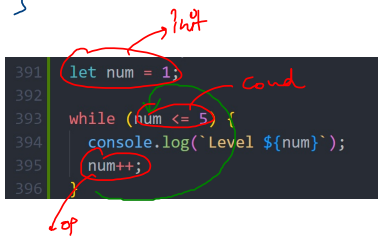
★ While Loop :

```
while (condition) {
  /* code */
}
```

★ Code repeats until condition fails.



⑤ num <= 5 ⇒ 5 <= 5 ⇒ true
→ "Level 5"
→ num ++ ⇒ num = 6

⑥ num <= 5 ⇒ 6 <= 5 ⇒ False
← stop the loop →*

(391) let num = 1;

num: 1 | 2 3 4 5 6

(393)

① num <= 5 ⇒ 1 <= 5 ⇒ True
→ "Level 1"
→ num++ ⇒ num = 2

② num <= 5 ⇒ 2 <= 5 ⇒ True
→ "Level 2"
→ num++ ⇒ num = 3

③ num <= 5 ⇒ 3 <= 5 ⇒ true
→ "Level 3"
→ num++ ⇒ num = 4

④ num <= 5 ⇒ 4 <= 5 ⇒ true
→ "Level 4"
→ num++ ⇒ num = 5

★ If num++ is not here Level 1 printed Infinite times. (this cases in memory overflow). (runtime Error) (Time limit Exceed)

→ Any problem that can be solved with for loop can be done with while loop as well and vice versa.

→ When to use for/while ⇒ you can use anything which is comfortable to write the code.

→ for has all 3 things together : init, cond, operation.

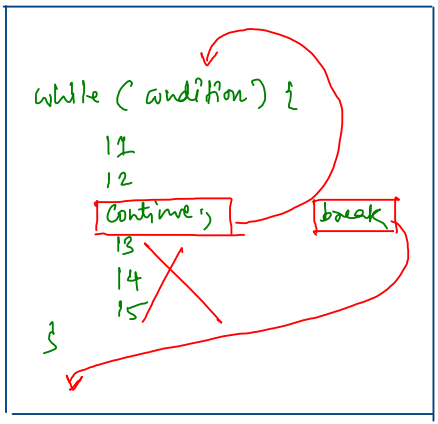
while all 3 thing are separately handled.

hence you have more flexibility to code

→ break, continue work as usual,

break - terminates while loop

continue - skip to next iteration



★ Sum of digits :

Eg : 12345

Op : 1+2+3+4+5
= 15

Eg : 1325

Op : 1+3+2+5 = 11

→ sum = 0+5+2+3+1
= 11

num = 1325

a = 1 c = 2
b = 3 d = 5

⇒ sum = a+b+c+d

★ how to extract each individual digit from the number.

1325 % 10 = 5

↓ 1325/10 = 132.5 = 132

132 % 10 = 2

↓ 132/10 = 13.2 = 13

13 % 10 = 3

↓ 13/10 = 1.3 = 1

1 % 10 = 1

↓ 1/10 = 0.1 = 0

0 stop the process

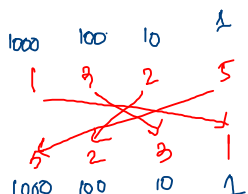
1. extract last digit
2. remove last digit
3. caclulation (sum += LD)

(Digit Extraction Technique)

★ Reverse Digits :

Eg : 1 9 2 5

op : 5 2 3 1



1925 % 10 = 5

↓ 1925/10

192 % 10 = 2

↓ 192/10

19 % 10 = 9

↓ 19/10

1 % 10 = 1

↓ 1/10

0 ✗

1. num % 10
2. num = num/10
3. Calculation
   ↳ rev = (rev * 10) + lastD

rev = 0

① rev = 0 × 10 + 5 = 5

② rev = 5 × 10 + 2 = 52

③ rev = 52 × 10 + 9 = 523

④ rev = 523 × 10 + 1 = 5231

```
466
467  // Reverse digits of a Number
468  // Link: https://course.acciojob.com/idle?question=817d51d8-e009-4322-8e51-257b76455a4c
469  readline.question("", (n) => {
470    //Write your code here
471    let rev = 0;
472    while (n != 0) {
473      // 1. extract last digit
474      const lastDigit = n % 10;
475
476      // 2. remove last digit
477      n = parseInt(n / 10);
478
479      // 3. calculation
480      rev = rev * 10 + lastDigit;
481    }
482
483    console.log(rev);
484
485    readline.close();
486  });
487
```

1325 , rev = 0 5 52 523 5231

① 1325 != 0
→ LD = 1325 % 10 = 5
→ n = 1325/10 = 132
→ rev = 0×10 + 5 = 5

② 132 != 0
→ LD = 132 % 10 = 2
→ n = 132/10 = 13
→ rev = 5×10 + 2 = 52

③ 13 != 0
→ LD = 13 % 10 = 3
→ n = 13/10 = 1
→ rev = 52×10 + 3 = 523

④ 1 != 0
→ LD = 1 % 10 = 1
→ n = 1/10 = 0
→ rev = 523×10 + 1 = 5231

⑤ 0 != 0 X

```
while(condition) {
    const lastDigit = n % 10;
    /* code */
}
```

eg: 1325

① LD = 1325 % 10 = 5
② LD = 132 % 10 = 2
③ LD = 13 % 10 = 3
④ LD = 1 % 10 = 1

} changing every time the how const works ?

★ Why const for last digit or it is changing every time? (How does it work)

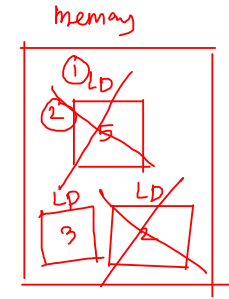
① const LD = n % 10;

After 1st Iteration is done and the memory cracked true is removed

② Const LD = n % 10;

③

⇒ same in any loop.



★ We create variable wherever necessary. (general practice)

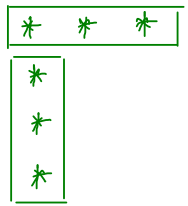
* N stars :

```
for(let i=0; i<N; i++) {
    process.stdout.write("*");
}
console.log(); // dummy
for(let i=0; i<N; i++) {
    console.log("*");
}
```

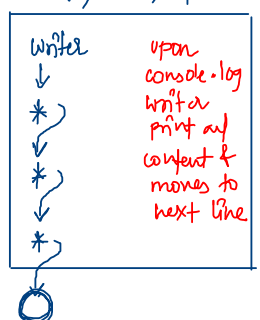
helps to move writer to next line

Ip: N = 3

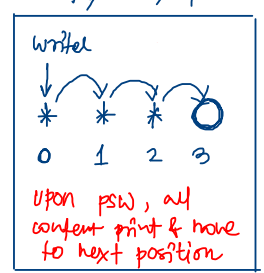
Op:



Notes/Console/Op

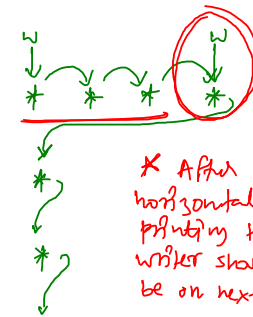


Notes/Console/Op



N = 3 i=0 X X

i=0 X X

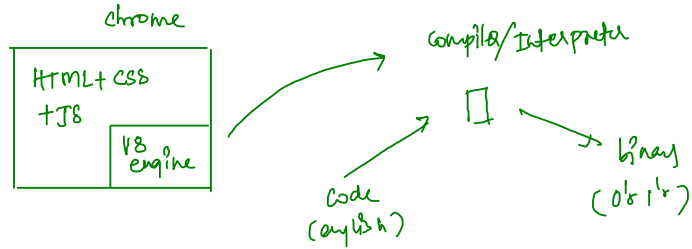


* After horizontal printing the writer should be on next line

Op: * * * *
* *

(wrong)

★ process . stdout . write :



C/C++	Java	python
mingw	JVM	python

⇒ Install compiler/software to run programs.

JS

→ Chrome is enough to run the JS code

→ google chrome is acting as a compiler.

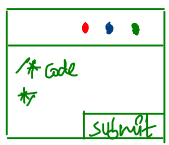
(Frontend)

API call

(some computer over Internet)

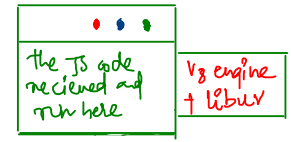
online IDE

Backend server



JS code

Request



Response

output

(USA)

(India)

⇒ this goes through multiple connections over Internet (using routers)

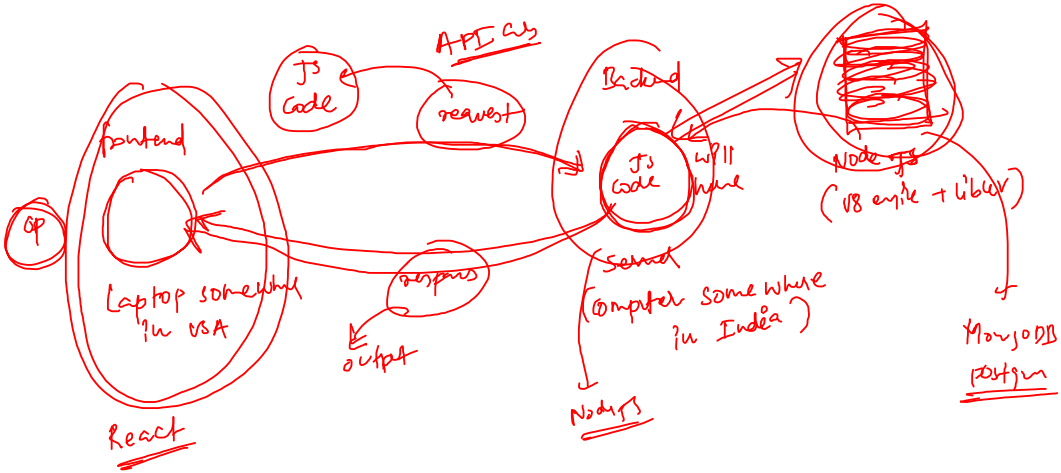
★ process.stdout.write is a part of libuv

⇒ As chrome doesn't have libuv it cannot understand PSW.

how will Backend Server run JS code?

⇒ Node.js provides the V8 engine

Node.JS is a runtime environment for JS (a playground for JS).


API call
JS Code
request
frontend
Backend
JS Code
V8 Engine
Node JS
(V8 engine + libUV)
OP
Laptop somewhere in USA
response
output
Server
(Computer somewhere in India)
MongoDB
postgres
NodeJS
React