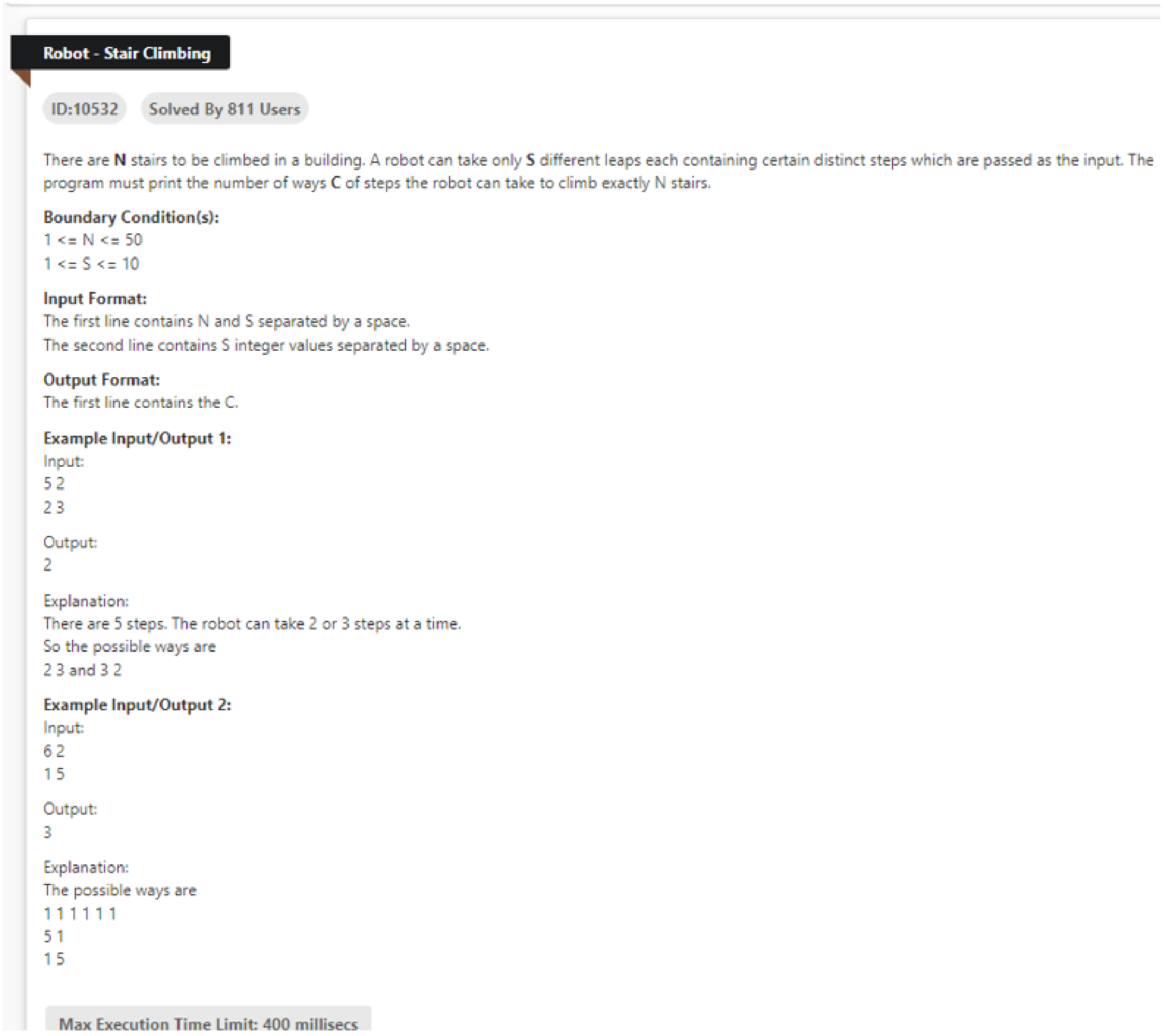
DAY 2:

SESSION 1:



Robot - Stair Climbing

ID:10532 Solved By 811 Users

There are **N** stairs to be climbed in a building. A robot can take only **S** different leaps each containing certain distinct steps which are passed as the input. The program must print the number of ways **C** of steps the robot can take to climb exactly N stairs.

Boundary Condition(s):
1 <= N <= 50
1 <= S <= 10

Input Format:
The first line contains N and S separated by a space.
The second line contains S integer values separated by a space.

Output Format:
The first line contains the C.

Example Input/Output 1:
Input:
5 2
2 3

Output:
2

Explanation:
There are 5 steps. The robot can take 2 or 3 steps at a time.
So the possible ways are
2 3 and 3 2

Example Input/Output 2:
Input:
6 2
1 5

Output:
3

Explanation:
The possible ways are
1 1 1 1 1 1
5 1
1 5

Max Execution Time Limit: 400 millisecs

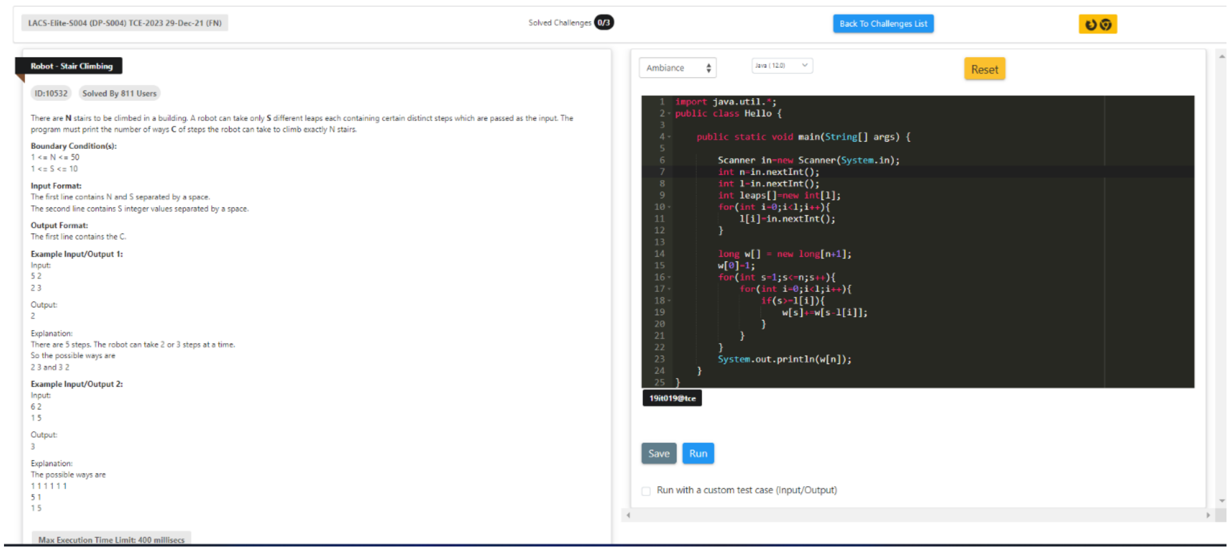
Visualization:

https://pythontutor.com/visualize.html#code=public%20class%20climbingstairs%7B%0A%20%20%20%20public%20static%20void%20main%28String%5B%5D%20args%29%20%7B%0A%0A%20%20%20%20%20%20%20%20int%20n%3D5%3B%0A%20%20%20%20%20%20%20%20int%20l%3D2%3B%0A%20%20%20%20%20%20%20%20int%20leaps%5B%5D%20%3D%7B2,3%7D%3B%0A%0A%20%20%20%20%20%20%20%20long%20ways%5B%5D%3Dnew%20long%5Bn%2B1%5D%3B%0A%20%20%20%20%20%20%20%20ways%5B0%5D%3D1%3B%0A%20%20%20%20%20%20%20%20for%28int%20step%3D1%3Bstep%3C%3Dn%3Bstep%2B%2B%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20for%28int%20i%

3D0%3Bi%3Cl%3Bi%2B%2B%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20if%28step%3E%3Dleaps%5Bi%5D%29%7B%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20ways%5Bstep%5D%2B%3Dways%5Bstep-leaps%5Bi%5D%5D%3B%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20%20%7D%0A%20%20%20%20%20%20%20%20System.out.println%28ways%5Bn%5D%29%3B%0A%20%20%20%20%7D%0A%7D&cumulative=false&curInstr=67&heapPrimitives=nevernest&mode=display&origin=opt-frontend.js&py=java&rawInputLstJSON=%5B%5D&textReferences=false

CODE:

```java
import java.util.Scanner;

class climbingstairs{
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n=in.nextInt();
        int l=in.nextInt();
        int leaps[] = new int[l];
        for(int i=0;i<l;i++){
            leaps[i]=in.nextInt();
        }
        long ways[]=new long[n+1];
        ways[0]=1;
        for(int step=1;step<=n;step++){
            for(int i=0;i<l;i++){
                if(step>=leaps[i]){
                    ways[step]+=ways[step-leaps[i]];
                }
            }
        }
        System.out.println(ways[n]);
    }
}
```



LACS-Elite-S004 (DP-S004) TCE-2023 29-Dec-21 (FN)

Solved Challenges 0/3

Robot - Stair Climbing

ID:10532 Solved By 811 Users

There are **N** stairs to be climbed in a building. A robot can take only **S** different leaps each containing certain distinct steps which are passed as the input. The program must print the number of ways **C** of steps the robot can take to climb exactly N stairs.

Boundary Condition(s):
1 <= N <= 50
1 <= S <= 10

Input Format:
The first line contains N and S separated by a space.
The second line contains S integer values separated by a space.

Output Format:
The first line contains the C.

Example Input/Output 1:
Input:
5 2
2 3

Output:
2

Explanation:
There are 5 steps. The robot can take 2 or 3 steps at a time.
So the possible ways are
2 3 and 3 2

Example Input/Output 2:
Input:
6 2
1 5

Output:
3

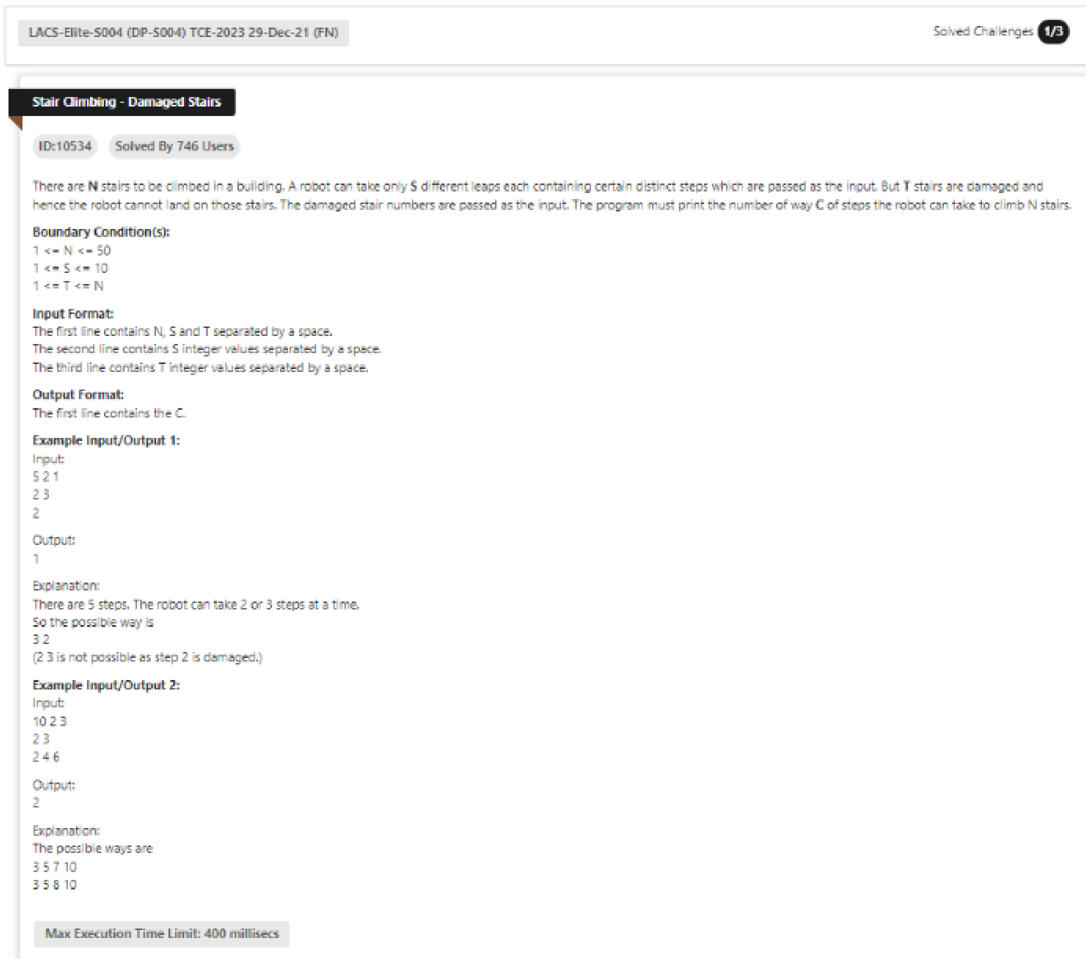
Explanation:
The possible ways are
1 1 1 1 1 1
5 1
1 5

Max Execution Time Limit: 400 millisecs

```java
import java.util.*;
public class Hello {

    public static void main(String[] args) {

        Scanner in=new Scanner(System.in);
        int n=in.nextInt();
        int l=in.nextInt();
        int leaps[]=new int[l];
        for(int i=0;i<l;i++){
            l[i]=in.nextInt();
        }

        long w[] = new long[n+1];
        w[0]=1;
        for(int s=1;s<=n;s++){
            for(int i=0;i<l;i++){
                if(s>=l[i]){
                    w[s]+=w[s-l[i]];
                }
            }
        }
        System.out.println(w[n]);
    }
}
```

Run with a custom test case (Input/Output)

Question 2:



LACS-Elite-S004 (DP-S004) TCE-2023 29-Dec-21 (FN)

Solved Challenges 1/3

Stair Climbing - Damaged Stairs

ID:10534 Solved By 746 Users

There are **N** stairs to be climbed in a building. A robot can take only **S** different leaps each containing certain distinct steps which are passed as the input. But **T** stairs are damaged and hence the robot cannot land on those stairs. The damaged stair numbers are passed as the input. The program must print the number of way **C** of steps the robot can take to climb N stairs.

Boundary Condition(s):
1 <= N <= 50
1 <= S <= 10
1 <= T <= N

Input Format:
The first line contains N, S and T separated by a space.
The second line contains S integer values separated by a space.
The third line contains T integer values separated by a space.

Output Format:
The first line contains the C.

Example Input/Output 1:
Input:
5 2 1
2 3
2

Output:
1

Explanation:
There are 5 steps. The robot can take 2 or 3 steps at a time.
So the possible way is
3 2
(2 3 is not possible as step 2 is damaged.)

Example Input/Output 2:
Input:
10 2 3
2 3
2 4 6

Output:
2

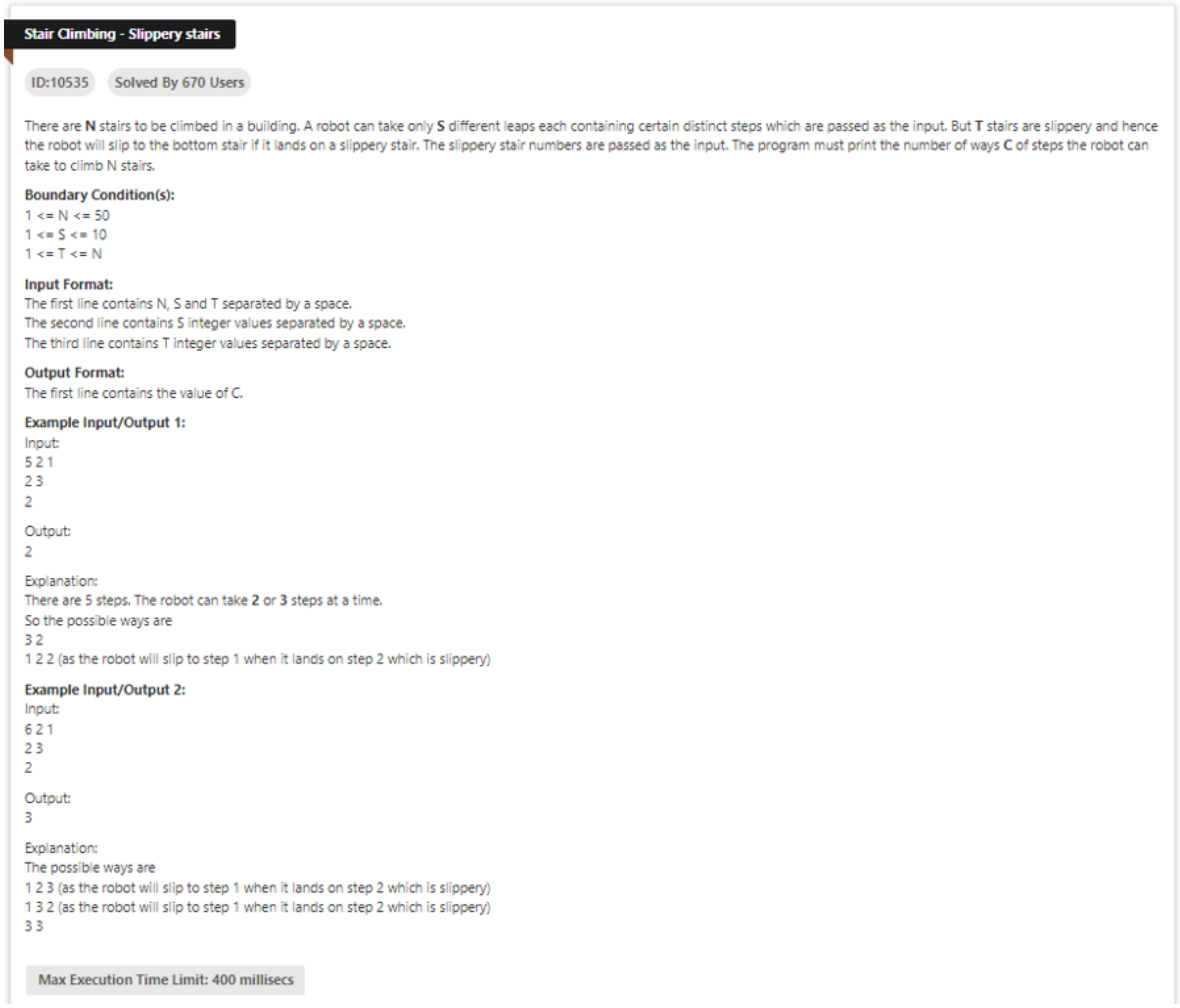
Explanation:
The possible ways are
3 5 7 10
3 5 8 10

Max Execution Time Limit: 400 millisecs

Code:

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

class damagedStairs{
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n=in.nextInt();
        int l=in.nextInt();
        int t=in.nextInt();
        int leaps[] = new int[l];
        for(int i=0;i<l;i++){
            leaps[i]=in.nextInt();
        }
        List<Integer> damaged =new ArrayList<>();
        for(int ctr=1;ctr<=t;ctr++){
            damaged.add(in.nextInt());
        }
        long ways[]=new long[n+1];
        ways[0]=1;
        for(int step=1;step<=n;step++){
            if(damaged.contains(step)){
                ways[step]=0;
                continue;
            }
            for(int i=0;i<l;i++){
                if(step>=leaps[i]){
                    ways[step]+=ways[step-leaps[i]];
                }
            }
        }
        System.out.println(ways[n]);
    }
}
```

Problem 3:



Stair Climbing - Slippery stairs

ID:10535 Solved By 670 Users

There are **N** stairs to be climbed in a building. A robot can take only **S** different leaps each containing certain distinct steps which are passed as the input. But **T** stairs are slippery and hence the robot will slip to the bottom stair if it lands on a slippery stair. The slippery stair numbers are passed as the input. The program must print the number of ways **C** of steps the robot can take to climb N stairs.

Boundary Condition(s):
1 <= N <= 50
1 <= S <= 10
1 <= T <= N

Input Format:
The first line contains N, S and T separated by a space.
The second line contains S integer values separated by a space.
The third line contains T integer values separated by a space.

Output Format:
The first line contains the value of C.

Example Input/Output 1:
Input:
5 2 1
2 3
2

Output:
2

Explanation:
There are 5 steps. The robot can take **2** or **3** steps at a time.
So the possible ways are
3 2
1 2 2 (as the robot will slip to step 1 when it lands on step 2 which is slippery)

Example Input/Output 2:
Input:
6 2 1
2 3
2

Output:
3

Explanation:
The possible ways are
1 2 3 (as the robot will slip to step 1 when it lands on step 2 which is slippery)
1 3 2 (as the robot will slip to step 1 when it lands on step 2 which is slippery)
3 3

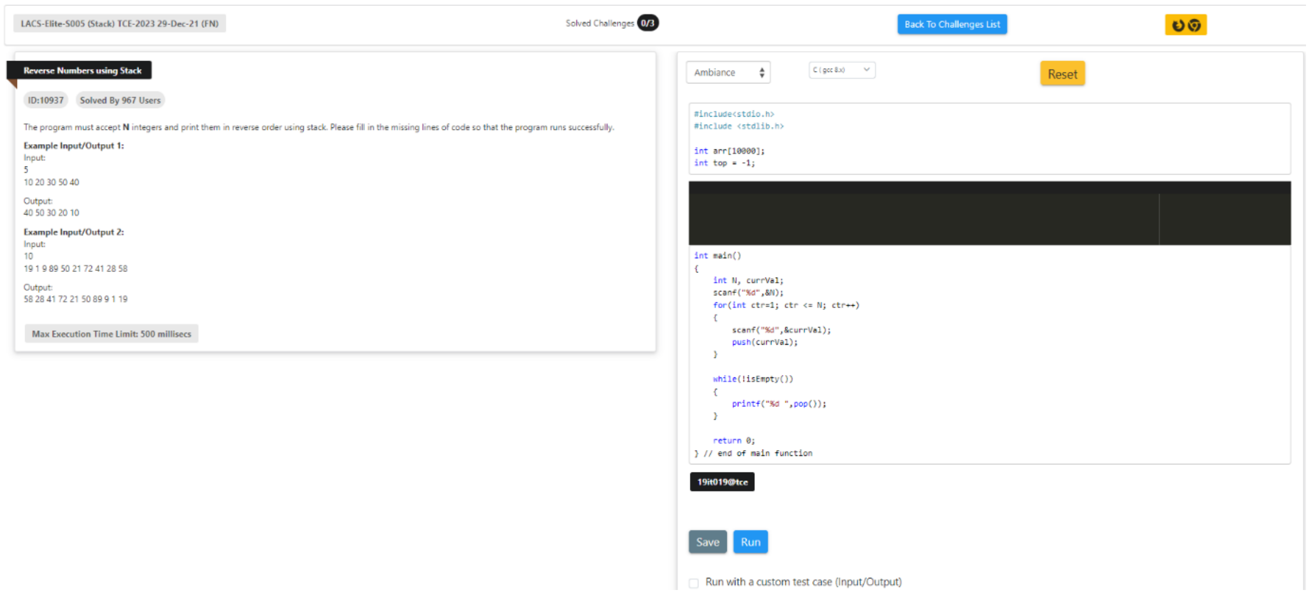
Max Execution Time Limit: 400 millisecs

Code:

```java
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

class slipperyStairs{
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n=in.nextInt();
        int l=in.nextInt();
        int t=in.nextInt();
        int leaps[] = new int[l];
        for(int i=0;i<l;i++){
            leaps[i]=in.nextInt();
        }
        List<Integer> slippery =new ArrayList<>();
        for(int ctr=1;ctr<=t;ctr++){
            slippery.add(in.nextInt());
        }
        long ways[]=new long[n+1];
        ways[0]=1;
        for(int step=1;step<=n;step++){
            for(int i=0;i<l;i++){
                if(step>=leaps[i]){
                    ways[step]+=ways[step-leaps[i]];
                }
            }
            if(slippery.contains(step)){
                int  lastNonSlippery = step-1;
                while(slippery.contains(lastNonSlippery)&& lastNonSlippery>0){
                    lastNonSlippery--;
                }
                if(lastNonSlippery>0){
                    ways[lastNonSlippery]+=ways[step];
                }
                ways[step]=0;
            }
        }
        System.out.println(ways[n]);
    }
}
```

SESSION 2:

Problem 1:


LACS-Elite-S005 (Stack) TCE-2023 29-Dec-21 (FN)
Solved Challenges 0/3
Back To Challenges List
Reverse Numbers using Stack
ID:10937
Solved By 967 Users
Ambiance
C (gcc 8.x)
Reset
19it019@tce
Save
Run
Run with a custom test case (Input/Output)

The program must accept **N** integers and print them in reverse order using stack. Please fill in the missing lines of code so that the program runs successfully.

Example Input/Output 1:
Input:
5
10 20 30 50 40

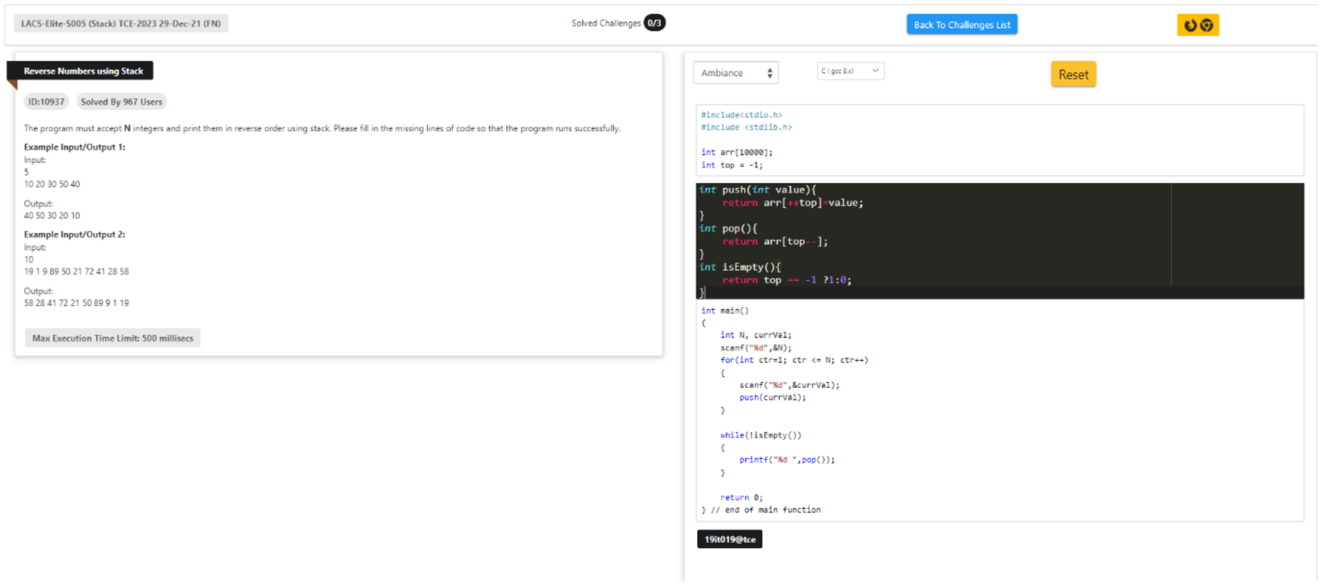
Output:
40 50 30 20 10

Example Input/Output 2:
Input:
10
19 1 9 89 50 21 72 41 28 58

Output:
58 28 41 72 21 50 89 9 1 19

Max Execution Time Limit: 500 millisecs

```c
#include<stdio.h>
#include <stdlib.h>

int arr[10000];
int top = -1;

int main()
{
    int N, currVal;
    scanf("%d",&N);
    for(int ctr=1; ctr <= N; ctr++)
    {
        scanf("%d",&currVal);
        push(currVal);
    }

    while(!isEmpty())
    {
        printf("%d ",pop());
    }

    return 0;
} // end of main function
```



The program must accept **N** integers and print them in reverse order using stack. Please fill in the missing lines of code so that the program runs successfully.

Example Input/Output 1:
Input:
5
10 20 30 50 40

Output:
40 50 30 20 10

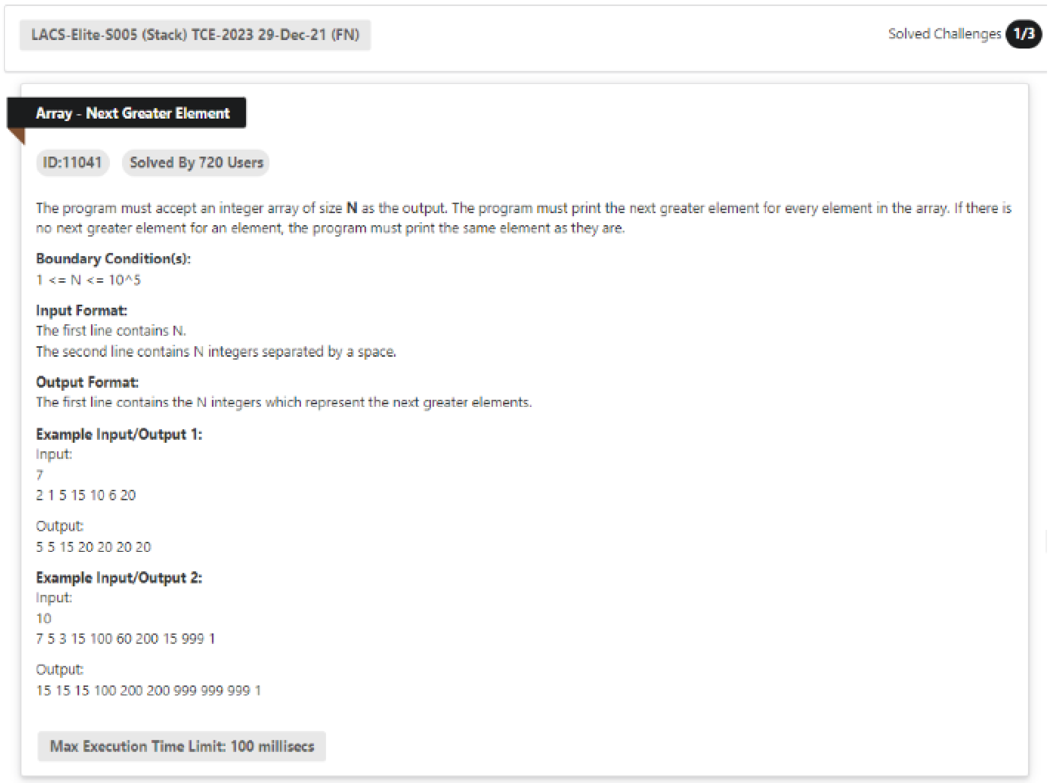
Example Input/Output 2:
Input:
10
19 1 9 89 50 21 72 41 28 58

Output:
58 28 41 72 21 50 89 9 1 19

Max Execution Time Limit: 500 millisecs

```c
#include<stdio.h>
#include <stdlib.h>

int arr[10000];
int top = -1;

int push(int value){
    return arr[++top]=value;
}
int pop(){
    return arr[top--];
}
int isEmpty(){
    return top == -1 ?1:0;
}
int main()
{
    int N, currVal;
    scanf("%d",&N);
    for(int ctr=1; ctr <= N; ctr++)
    {
        scanf("%d",&currVal);
        push(currVal);
    }

    while(!isEmpty())
    {
        printf("%d ",pop());
    }

    return 0;
} // end of main function
```

Problem 2:



LACS-Elite-S005 (Stack) TCE-2023 29-Dec-21 (FN)

Solved Challenges 1/3

Array - Next Greater Element

ID:11041 Solved By 720 Users

The program must accept an integer array of size **N** as the output. The program must print the next greater element for every element in the array. If there is no next greater element for an element, the program must print the same element as they are.

Boundary Condition(s):
1 <= N <= 10^5

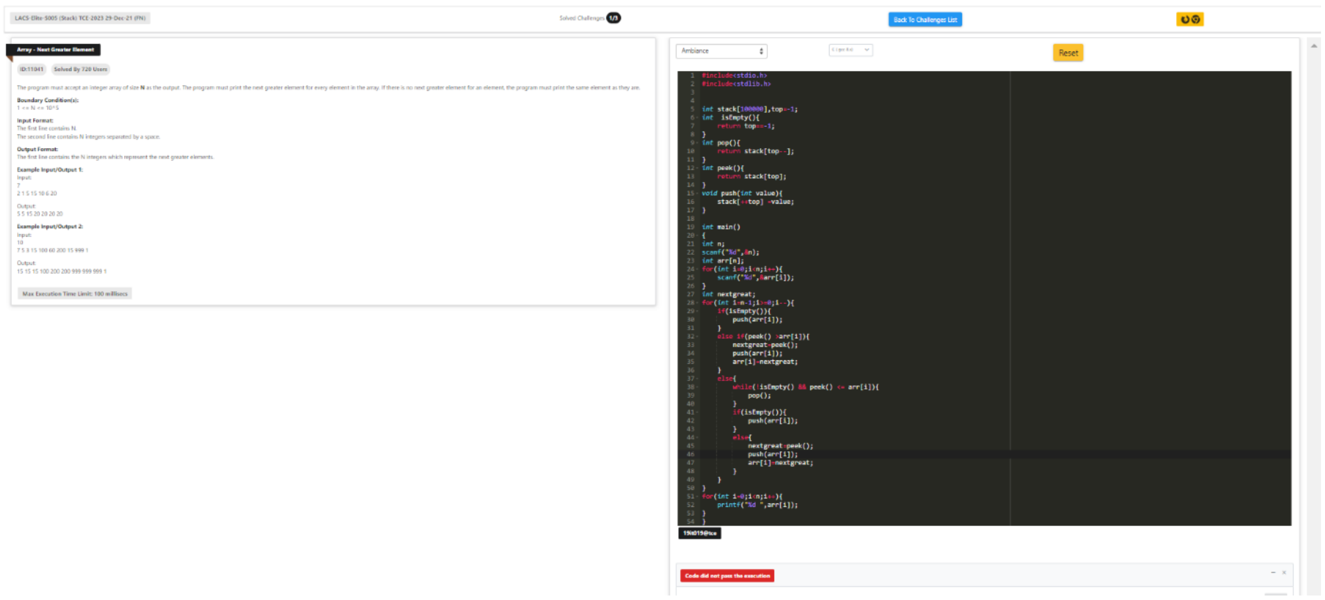
Input Format:
The first line contains N.
The second line contains N integers separated by a space.

Output Format:
The first line contains the N integers which represent the next greater elements.

Example Input/Output 1:
Input:
```
7
2 1 5 15 10 6 20
```
Output:
```
5 5 15 20 20 20 20
```

Example Input/Output 2:
Input:
```
10
7 5 3 15 100 60 200 15 999 1
```
Output:
```
15 15 15 100 200 200 999 999 999 1
```

Max Execution Time Limit: 100 millisecs



LACS-Elite-S005 (Stack) TCE-2023 29-Dec-21 (FN)

Solved Challenges 1/3

Array - Next Greater Element

ID:11041 Solved By 720 Users

The program must accept an integer array of size N as the output. The program must print the next greater element for every element in the array. If there is no next greater element for an element, the program must print the same element as they are.

Boundary Condition(s):
1 <= N <= 10^5

Input Format:
The first line contains N.
The second line contains N integers separated by a space.

Output Format:
The first line contains the N integers which represent the next greater elements.

Example Input/Output 1:
Input:
7
2 1 5 15 10 6 20

Output:
5 5 15 20 20 20 20

Example Input/Output 2:
Input:
10
7 5 3 15 100 60 200 15 999 1

Output:
15 15 15 100 200 200 999 999 999 1

Max Execution Time Limit: 100 millisecs

```c
#include<stdio.h>
#include<stdlib.h>


int stack[100000],top=-1;
int  isEmpty(){
    return top==-1;
}
int pop(){
    return stack[top--];
}
int peek(){
    return stack[top];
}
void push(int value){
    stack[++top] =value;
}

int main()
{
int n;
scanf("%d",&n);
int arr[n];
for(int i=0;i<n;i++){
    scanf("%d",&arr[i]);
}
int nextgreat;
for(int i=n-1;i>=0;i--){
    if(isEmpty()){
        push(arr[i]);
    }
    else if(peek() >arr[i]){
        nextgreat=peek();
        push(arr[i]);
        arr[i]=nextgreat;
    }
    else{
        while(!isEmpty() && peek() <= arr[i]){
            pop();
        }
        if(isEmpty()){
            push(arr[i]);
        }
        else{
            nextgreat=peek();
            push(arr[i]);
            arr[i]=nextgreat;
        }
    }
}
for(int i=0;i<n;i++){
    printf("%d ",arr[i]);
}
}
```

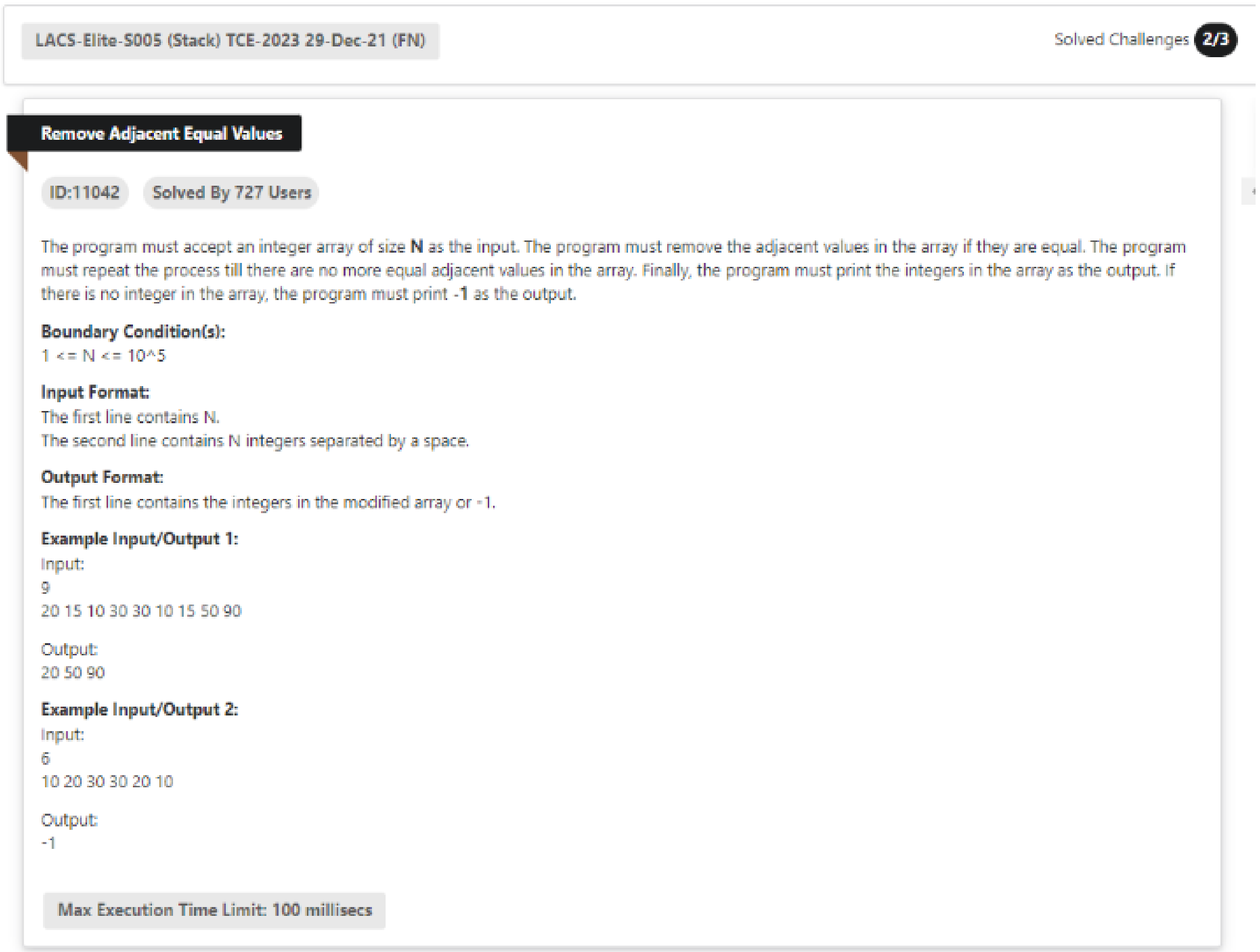
19it019@tce

Code did not pass the execution

Code:

```
#include<stdio.h>
#include<stdlib.h>
int stack[100000];
int top=-1;
void push(int a){
    stack[++top]=a;
}
int pop(){
    return stack[top--];
}
int isEmpty(){
    return top==-1;
}
int peek(){
    return stack[top];
}
int main()
{
    int a;
    scanf("%d",&a);
    int arr[a];
    for(int i=0;i<a;i++){
        scanf("%d",&arr[i]);
    }
    int nextGreater;
    for(int i=a-1;i>=0;i--){
        if(isEmpty()){
            push(arr[i]);    //last ele is pushed
        }else if(peek() > arr[i] ){      // eleemnt in stack > traversing ele
            nextGreater = peek();
            push(arr[i]);
            arr[i] = nextGreater;
        }else{
            while(!isEmpty() && peek() <= arr[i]){
                pop();
            }
            if(isEmpty()){
                push(arr[i]);
            }
            else{
                nextGreater=peek();
                push(arr[i]);
                arr[i]=nextGreater;
            }
        }
    }
    for(int i=0;i<a;i++){
        printf("%d ",arr[i]);
    }
}
```

Problem 2:



LACS-Elite-S005 (Stack) TCE-2023 29-Dec-21 (FN)

Solved Challenges 2/3

Remove Adjacent Equal Values

ID:11042 Solved By 727 Users

The program must accept an integer array of size **N** as the input. The program must remove the adjacent values in the array if they are equal. The program must repeat the process till there are no more equal adjacent values in the array. Finally, the program must print the integers in the array as the output. If there is no integer in the array, the program must print **-1** as the output.

Boundary Condition(s):
1 <= N <= 10^5

Input Format:
The first line contains N.
The second line contains N integers separated by a space.

Output Format:
The first line contains the integers in the modified array or -1.

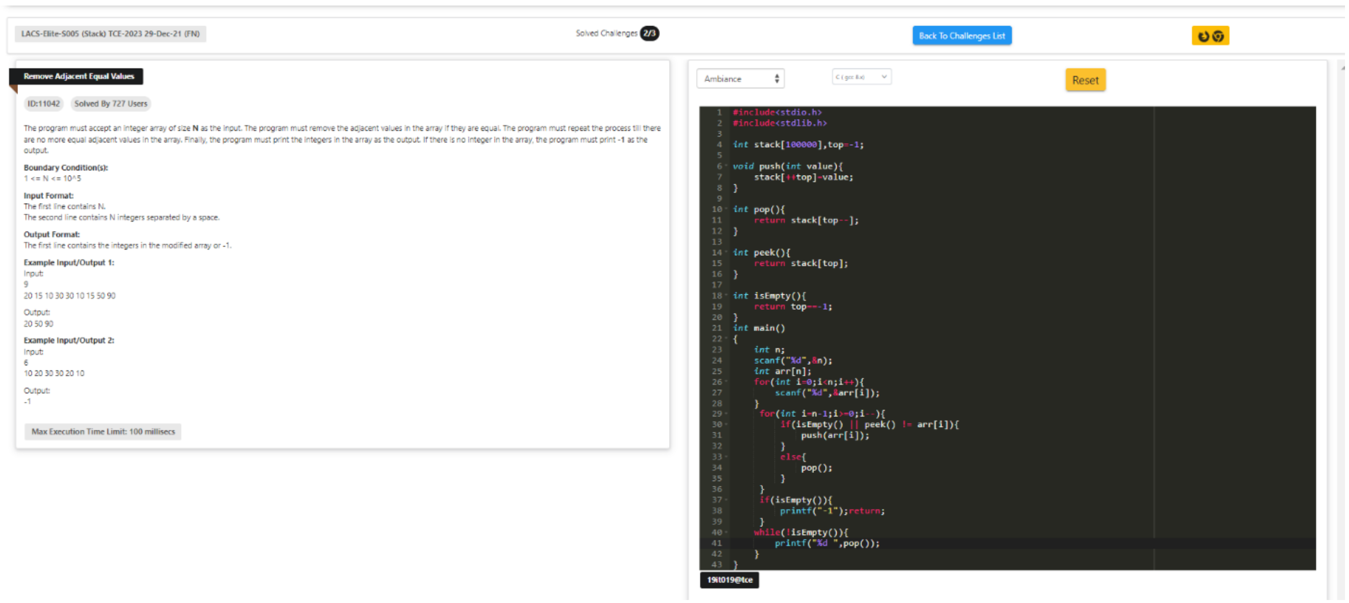
Example Input/Output 1:
Input:
9
20 15 10 30 30 10 15 50 90

Output:
20 50 90

Example Input/Output 2:
Input:
6
10 20 30 30 20 10

Output:
-1

Max Execution Time Limit: 100 millisecs



LACS-Elite-S005 (Stack) TCE-2023 29-Dec-21 (FN)

Solved Challenges 2/3

Remove Adjacent Equal Values

ID:11042 Solved By 727 Users

The program must accept an integer array of size **N** as the input. The program must remove the adjacent values in the array if they are equal. The program must repeat the process till there are no more equal adjacent values in the array. Finally, the program must print the integers in the array as the output. If there is no integer in the array, the program must print **-1** as the output.

Boundary Condition(s):
1 <= N <= 10^5

Input Format:
The first line contains N.
The second line contains N integers separated by a space.

Output Format:
The first line contains the integers in the modified array or -1.

Example Input/Output 1:
Input:
9
20 15 10 30 30 10 15 50 90

Output:
20 50 90

Example Input/Output 2:
Input:
6
10 20 30 30 20 10

Output:
-1

Max Execution Time Limit: 100 millisecs

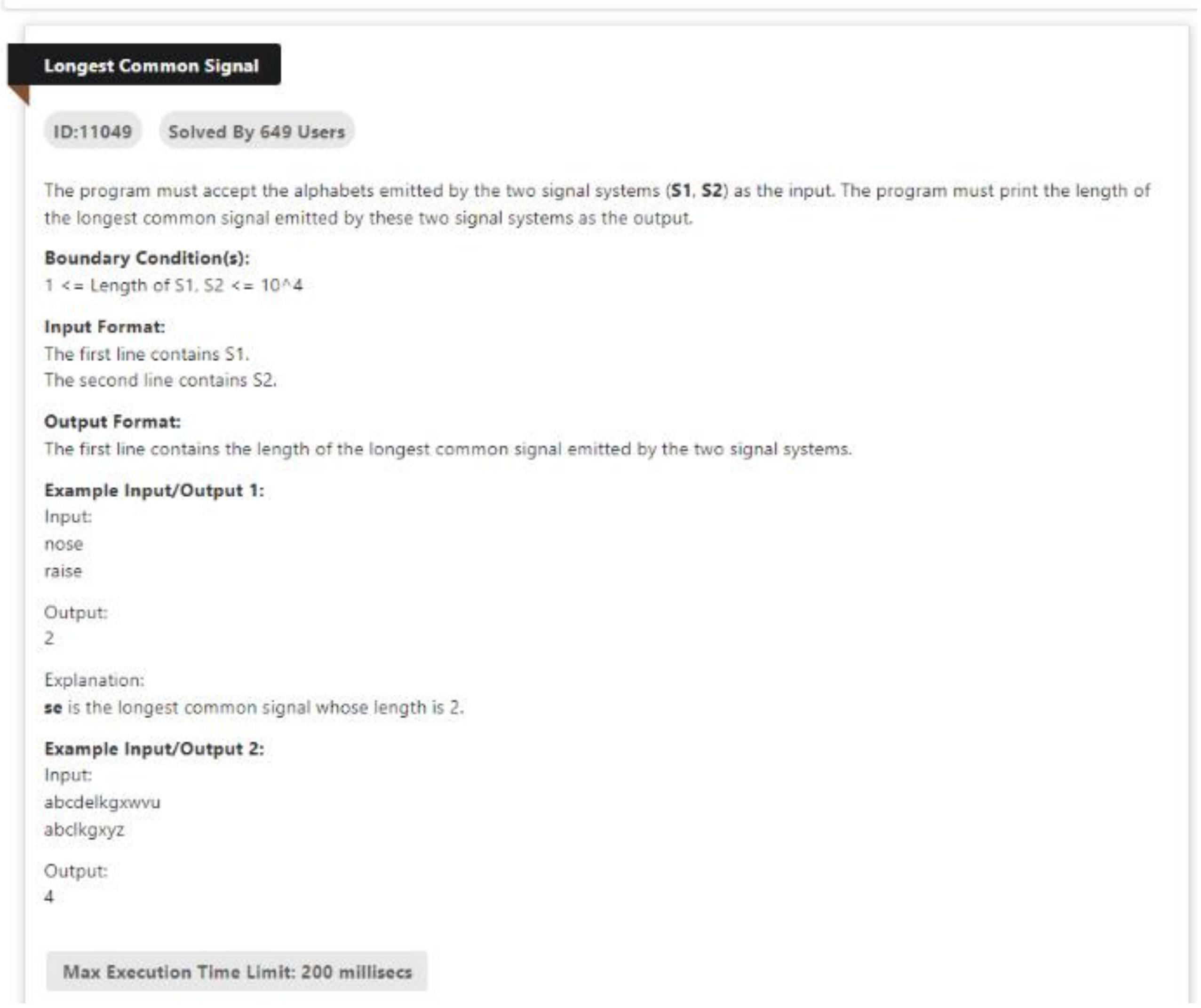
```
#include<stdio.h>
#include<stdlib.h>

int stack[100000],top=-1;

void push(int value){
    stack[++top]=value;
}

int pop(){
    return stack[top--];
}

int peek(){
    return stack[top];
}

int isEmpty(){
    return top==-1;
}
int main()
{
    int n;
    scanf("%d",&n);
    int arr[n];
    for(int i=0;i<n;i++){
        scanf("%d",&arr[i]);
    }
     for(int i=n-1;i>=0;i--){
         if(isEmpty() || peek() != arr[i]){
             push(arr[i]);
         }
         else{
             pop();
         }
     }
     if(isEmpty()){
         printf("-1");return;
     }
    while(!isEmpty()){
        printf("%d ",pop());
    }
}
```

19it019@tce

Code:

```
#include<stdio.h>
#include<stdlib.h>
int stack[100000];
int top=-1;
void push(int a){
    stack[++top]=a;
}
int pop(){
    return stack[top--];
}
int isEmpty(){
    return top==-1;
}
int peek(){
    return stack[top];
}
int main()
{
    int a;
    scanf("%d",&a);
    int arr[a];
    for(int i=0;i<a;i++){
        scanf("%d",&arr[i]);
    }
    for(int i=a-1;i>=0;i--){
        if(isEmpty() || peek() != arr[i]){
            push(arr[i]);
        }
        else{
            pop();
        }
    }
    if(isEmpty()){
        printf("-1");
        return;
    }
    while(!isEmpty()){
        printf("%d ",pop());
    }
}
```

SESSION 3:



Longest Common Signal

ID:11049 Solved By 649 Users

The program must accept the alphabets emitted by the two signal systems (**S1**, **S2**) as the input. The program must print the length of the longest common signal emitted by these two signal systems as the output.

Boundary Condition(s):
1 <= Length of S1, S2 <= 10^4

Input Format:
The first line contains S1.
The second line contains S2.

Output Format:
The first line contains the length of the longest common signal emitted by the two signal systems.

Example Input/Output 1:
Input:
nose
raise

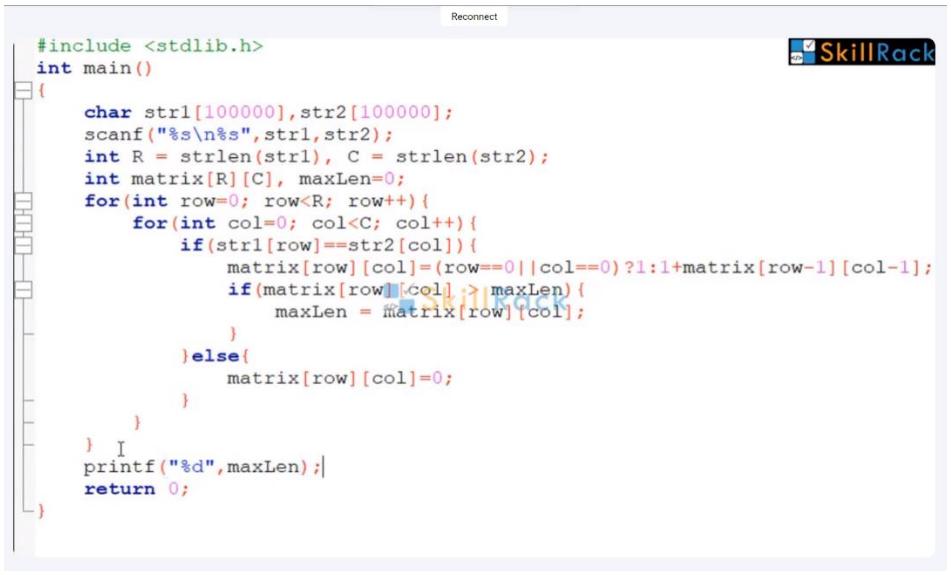
Output:
2

Explanation:
se is the longest common signal whose length is 2.

Example Input/Output 2:
Input:
abcdelkgxwvu
abclkgxyz

Output:
4

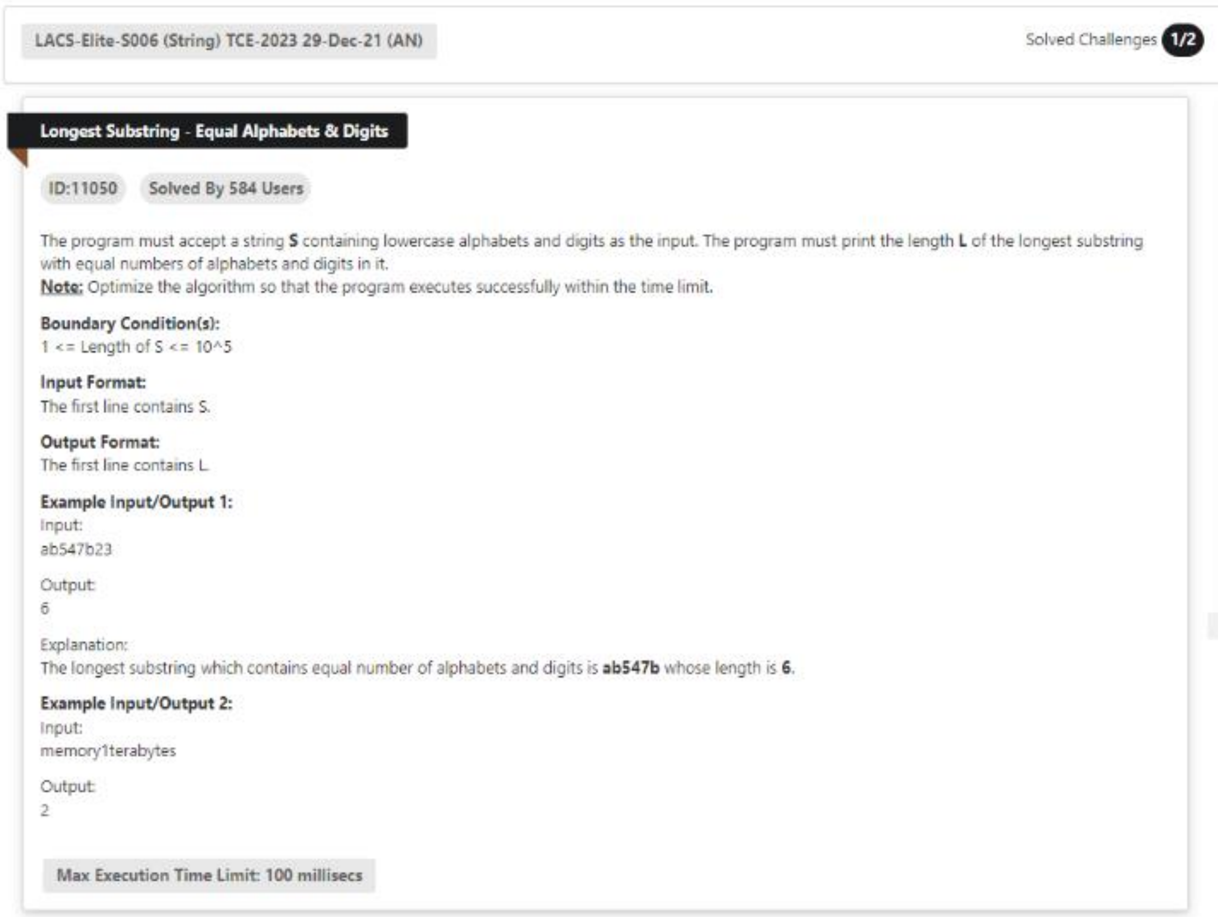
Max Execution Time Limit: 200 millisecs



```
#include <stdlib.h>
int main()
{
    char str1[100000],str2[100000];
    scanf("%s\n%s",str1,str2);
    int R = strlen(str1), C = strlen(str2);
    int matrix[R][C], maxLen=0;
    for(int row=0; row<R; row++){
        for(int col=0; col<C; col++){
            if(str1[row]==str2[col]){
                matrix[row][col]=(row==0||col==0)?1:1+matrix[row-1][col-1];
                if(matrix[row][col] > maxLen){
                    maxLen = matrix[row][col];
                }
            }else{
                matrix[row][col]=0;
            }
        }
    }
    printf("%d",maxLen);
    return 0;
}
```

Code:

```
#include <stdio.h>
#include <string.h>
int main()
{
    char str1[100000],str2[100000];
    scanf("%s\n%s",str1,str2);
    int R=strlen(str1),C=strlen(str2);
    int matrix[R][C],maxLen=0;
    for(int row=0;row<R;row++){
        for(int col=0;col<C;col++){
            if(str1[row]==str2[col]){
                matrix[row][col]=(row==0||col==0)?1:1+matrix[row-1][col-1];
                if(matrix[row][col]>maxLen){
                    maxLen=matrix[row][col];
                }
            }
            else{
                matrix[row][col]=0;
            }
        }
    }
    printf("%d",maxLen);
    return 0;
}
```


LACS-Elite-S006 (String) TCE-2023 29-Dec-21 (AN)

Solved Challenges 1/2

Longest Substring - Equal Alphabets & Digits

ID:11050 Solved By 584 Users

The program must accept a string **S** containing lowercase alphabets and digits as the input. The program must print the length **L** of the longest substring with equal numbers of alphabets and digits in it.
Note: Optimize the algorithm so that the program executes successfully within the time limit.

Boundary Condition(s):
1 <= Length of S <= 10^5

Input Format:
The first line contains S.

Output Format:
The first line contains L.

Example Input/Output 1:
Input:
ab547b23

Output:
6

Explanation:
The longest substring which contains equal number of alphabets and digits is **ab547b** whose length is **6**.

Example Input/Output 2:
Input:
memory1terabytes

Output:
2

Max Execution Time Limit: 100 millisecs

Code:

```
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class equalAlphaNDigits {
    public static void main(String[] args) {
        Scanner in= new Scanner(System.in);
        String str=in.nextLine();
        int counter=0,position=0,maxlen=0;
        Map<Integer,Integer> startposmap=new HashMap<>();
        startposmap.put(counter, position);
        for(char ch:str.toCharArray()){
            position++;
            if(Character.isAlphabetic(ch)){
                counter++;
            }
            else{
                counter--;
            }
            if(startposmap.containsKey(counter)){
                int currLen=position-startposmap.get(counter);
                if(currLen>maxlen){
                    maxlen=currLen;
                }
            }
            else{
                startposmap.put(counter, position);
            }
        }
        System.out.println(maxlen);
    }
}
```