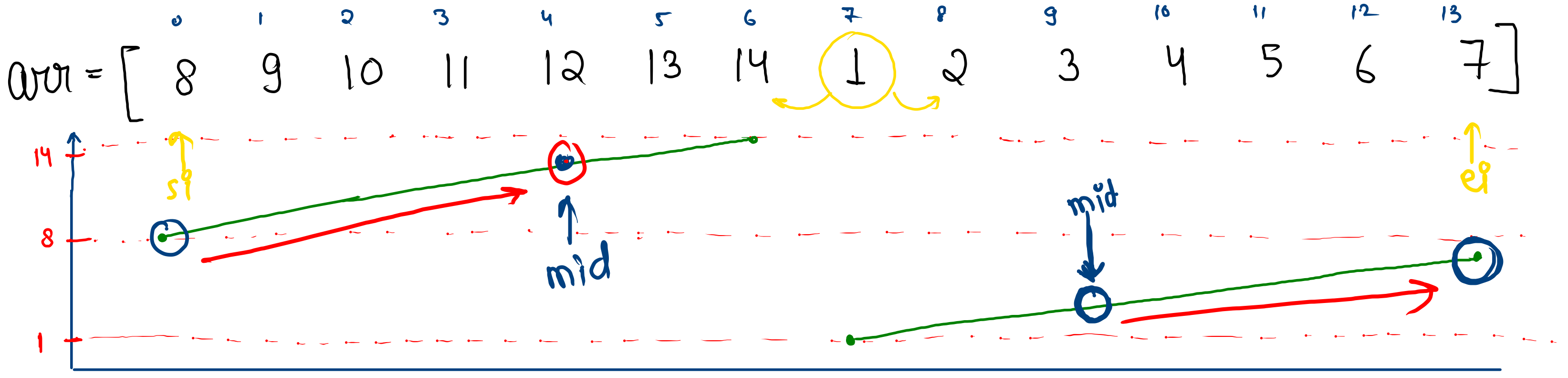
Find The Index of Rotation



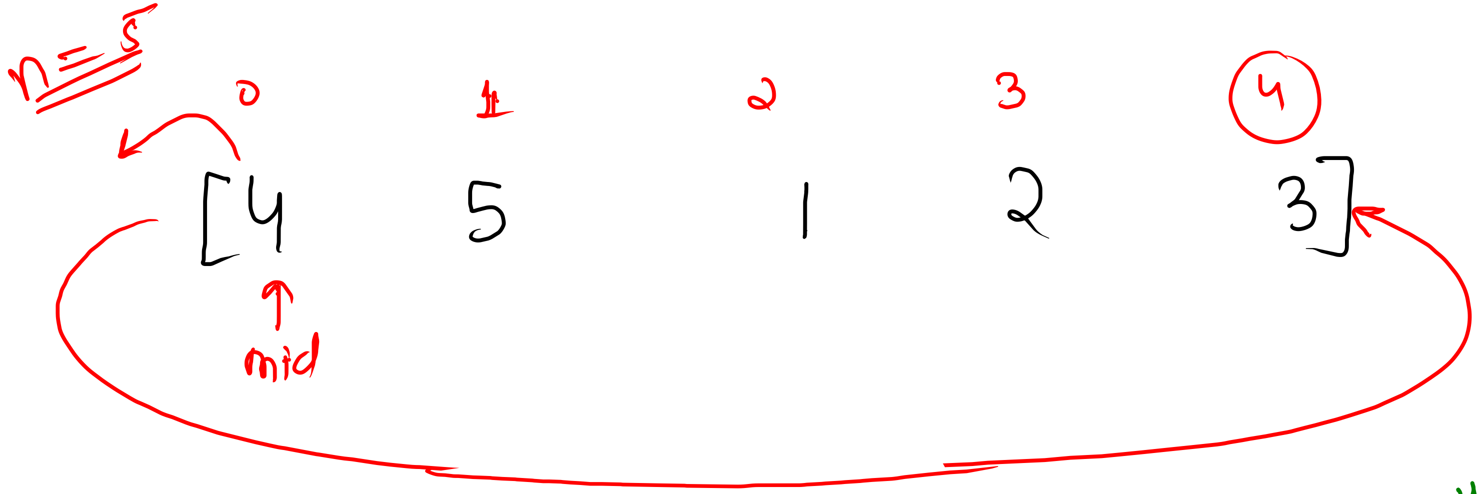
psudo code

```
if ( arr[mid] <= arr[prev] && arr[mid] <= arr[next])
        return (mid - 1);

else if ( arr[mid] <= arr[ei] )
        ei = mid - 1;          // shift range to left

else if ( arr[mid] >= arr[si] )
        si = mid + 1;          // shift range to right
```

Important



for rotation in forward dir :- $(x+1) \% \underbrace{n}_{}$ → length

(positive)

for rotation in backward dir :- $(x-1+n) \% \underbrace{n}_{}$

(negetive)

code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }

    binarySearch(arr, n);
}

public static void binarySearch(int[] arr, int n) {
    int si = 0;
    int ei = n - 1;
    while ( si <= ei ) {
        int mid = (si + ei) / 2;
        int prev = (mid - 1 + n) % n;
        int next = (mid + 1) % n;
        if ( arr[mid] <= arr[prev] && arr[mid] <= arr[next] ) {
            System.out.println( mid - 1 );
            return;
        } else if ( arr[mid] <= arr[ei] ) {
            ei = mid - 1;
        } else if ( arr[mid] >= arr[si] ) {
            si = mid + 1;
        }
    }
    System.out.println("Nothing");
}
```

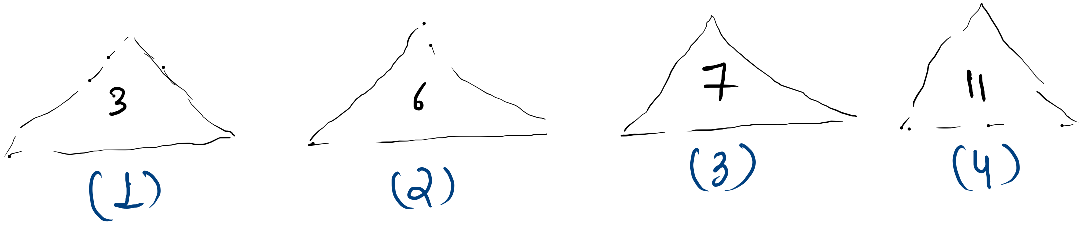
The banana challenge

$n = 4$

0	1	2	3
3	6	7	11

totalHours

$\boxed{h = 8}$ ✓



$\Rightarrow$ currHours

10 hrs

-ve   +ve

$K = \boxed{3}$ banans / hour

(we have to find K)

(speed of eating bananas)

$\frac{3}{3} = 1 \qquad \frac{6}{3} = 2 \qquad \frac{7}{3} = 2+1 \qquad \frac{11}{3} = 3+1$

0	1	2	3
3	6	7	11

mini = 1 // at least she will eat 1 b per hour

maxi = 11 // at most she can eat 11 per hour

(max of array)

$d = s \times t$

$$\boxed{t = \frac{d}{s}}$$

```java
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            arr[i] = scn.nextInt();
        }
        int h = scn.nextInt();

        kokoEatingBananas(arr, n, h);
    }

    public static void kokoEatingBananas(int[] arr, int n, int totalHours) {
        int mini = 1;
        int maxi = Integer.MIN_VALUE;
        for (int i = 0; i < arr.length; i++) {
            maxi = Math.max(arr[i], maxi);
        }

        int si = mini;
        int ei = maxi;
        while ( si <= ei ) {
            int mid = (si + ei) / 2;    // mid is speed of eating bananas
            if ( check( arr, mid, totalHours ) == true ) {
                ei = mid - 1;        // decreasing the speed
            } else {
                si = mid + 1;        // increasing the speed
            }
        }
        System.out.println(si);
        return;
    }

        public static boolean check(int[] arr, int currSpeed, int limit) { // IDENTIFY IF KOKO IS ABLE TO EAT ALL THE BANANAS
    IN "MID" HOURS

            int totalHours = 0;
            for (int i = 0; i < arr.length; i++) {
                int time = arr[i] / currSpeed;
                if ( arr[i] % currSpeed != 0 ) {
                    time++;
                }
                totalHours += time;
            }

            if (totalHours <= limit) {
                return true;
            } else {
                return false;
            }
        }
    }
```

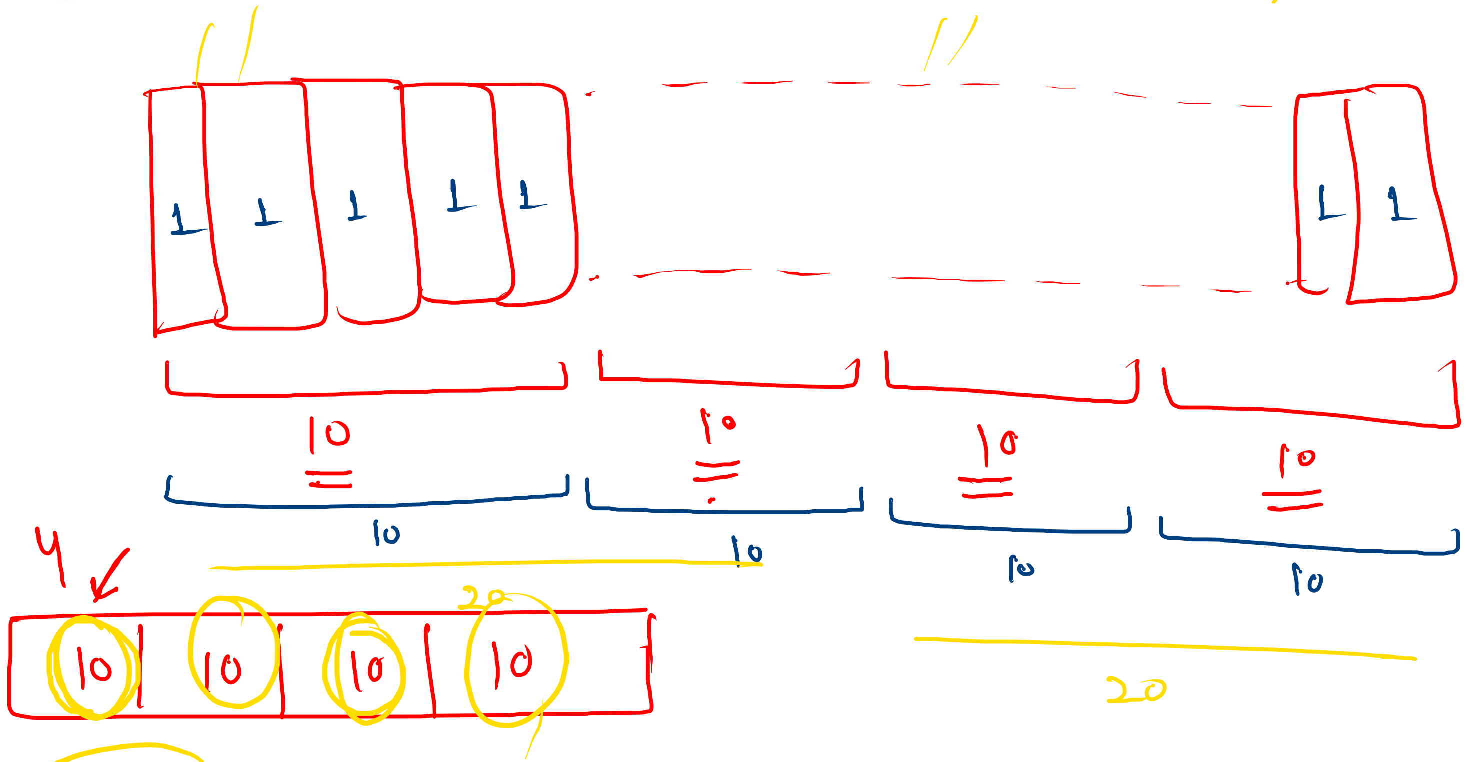
$\boxed{K}$

$$T.C = O(\log(N) * N)$$

$$= O(N \log N)$$

The painter

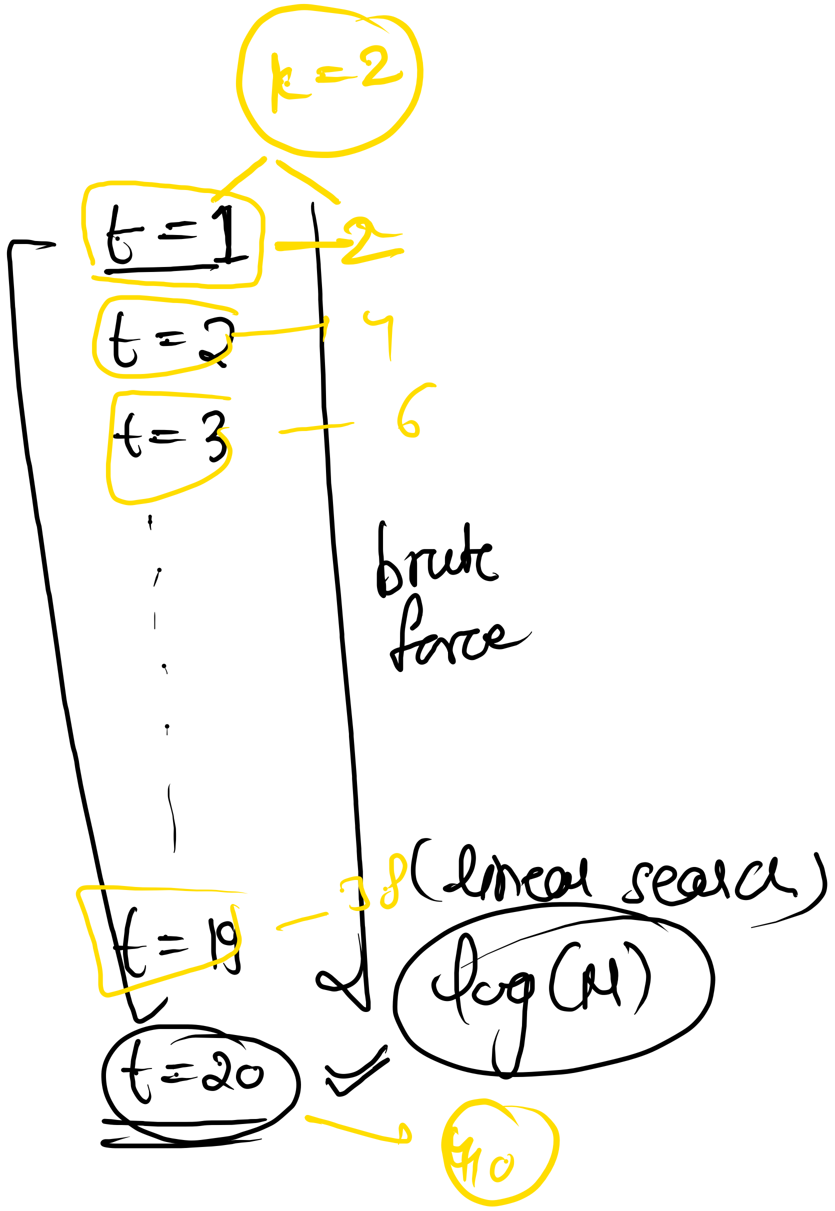
$k = \textcircled{2}$



$\boxed{k = 2}$

20 hrs ←

$$\text{time} = 10 + 10 = \underline{\underline{20}}$$



$\cancel{11 \% 3} = \boxed{1}$

$1 \% 3 = 1$

$2 \% 3 = 2$

$3 \% 3 = 0$

$4 \% 3 = 1$

$5 \% 3 = 2$

$6 \% 3 = 0$