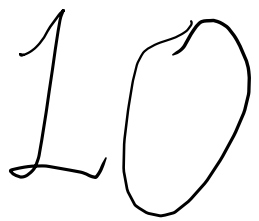
Let’s solve it



Logical operators

- It has to do with conditions (expressions)

condition1 has to be true and
condition2 has to be true

&& logical and

|| logical or

! NOT

Truth table

&& → All have to be true

True	&&	True	⇒ True
True	&&	False	⇒ False
False	&&	True	⇒ False
False	&&	False	⇒ False

|| → Atleast one

True	||	True	⇒ True
True	||	False	⇒ True
False	||	True	⇒ True
False	||	False	⇒ False

Ternary (conditional) operator

(data%2)==0 ? printf("even"); : printf("odd");

op1 — op2 (this) — op3 (that)

True / False

3 operands

? — colon

; — semi colon

colon / column (rows)

2 | 37 → 18, 36, remainder 1 → odd

2 | 24 → 12, 24, remainder 00 → even

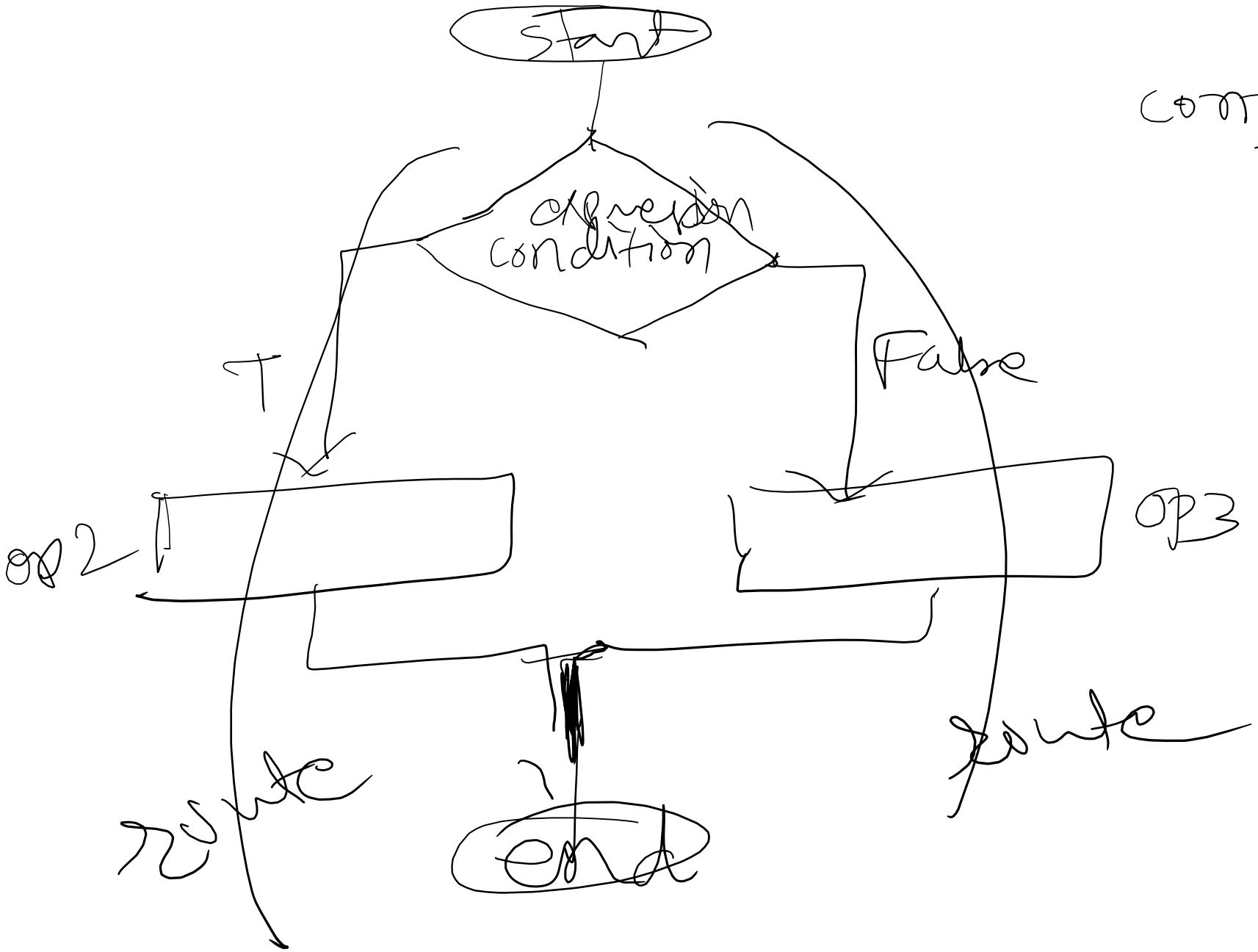
int data;

scanf("%d", &data);

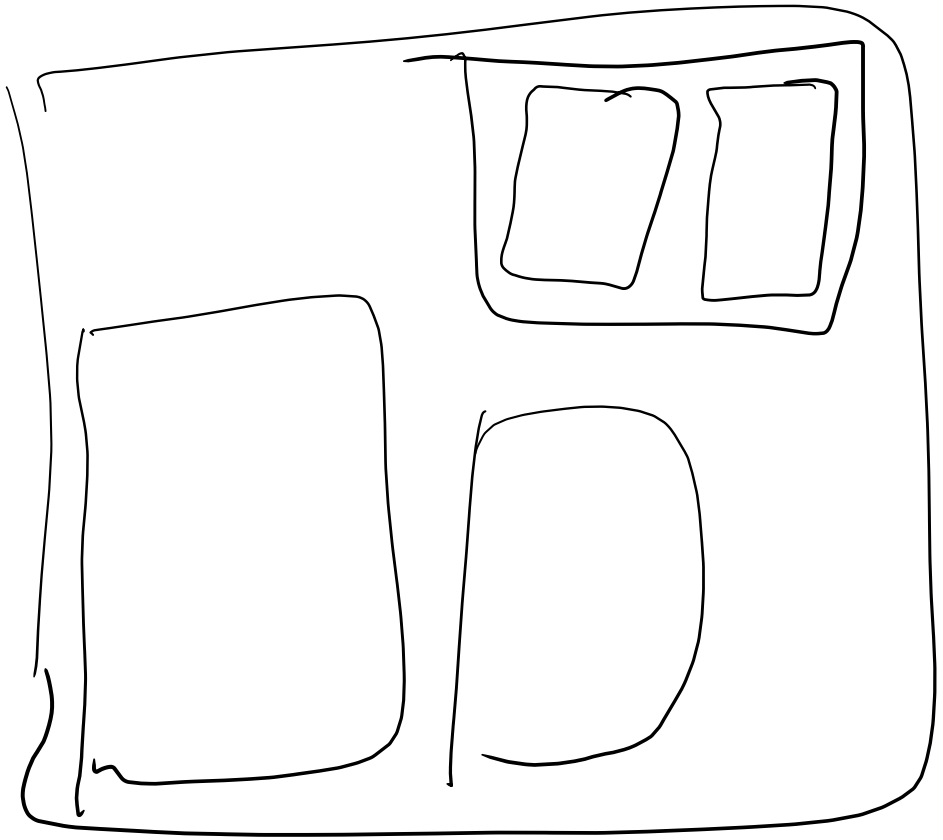
printf("odd");

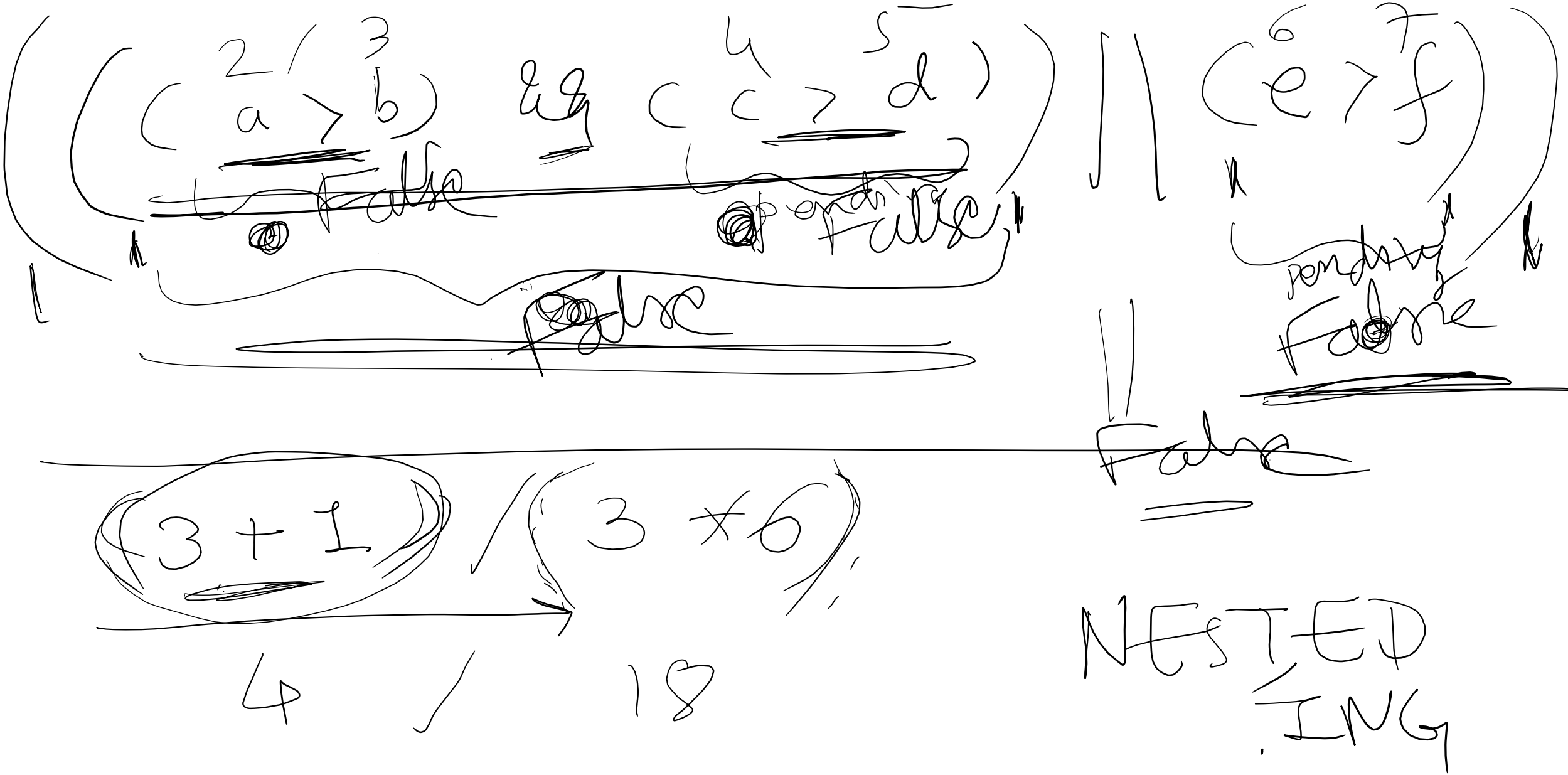
printf("even");

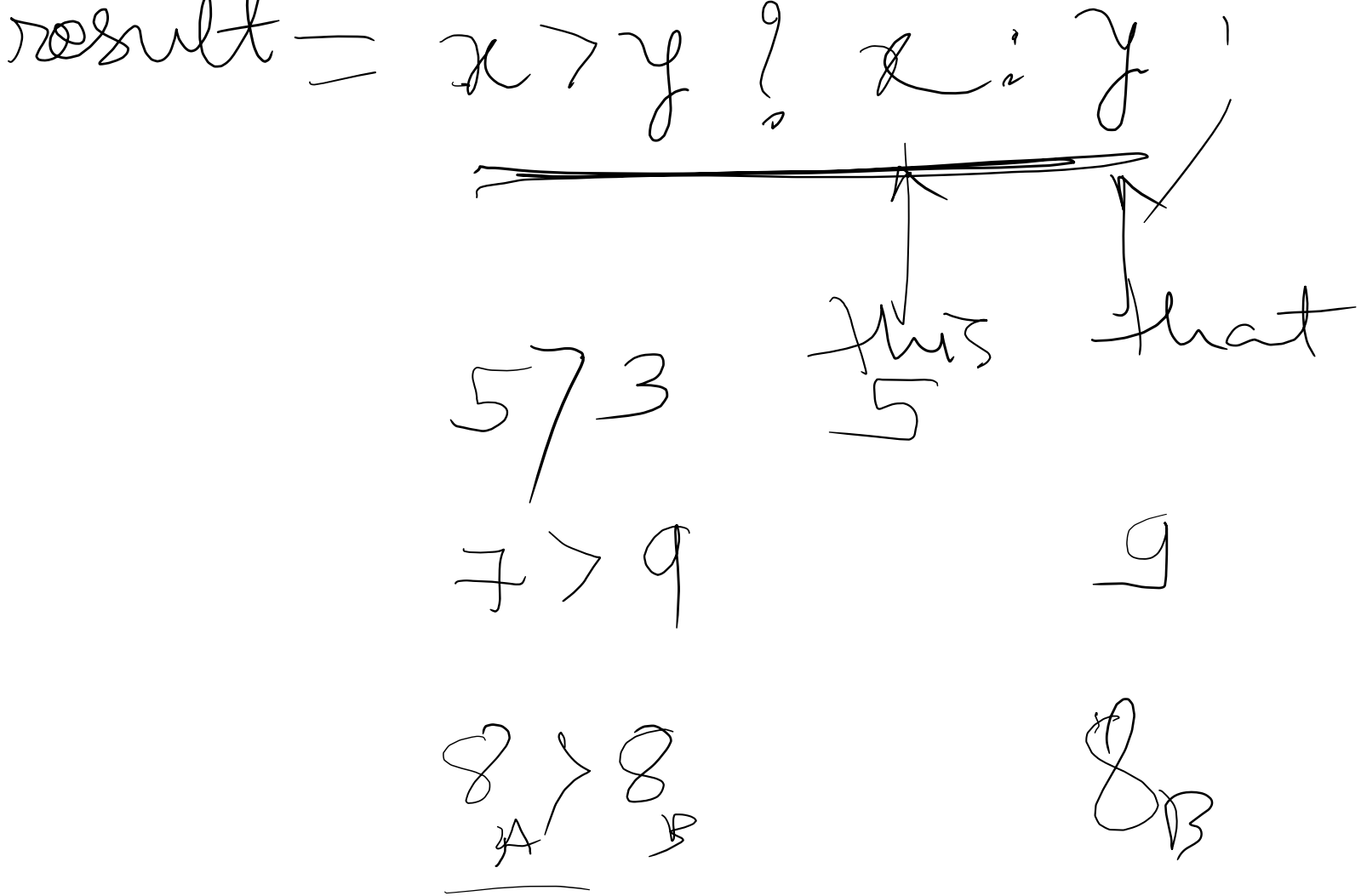
op1 is expression evaluation is done.


Start
operation condition
control flow
T
False
op2
op3
write
write
end

Delivery by Amazon




2 3
(a > b)
&&
4 5
(c > d)
||
6 7
(e > f)
False
False
False
pending
False
||
False
(3 + 1)
/
(3 * 6)
4
/
18
NESTED
ING


result = x > y ? x : y ;
this
that
5 7 3
5
7 > 9
9
8 > 8
8
8

Bitwise operators

& bitwise and

| bitwise or

caret ^ bitwise exclusive OR (XOR)

Left... << left shift

>> ... right Right shift

bitwise &

$(1)_2 \& (0)_2 \Rightarrow 0$

$(1)_2 \& (1)_2 \Rightarrow 1$

0

0

---

masking &

```
  1 1 0 1 0 1
  0 0 0 0 0 0
-------------
  0 0 0 0 0 0
```

255.255.255.0

Networking

```
int x = 255;
x << 1;
```

1 1 1 1 1 1 1 1

Hindi

- multiply without using *
  by two
- divide by two without using /

Can shift bitwise help here?

Note that shifting either prepends or appends empty positions with ϕ. (zeroes)
It's called padding.

Note that circular shift is a different thing.