CSE525 Monsoon 2020
Homework 2 Problem a

Varun Singhal (MT20099)
Sachin Kotiyal (MT20093)

You are given an array $H[1 \ldots n]$ of distinct integers. Refer to Problem-1 of JE-CS374-SP2018A-HW4 [1].

(a) Write a recursive $O(n \log n)$ algorithm to compute the arrays $L[1 \ldots n]$ and $R[1 \ldots n]$.

Solution: The algorithm is to find index of the closest taller hero in both side for all hero i.e H[1...n] and store that indices in L[1...n] and R[1...n].
Step1. Divide the Hero height array i.e H[start...end] Recursively into two halves until it can no more be divided.
Step2. While merge call 'LeftMax' function that will update the left closed taller hero index in array L[start...end].
Step3. While merge call 'RightMax' function that will update the right closed taller hero index in array R[start...end].

```
⟨⟨H height of all n heroes.⟩⟩
⟨⟨L is for storing the index of left side Closest taller hero for every hero in H⟩⟩
⟨⟨R is for storing the index of right side closest taller hero for every hero in H⟩⟩

def WhoTargetsWhom(H[1...n],L[1...n],R[1...n],start,end):
   if start = end :          return
   mid = start + (end-start)/2;
   WhoTargetsWhom(H,L,R,start,mid);
   WhoTargetsWhom(H,L,R,mid+1,end);
   LeftMax(H,L,start,mid,end);
   RightMax(H,R,start,mid,end);

⟨⟨Storing index of left closest taller hero for hero array H[start...mid...end]⟩⟩
def LeftMax(H[1...n],int L[1...n],start,mid,end):
   i = mid        j = mid + 1
   while i >= start  &  j <= end:
        if L[j] = −1:
             if H[i] > H[j]:    L[j++] = i
             else:      i−−
        else:      j++
   return

⟨⟨Storing index of right closest taller hero for hero array H[start...mid...end]⟩⟩
def RightMax(H[1...n],R[1...n],start,mid,end):
   i = mid        j = mid + 1
   while i >= start  &  j <= end:
        if R[i] = −1:
             if H[j] > H[i]:    R[i−−] = j
             else:      j++
        else:      i−−
   return
```

$T(n) = 2T(n/2) + O(2n)$. ⟨⟨Recurrence Relation⟩⟩
Using Master Theorem
$T(n) = \Theta(nLog(n))$

■

[1] https://courses.engr.illinois.edu/cs374/sp2018/A/homework/hw4.pdf

CSE525 Monsoon 2020 Varun Singhal (MT20099)
Homework 2 Problem b Sachin Kotiyal (MT20093)

(b) Prove that at least $\lfloor n/2 \rfloor$ positions will be chosen as a "target".

Solution: $\langle\langle H$ *height of all 'n' heroes which are distinct.* $\rangle\rangle$
$\langle\langle L$ *is for storing the index of left side Closest taller hero for every hero in* $H\rangle\rangle$
$\langle\langle R$ *is for storing the index of right site closest taller hero for every hero in* $H\rangle\rangle$

We know that
If n=1 Then 0 Distinct value in L,R array.
If n=2 Then at least 1 Distinct value in L,R array.
If n=3 Then at least 1 Distinct value in L,R array.

For n>1:
If n is EVEN and If I divide the height hero array 'H' in group of 2-2 Heroes i.e n/2 group each having two Distinct value.
Then at Least n/2 distinct value in L,R array.

If n is ODD and If I divide the height hero array 'H' in group of 2-2 Hero and 1 group of 3 Hero.
Then at least (n-3)/2 + 1 = $\frac{n-1}{2}$ Distinct value in L,R array.

Hence I can say that for size 'n' Height of hero Then Distinct value in L,R array is $\lfloor n/2 \rfloor$.

Hence Proved

■

CSE525 Monsoon 2020 Varun Singhal (MT20099)
Homework 2 Problem c Sachin Kotiyal (MT20093)

(c) Design and analyse a recursive algorithm to output the number of rounds to identify the smallest number in the linked list by repeatedly removing the targets in the manner specified in the above link.

Solution: Algorithm is find the no of rounds before Dr. Metaphor's deadly process finally ends by Recursively call 'CountRound' function.
Step1. CountRound function call recursively with update Hero array 'H' and count the no of rounds that will return at the end.
Step2. For Shoot Adjacent taller hero simultaneously call 'DeleteAdjacentTall' function and update the Hero height array with new Hero height array which are left in current round.

```
⟨⟨H height of all n heroes.⟩⟩
⟨⟨'CountRound' Recursive function used to find no of round before deadly end ⟩⟩
⟨⟨pass NoOFRound var with initial value 0⟩⟩

def CountRound(H[1...n],NoOFRound):
  if H.size = 1 :
        return NoOFRound
  DeleteAdjacentTall(H[1...n])
  conntRound(H,NoOfRound++)

⟨⟨Function will delete the adjacent tall hero in H[1...n]⟩⟩
def DeleteAdjacentTall(H[1...n]):
  Let T[1...n] with initial all value are '0'
  if H[1] < H[2] ⟨⟨Boundary Conditions⟩⟩
      T[2]=1
  if H[n-1] < H[n] ⟨⟨Boundary Condition⟩⟩
      T[n]=1

  for i = 2 ; i < n ; i++:
      if H[i-1] > H[i]: T[i-1] = 1
      if H[i+1] > H[i]: T[i+1] = 1
  ⟨⟨Create 'NewHero' array contain hero from H array for which T[i]=0⟩⟩
  Let NewHero[] array
  k=1
  for i = 1 ; i <= n ; i++:
      if T[i] = 0:
          NewHero[k++]=H[i];
  ⟨⟨Upadte NewHero array to H⟩⟩
  H=Newarray
```

$T(n) = T(n/2) + 0(2n)$⟨⟨*Recurrence Relation*⟩⟩
$T(n) = 2n + n + \frac{n}{2} + \frac{n}{4} + ...log(n)terms$
$T(n) = n(2 + 1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + ...log(n)terms$
$T(n) = n\left(2 + \frac{1\left(1-\frac{1}{2}^{logn}\right)}{1-\frac{1}{2}}\right)$
$T(n) = n\left(2 + \frac{\left(1-\frac{1}{n}\right)}{\frac{1}{2}}\right)$
$T(n) = n(2 + 2 - \frac{2}{n})$
$T(n) = 4n - 2$
$T(n) = \Theta(4n)$

■