Name Monal Sharma, Uid 23BCS10366, Class KRG-1B

Ques → A [illegible] integer is magical [illegible] when [illegible] is divisible by either a or b.

Given 3 integers n, a & b, return the n^{th} magical no. [illegible]

TC → 1 ≤ n ≤ 10^9, 1 ≤ a ≤ 10^4, 1 ≤ b ≤ 10^4

1) n = 1, a = 2, b = 3
   o/p = 2

2) n = 4, a = 2, b = 3
   o/p = 6

3) a = 2, b = 3, n = 5
   o/p = 8

```
→ int a,b,n;
  [illegible] int cnt = min(a,b);
while(n){
    if(cnt % a == 0 || cnt % b == 0)
        cnt++;
    n--;
```

Due to large constraints of n, the value can overflow

```
So  int cnt = min(a,b); if(a == b) return n * a;
→ while(n){
      if(a <= b)
      {
         if(a == b)
            a += a;
         else{
            n--;
            a += a;
         }
      }
      else {
         n--;
         b += b;
      }
  }
  return min(a,b);
```

LCM × GCD = a × b

LCM = (a × b) / GCD

LCM [illegible] = n × [illegible]

[illegible] = [illegible] [1-1] + [1]

[illegible]

Approach :-

1) Find gcd(a, b)
2) use GCD to find LCM of a, b
3) Run binary search on 1 to $n \times min(a, b)$
4) Calculate mid & for mid calculate count

$$count = \frac{mid}{a} + \frac{mid}{b} + \frac{mid}{lcm(a, b)}$$

5) If req'd count is less, increase search space by low = mid + 1
   else high = mid - 1.
6) If ans is big calc using mod 1e9+7.