Q1 Square root of +ve integer biggest x st $x*x \le N$

25 → 5 47 → 6

20 → 4

Idea Binary search l = 1 h = N

mid * mid $\le$ N ans = mid, l = mid + 1

$>$ N h = mid − 1

Dry run

N = 47

mid $\le \frac{N}{mid}$

l	h	m	
1	47	24	24×24 > 47
1	23	12	12×12 > 47
1	11	6	6×6 ≤ 47 ans = 6
7	11	9	9×9 > 47
7	8	7	7×7 > 47
7	6	STOP!!	

mid * mid $\le$ N

mid $\le \frac{N}{mid}$

```
int sqrt (int N) {
    l = 1        h = n
    while (l ≤ h) {
        mid = (l+h)/2
        if (mid ≤ N/mid) {
            ans = mid
            l = mid+1
        }
        else
            h = mid-1
    }
    return ans
}
```

TC : $O(\log n)$
SC : $O(1)$

4 5 6 1 2 3

4 5 6 7 0 1 2

02 Search in Rotated sorted array

unique values

```
   0  1  2  3  4  5  6
Eg 4  5  6  7  0  1  2        k = 0
                              ans = 4

   0  1  2  3  4  5
Eg2 4  5  6  7  0  1          k = 100
                              ans = -1
```

Idea Modify BS algo. But how?

Need to know whether to go left or right

```
   0  1  2  3  4  5  6
   4  5  6  7  0  1  2
```

all yellow > all green

⟹ 2 sorted parts.

2 cases

⇒ target < ar[0] target is green

if nums[mid] >= nums[0] go right

else

if (nums[mid] < target) go right

else go left

target > arr[0] target is yellow
if nums[mid] < nums[0] go left
else
if (nums[mid] < target) go right
else go left

Code

```
int search ( int arr[], int k ) {
  l = 0   h = n-1
  while ( l <= h ) {
    m = (l+h)/2
    if ( arr[m] == k )
        return m
    if (target < arr[0])                    target is green
        if nums[mid] >= nums[0]             l = m+1
        else
            if ( nums[mid] < target )       l = m+1
            else                            h = m-1

    else {                                  target is yellow
        if nums[mid] < nums[0]              h = m-1
        else
            if ( nums[mid] < target )       l = m+1
            else                            h = m-1
    }
```

```
}
return -1
}
```

TC: O(logn)
SC: O(1)

0	1	2	3	4	5	6	7
70	0	10	20	30	40	50	60

k = 21

l	h	m	
0	7	3	
4	7	5	
4	4	4	
4	3	STOP!!!	return -1

k = 10

l	h	m	
0	7	3	
0	2	1	
2	2	2	a[2] = 10 return 2

Tough ques.

03 Median of 2 sorted arrays

Given 2 sorted array, find median (middle element) of the merged array. If merged array is even length, then return average of 2 middle elements

Eg 1 A = [1, 3] B = [2] merged = 1, 2, 3 (0 1 2)

ans = 2

Eg 2 A = [1, 2] B = [3, 4] merged = 1, 2, 3, 4 (0 1 2 3)

ans = 2.5

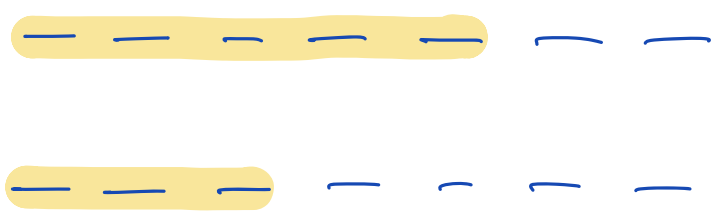
25 ⇒ 10 15

Brute: Merge the 2 sorted arrays & get answer.

TC: $O(n+m)$

We want answer in logarithmic TC

Idea



Some elem of A & some of B will be part of first half of merged array

Now total size = $n_1 + n_2$

$\Rightarrow$ people in first half = $\frac{n_1+n_2+1}{2}$

1 2 3 4 5 6

So, if x elem from A,

$\frac{n_1+n_2+1}{2} - x$ elem from B

1 2 3 4 5 6 7

min_1 max_1

- A = 3 4 5 6 7
- B = 2 5 7 8 10

min_2 max_2

How to verify this

$min_1 \leq max_2$ && $min_2 \leq max_1$

Now if even total size

$(max(min_1, min_2) + min(max_1, max_2))/2$

else $max(min_1, min_2)$

else if $min_1 > max_2$

$h = mid - 1$

else

$l = mid + 1$

Code

```
double median ( int A[], int B[] ){
  n1 = A.size()      n2 = B.size()

  l = 0              h = n1

  while (l <= h) {
     mid = (l+h)/2
     count1 = mid
     count2 = (n1+n2+1)/2 - count1

  {
    int min1, max1, min2, max2

    if ( count1 == 0)
          min1 = INT_MIN
    else
          min1 = A[count1 - 1]

    if ( count2 == 0)
          min2 = INT_MIN
    else
          min2 = B[count2 - 1]
```

```
{
    if (count1 == n1)
        max1 = INT_MAX
    else
        max1 = A[count1]

    if (count2 == n2)
        max2 = INT_MAX
    else
        max2 = B[count2]

    if (min1 <= max2 && min2 <= max1) {
        if ((n1+n2) % 2 == 0)
            return [max(min1, min2) +
                    min(max1, max2)] / 2.0
        else
            return max(min1, min2)
    }

    else if (min1 > max2)
        h = mid - 1
    else
        l = mid + 1
}
```

TC: $\log(n_1)$

SC: $O(1)$

Dry run

$4+3+1$

$\frac{n_1+n_2+1}{2} = 4$

	0	1	2	3	
A =	1	3	5	7	x
B =	2	4	6		$4-x$

1 2 3 4 5 6 7

l	h	m	
0	4	2	$\Rightarrow$ ans = 4

	0	1	2	3	
A =	7	10	11	15	x
B =	10	12	14	20	$4-x$

l	h	m
0	4	2
3	4	3

$\frac{11+12}{2} = 11.5$

7 10 10 11 12 14 15 20

4

$\frac{12 \times 9}{3}$

12 9

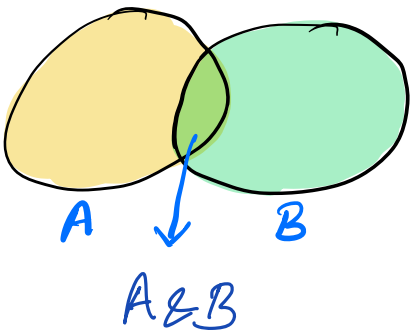
LCM = 36

01) LCM of a, b

$$\frac{a \times b}{gcd(a,b)}$$

02) Venn diagram

A or B = A + B − A and B



03 No of multiples of a in [1:b]

a 2a 3a 4a ----- b]

b = 14

a = 3 1 2 3 4 5 6 7 8 9 10 11 12 13 14

⇒ ans = 4 ⇒ $\frac{b}{a}$

$ka \le b$

a 2a 3a ---- ka

$k \le b/a$

04 No of multiples of B or C in [1:A]

⇒ ans(B) + ans(C) − ans(B and C)

$a/b + a/c - a/lcm(b,c)$

Qs Given A, B, C find A^{th} magical no.

number is magical if div by B or C

Eg 1 A = 8 B = 2 C = 3

1 2 3 4 5 6 7 8 9 10 11 12

ans = 12

Eg 2 A = 5 B = 4 C = 6

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

ans = 16

Brute: Loop from 1 till you get A^{th} magical numbers.

Idea Binary Search

A^{th} magical numbers

l = 1 h = A × min(B, C)

A = 10 B = 4 C = 6

Is 36 10^{th} magical numbers ?

36/4 + 36/6 − 36/lcm(4,6) = 12

12 > 10

go left

Is 24 10^{th} magical no

$24/4 + 24/6 - 24/lcm(4,6) = 8 < 10$

go right

21 22 23 24 ---- ... 34 35 36

l	h	m
1	40	20
21	40	30
21	29	25
26	29	27
28	29	28
29	29	29
30	29	stop

A=10 B=4 C=6

$\frac{29}{4} + \frac{29}{6} - \frac{29}{12} = 10$

ans = 30

ans = 30

Tdone

Code

```
int magical (int A, int B, int C) {
    lo = 1          hi = A x min(B, C)
    lcm = B*C / gcd(B, C)
    while( lo <= hi ) {
        int m = (lo+hi)/2
        int c = m/B + m/C - m/lcm
        if (c < A)
            l = m+1
        else if (c > A)
            h = m-1
        else {      // c == A
            ans = mid
            h = m-1
        }
    }
    return ans
}
```

1 ..... A x min(B,C)

TC: log(A x min(B,C))

SC: O(1)