Operators :->

int a = 10;
int b = 20;
int sum = a+b;

Keyword → int; Identifier / variable → sum; operands → a, b; operator → +; Terminating semicolon → ;

Expression

1. Arithmetic Operators : +, -, /, *, % , Python: **, //
2. Assignment ✓
   * Augmented ✓
   * Short-hand ✓

   Operators: =, +=, -= . . . . .

   a = 10
   a = a+2
   a += 2

   $3 \overline{)10}$ → 3, 9, remainder ①; 10%3 = 1; 10//3 = 3

   3%4 = 3

3. Comparison / Relational : >, <, >=, <=, ==, != [Boolean]
4. Logical : &&, ||, ! [and, or, not]
5. Unary : ++ / -- (Inc/dec) (Prefix / Postfix)
   C/C++/J/JS
6. Ternary Operator : (con) ? :
   C/C++/JS: (condition) ? TV : FV;
   Java: datatype var = (condition) ? TV : FV;
   Short-hand if-else operator.

Bitwise Operators:

(I) AND → & → Ampersand
(II) OR → | → Pipe
(III) XOR → ^ → Caret
(IV) RS → >> → Angular Brackets
(V) LS → <<
(VI) NOT → ~ → Negation / Tilde

6 → 0110
9 → 1001
6&9 → 0000

5 0101
8 1000
5|8 1101 = 13

S-0, D-1
11 → 1011
4 → 0100
11^4 → 1111 = ⑮

Bitwise - NOT

1	0
0	1
T	F
F	T

5 — 0101
~5 — 1010 = 10
~5 = 10
~5 = -6
[-6 = 10]

-6 = 10 → Theoretical Value
abs(-6) = 6 → 0110

double negation → Reciprocity

2's com { 1's com → 1001, +1 → 0001 } → 1010 = 10

~n = -n-1
~5 = -5-1 = -6
~-500 = -(-500) 500-1 = 499

MCQ → ~5 (a) 1 (b) 2 (c) 5 (d) -6 ✓

~~1, 2, 3, 6, 3, 2, 1~~

0000
0001
0001
0010
0011
0011
0000

0000
0110
0110
0011
0101
0010
0111

DRY RUN r = 0
0111
0001
0110 = 6

Decision Making / Looping / Branching

① Conditional Statements :
  a) Simple if - statement → 1 condition
  b) if - else statement → 2 condition
  ✓c) if - else if - else → More than 2
  d) Nested - if → Branching / Nesting
  e) Switch case → Better version of ©
  f) Ternary Operator → ✓

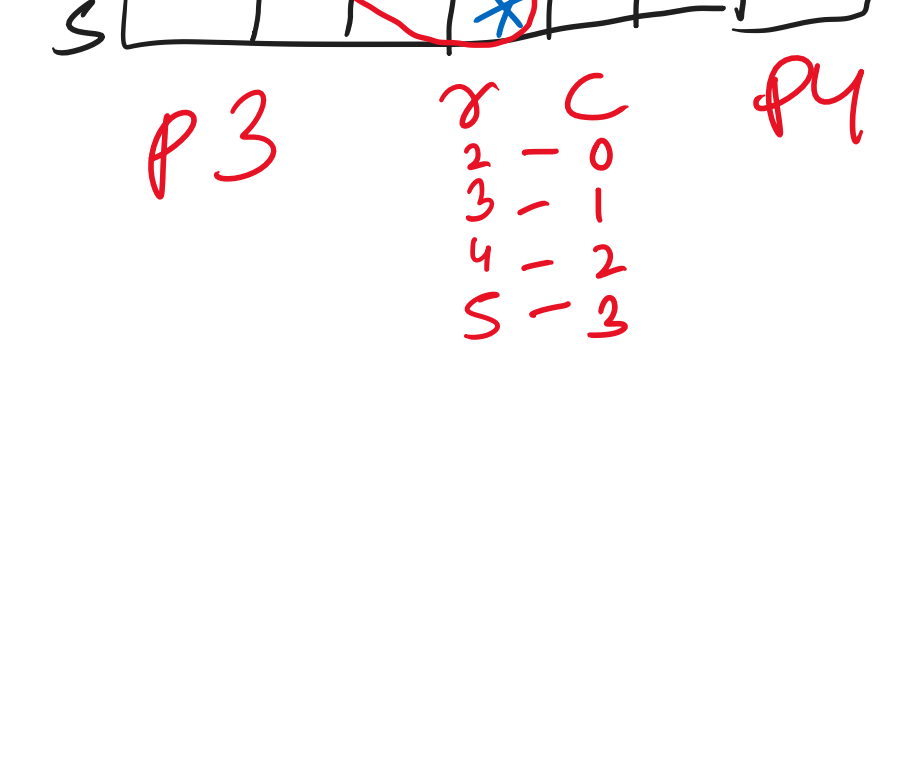
Loops

(I) While → Entry Controlled Loop (we don't know the no. of steps)
(II) do - while → Exit Controlled Loop
(III) for loop → we know the no. of steps
(IV) for-each loop / Enhanced for loop
(V) Nested Loops

Patterns ✓

TCS → Ninja → ~~6-8 LPA → 3 mins~~
→ NQT → ~~5-6 LPA → 4 mins~~
→ Basic → ~~2.5-3.5 LPA → 5 mins~~

Divide into Parts



P1 r = 0, c = 1, 2, 4, 5
r == 0 && c%3 != 0
P2 r == 1 && c%3 == 0
P3 r-c == 2 (r c: 2-0, 3-1, 4-2, 5-3)
P4 r+c == 8 (r c: 2+6, 3+5, 4+4)