**采用了两种方法，第一种想法是比较朴素，操作类似于栈，采用额外数组存储可以看见的牛的数目，第二种借用了单调栈的概念，模拟了单调栈，并用坐标位置代表牛的数目；**

**两种方法都是O（n）级的算法，第一种占用空间较多。**

1. 没有使用单调栈的方式，思路是将牛的高度先读入保存，在**由后向前读入另一个数组**，若后读入的牛高于先前读入的牛，就将前面的牛出栈，并修改可以看到的牛数。

#include<stdlib.h>

#include<stdio.h>

#define N 80000

int main()

{

int cow[N],num[N],p[N];

int n,i,j;

long long sum=0;

scanf("%d",&n);

for(i=0;i<n;i++) scanf("%d",&p[i]);

for(i=n-1,j=0;i>=0;i--,j++)

{

cow[j]=p[i];

num[j]=0;

while(j&&cow[j-1]<cow[j])

{

cow[j-1]=cow[j];

num[j-1]=num[j-1]+num[j]+1;

cow[j]=0;

num[j--]=0;

}

sum+=num[j];

}

printf("%lld",sum);

return 0;

}

1. 使用了单调栈的方式，思路是先将牛的高度全部读入保存，在**从前向后读入另一个数组**中，若后读入的牛比先读入的矮，那么继续读入下一个元素，并将位置加入总数（位置信息代表了可以看见这头牛的牛的数目），若比先读入的高，那么将比读入元素矮的元素全部出栈，将读入元素入栈，并将位置加入总数；循环操作。

#include<stdlib.h>

#include<stdio.h>

#define N 80000

int main()

{

int cow[N]={0},p[N];

int n,i,j;

long long sum=0;

scanf("%d",&n);

for(i=0;i<n;i++) scanf("%d",&p[i]);

for(i=0,j=0;i<n;i++,j++)

{

while(j&&cow[j-1]<=p[i])

cow[--j]=0;

cow[j]=p[i];

sum+=j;

}

printf("%lld",sum);

return 0;

}