Basic way to find LCM of 2 numbers [BRUTE FORCE APPROACH]

eg: 2 and 5

Multiplers of 2 ⇒ [2 4 6 8 (10) 12 14 16 18 (20) .....]   LCM ⇒ Least Common Multiple
Multiplers of 5 ⇒ [5 (10) 15 (20) 25 30 35 40 45 50 ---]

NOTE

* We need to find largest number

10 and 20 are common but the least one was 10
∴ LCM of 2 and 5 is (10)

Reason

We are searching common multiplers of both given numbers (in this case ⇒ 2 & 5)

(2, 3, 4) ⇒ will not comes in the multiplers of "5" ⇒ NOT POSSIBLE to find common factor

2 ⇒ Least number
(5 ⇒ Largest number) ⇒ We can find the common multiplers of 2 & 5 only after "5"
(i.e) only after largest number

Hence
    step 1: FIND THE LARGEST NUMBER ....

LOGIC

```
int res = max(a, b);
while (true)  // Infinite loop
{
    if (res%a == 0 && res%b == 0) // checking whether "res" is a multiple of both
    {                                                                'a' and 'b'
        System.out.print(res);
        break;
    }
    res++;
}
```

if the condition NOT TRUE, increment the value of res.

HCF (or) GCD

* HCF ⇒ Highest Common [Factor]
* GCD ⇒ Greatest Common [Divisor]

Factor (or) Divisor
Any digit that can perfectly divide the given number is known as divisor or a factor.

Two numbers ⇒ 5 and 10

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

Factors / Divisor

5 → 1 5
10 → 1 2 5 10

Common factors

5 → 1 [5]
10 → 1 [5]   ⇒ 5 is the highest common factor
             ∴ HCF of 5 and 10 is (5)

NOTE

Any number that is > 5 ⇒ will not present in the common factor of given number

5 → Least Number
10 → Largest Number

Factors
5 → 1 5
10 → 1 2 5 | 10   ⇒ Here we don't want to check for the factor which is greater than the least number (i.e ⇒ 5 (here)).

REASON

We are looking for "COMMON FACTORS"

There will be no common factor present greater than the least number (5)

(i.e)   5 → 1, 5   (and) ⇒ No chances for 6, 7, 8, 9, (10) (here)
         10 → 1, 2, 5, (10) (No need to check, factor which was > least number).

∴ Step : 1 ⇒ FIND THE MINIMUM OF GIVEN TWO NUMBERS

Since we are going to find the (Highest) common Factor
↳ so that we are starting our iteration from the

LOGIC
```
int res = min (a,b);
for (int i=res; i>=1; i--)
{ if (a%i ==0 && b%i==0)
  { system.out.print (i);
    break;
  }
}
```

Step 2 :

(min (a,b) ——→1)

⇒ "BRUTE FORCE APPROACH"

DRAWBACK

a=173 , (b=133) ⇒ Min

No common divisor blw them

∴ HCF = 1

But it will take
133 iterations to get that
answer.

Euclid's Algorithm

Steps
* Find the difference blw given numbers
* Replace that difference with larger number
* Continue this steps until both numbers become equal.

① a=15, b=10
* (15) − 10 ⇒ GCD = 5
                D = 5

② a=5, b =10
   10 − 5 ⇒ GCD = 5
            D = 5

③ a=5 , b=5 ⇒ Both numbers becomes equal (Stop the process) and GCD of (15,10) = ⑤.

LOGIC
```
while (a != b) {
  if (a>b){ a= a-b;}  ⇒ a is greater, find diff, assign it to 'a' (Longest number)
  else { b=b-a;}  ⇒ b is greater, find diff, assign it to 'b' (Longest number)
}
system.out.print (a); // we can print either `a' or 'b' ⇒ Because (a ==b),
```

Modified Euclid's Algorithm

Steps
* Find the larger number.
* Do [larger number % smaller number]
* Replace the result with larger number
* Replace this until any of the given number becomes zero.

∴ GCD of (20, 15) ⇒ ⑤,

a [20]   b [15]

20 % 15 ⇒ 5

a [5]   b [15]

15 % 5 ⇒ 0

(a [5]   b [0])

↓
`b' becomes zero
∴ `a' is answer

LOGIC
```
while (a!= 0 && b!=0)
{
  if (a>b) { a= a%b;}
  else { b= b%a;}
}
if (a !=0) system.out.print (a);
else system.out.print (b);
```

To find LCM in the efficient way

LCM * GCD = Product of two numbers (a*b)

∴ $LCM = \dfrac{a*b}{GCD}$

⇒ To find GCD we modified Euclid's algorithm

LCM = (a*b) / (modifiedEuclidAlgorithm (a,b));
                              ↓↓
                          Function