30 Days Leetcode challenge - Day 17

public class leetday17

{

class Solution1 {

public int findPeakElement(int[] arr) {

int n = arr.length;

if(n==1)

{

return 0 ;

}

if(arr[0]>arr[1])

{

return 0;

}

if(arr[n-1]>arr[n-2])

{

return n-1;

}

int low = 1;

int high = n-2;

while(low<=high)

{

int mid = (low+high)/2;

if(arr[mid]>arr[mid+1] && arr[mid]>arr[mid-1])

{

return mid;// program will always end here

}

else if(arr[mid]>arr[mid-1])

{

low = mid +1;

}

else

{

high = mid -1;

}

}

return -1; /\* program will never come to this point but

the compiler wont accept a return type inside a conditional statement

so we use -1 just because it has a integer return type\*/

}

}

class Solution2 {

public int minEatingSpeed(int[] arr, int limit) {

int low = 1;

int high = getMax(arr);

int ans = high;

while(low<=high)

{

int mid = low + (high - low) / 2;

long bananaPerHour = findHours(arr,mid); //6

if(bananaPerHour<=limit)

{

ans = mid;

high = mid -1;

}

else

{

low = mid +1;

}

}

return ans;

}

public static long findHours(int arr[],int k)

{

long hours = 0;

for(int i=0;i<arr.length;i++)

{

hours += (arr[i] + k - 1L) / k;

}

return hours;

}

public static int getMax(int arr[])

{

int max = arr[0];

for(int i=0;i<arr.length;i++)

{

if(arr[i]>max)

{

max = arr[i];

}

}

return max;

}

}

}