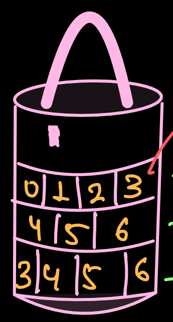
Complete Beginner
While loop-1

Question: Open the lock:

Given that you have a number lock. it contains 3 strips of number from 0 to 9. write a code for opening this lock. [NOTE: we don't the password]



Every strips hold digits from 0 to 9

→ Strip No. 1

→ Strip No. 2

→ Strip No. 3

try all combinations

(i) from 000 to 999

000 → 0
001 → 1
002 → 2
⋮
010 → 10
⋮

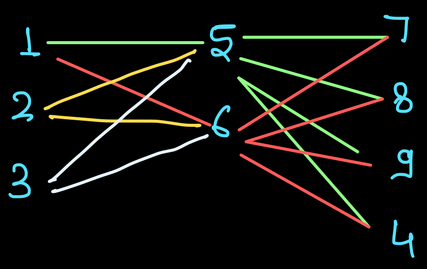
100 → 100
101 → 101
⋮
200 → 200
⋮

900 → 900
901 → 901
902 → 902
⋮
999 → 999

Assumption

000 → 0
001 → 1
002 → 2
⋮
010 → 10
⋮
099 → 99
100 → 100
⋮
999 → 999

	B-1	B-2	B-3
	□	□	□
Strip digit	1, 2, 3	5, 6	7, 8, 9, 4
Replace	a, b, c	d, e	f, g, h, i



157
158
159
154
167
168
169
164
257
258
259
254

267
268
269
264
357
358
359
354
367
368
369
364

Count of Total combination = 3 × 2 × 4

= 24

General Combination

B1	B2	B3
a	b	c

$$\boxed{\text{Total Combination} = a \times b \times c}$$

Strip 1	strips 2	strips 3
0 – 9	0-9	0 – 9
10 opt's	10 option	10 options

Total combination = 10 x 10 x 10

= 1000

options are → 000 to 999

```
if ( still lock ) {
    test combination 000
} else if (still lock) {
    test combination 001
} else if (still lock) {
    test combination 002
} else if (still lock) {
    test combination 003
} . . . and so on . . .
```

How many time we have to write this if-else ? → 1000 time

Is this a good way of writing a code ?

→ No

Is there is any construct provided by language ?

↳ called **loop**

```
combination = 0 // → 000
while ( lock is closed )
    ✓ test combination;
    ✓ combination = combination + 1
}
```

10 → 010

000
↓
001

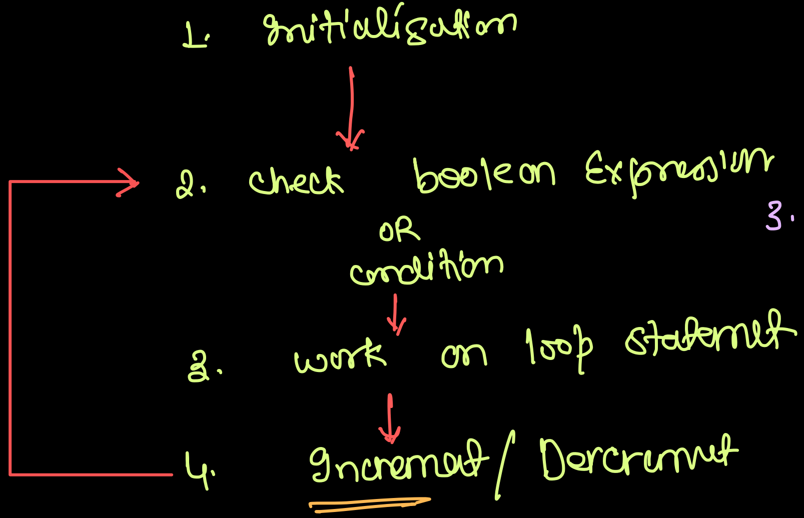
Assumption = 128 password

Combination	lock status	Comb.= com. + 1
0	closed	1
1	closed	2
2	closed	3
3	closed	4
⋮		
127	close	128
128	closed	129
129	open.	

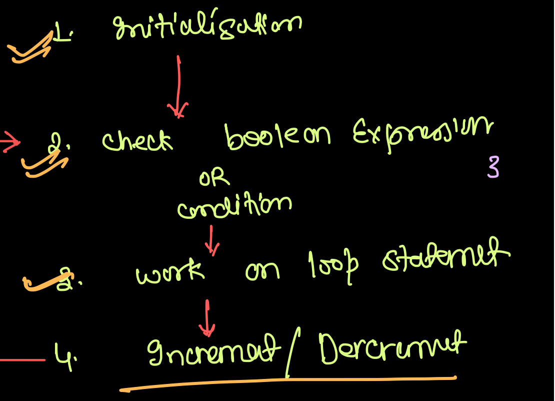
Syntax of while loop →

```
// 1. Initialisation
while (2. Expression / condition) {
    3. loop Statement / task
    4. Update → Increment / Decrement
```

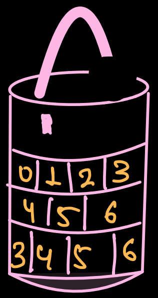
order of Execution →


1. Initialisation
2. check boolean Expression OR condition 3.
3. work on loop statement
4. Increment / Decrement

```
combination = 0
while (lock is closed)
    test combination;
    combination = combination + 1
}
```


1. Initialisation
2. check boolean Expression OR condition 3
3. work on loop statement
4. Increment / Decrement

Assumption = pass = 128



combination :	lock Status	Testing.	comb = Comb + 1
0	closed	✓	1
1	closed.	✓	2
2	closed	✓	3
3	Closed	✓	4
⋮			
127	closed	✓	128
128	closed	✓ pass this	129
129	open.		

Problem! Print Number from 1 to 10;

↳ target

```
int   i= 1;             i<11
while ( i <=10) {
    System.out.println( i);
    i= i+1;             i++;  ++1
}                       i+=1;
```

Steps to write a loop

✓ 1. Initialisation

✓ 2. Condition.

✓ 3. loop statement , work

✓ 4. Update → Increment / Decrement

i= ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~ ~~7~~ ~~8~~ ~~9~~

~~10~~ 11

```
1    5    9
2    6    10
3    7
4    8
```

var= i++;

var= ++i;

i++;

OR

++i;

Pre Incrementation

Post Incrementation.

```
i++  ⇒  i= i+1;
         i+=1

++i
```

```
int a = 10;          a = ~~10~~ 11
int b = a++;         b= 10 ] → assigned then Incremeted
SOP(a). → 11
```

```
a = ~~10~~ 11
b = 10
```

SOP(a++)

```
int a = 10;          a = ~~10~~ 11
int b = ++a;         b = 11 ] → Increment then assigned.
```

```
boolean x, y, z

x = y = z = true
                          false → false
     false        false       fals
if( !x || ( !y && !z )){
    S.O.P.("why");
}else {
    S.O.P.("what");
}
                 → what
```

x = true

y = true

z = true

!x = false

!y = false

!z = false

sum.

```
{
    int num1.1
    int num2;
    sum= num1+n.
}
{ int num1 :
```