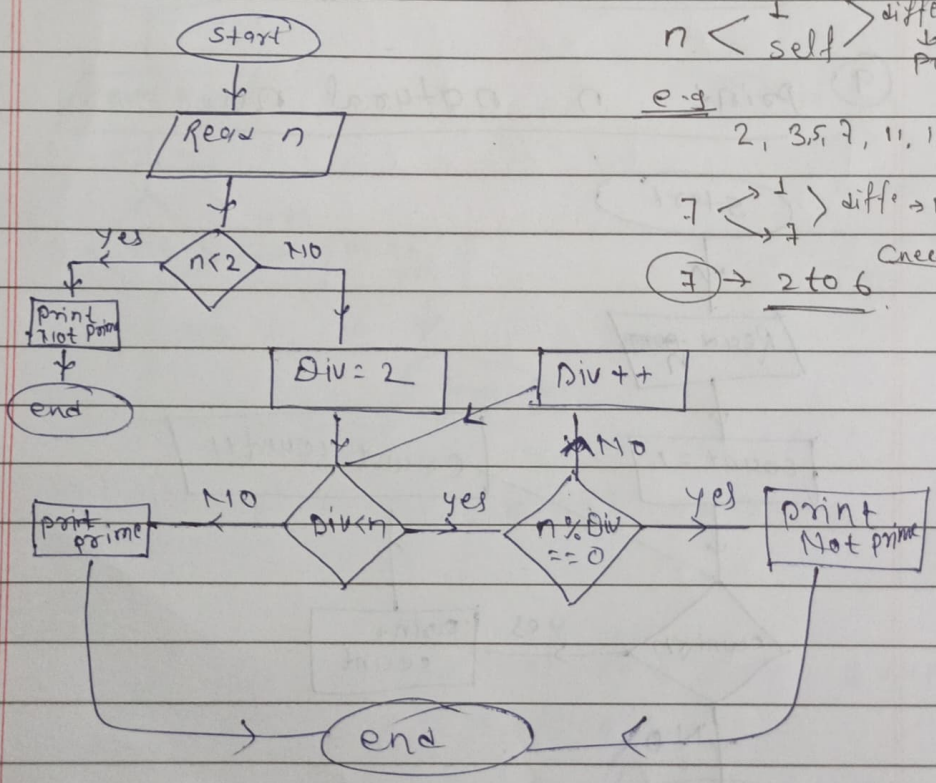
(10) Sum of n natural no.



sum = 0
count = 1

1 → 1
2 → 3
3 → 6
4 → 10

(11) prime or not.



Divide
n < 1, self > different
↓
prime.

e.g
2, 3, 5, 7, 11, 13 ...

7 < 1, 7 > diff → prime.

7 → 2 to 6 Check