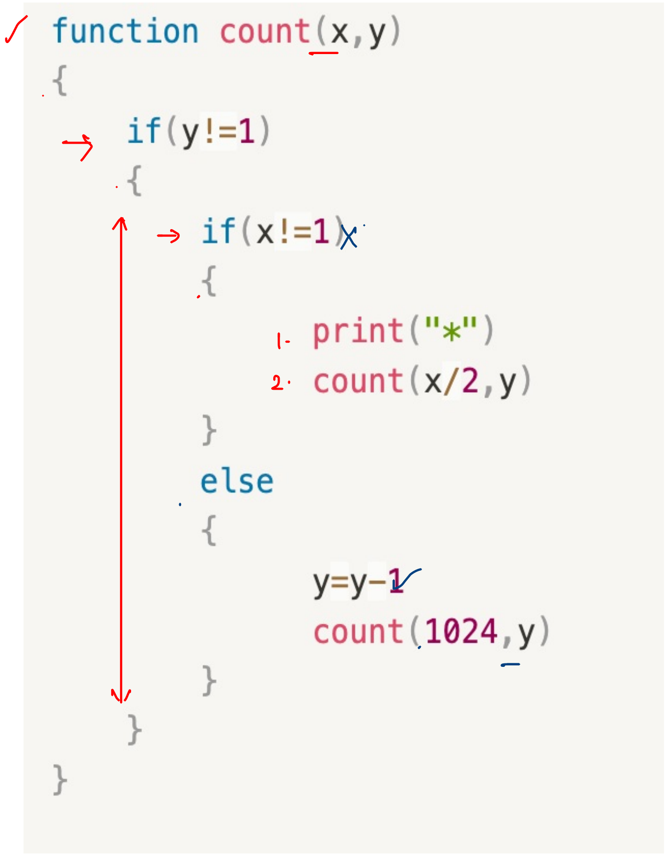
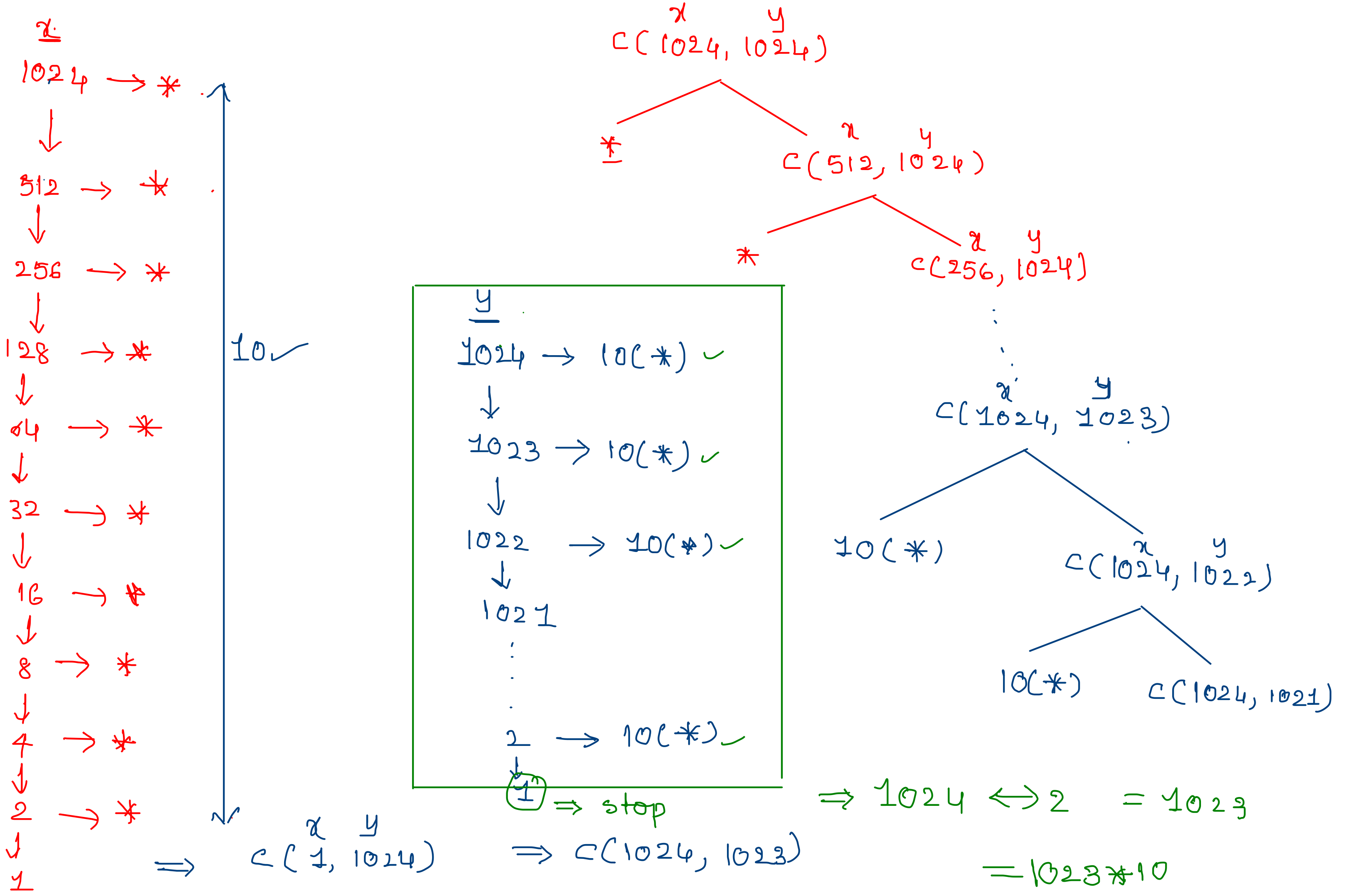
how many number of * will be printed, when the function count is called with count(1024,1024) ?



```
function count(x,y)
{
    if(y!=1)
    {
        if(x!=1)
        {
            print("*")
            count(x/2,y)
        }
        else
        {
            y=y-1
            count(1024,y)
        }
    }
}
```



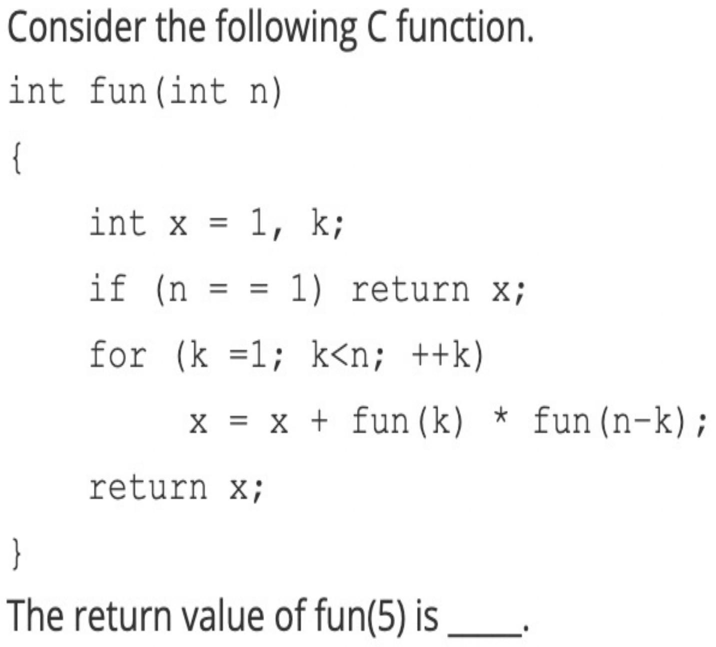
⇒ 1024 ⟷ 2 = 1023

= 1023*10

= 10230 (*)

what is the output for the following code ?

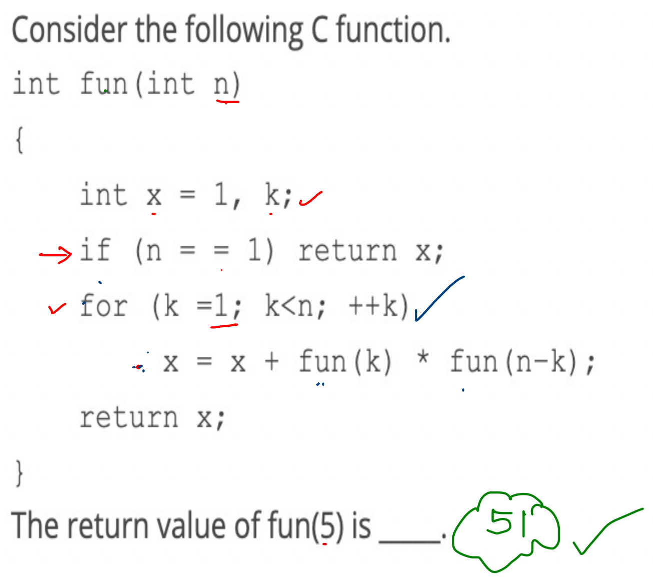
10 - mins



Consider the following C function.

```
int fun(int n)
{
    int x = 1, k;
    if (n = = 1) return x;
    for (k =1; k<n; ++k)
         x = x + fun(k) * fun(n-k);
    return x;
}
```

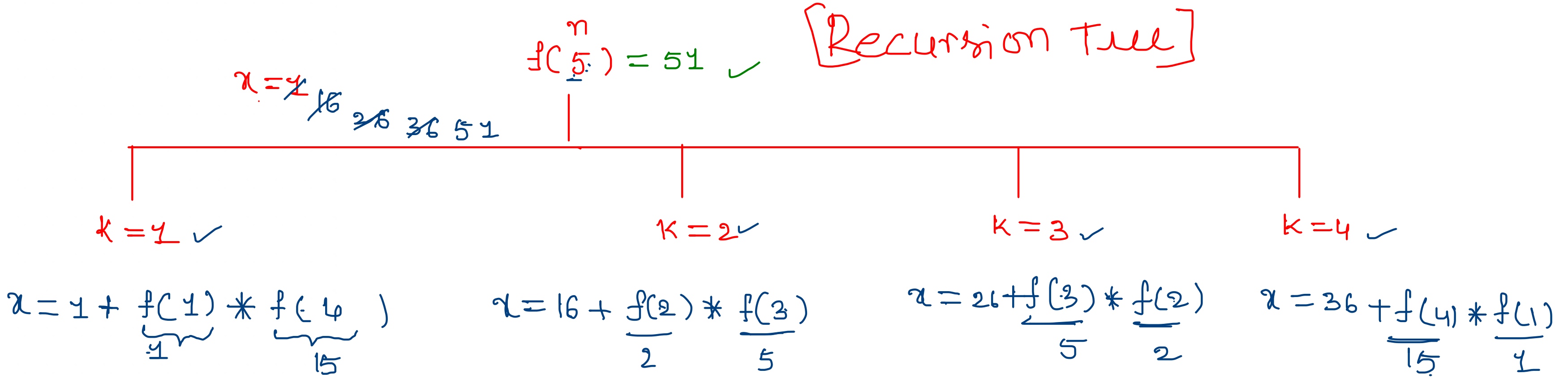
The return value of fun(5) is ____.

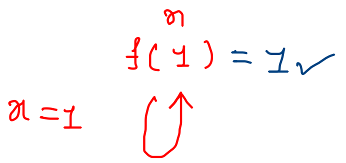


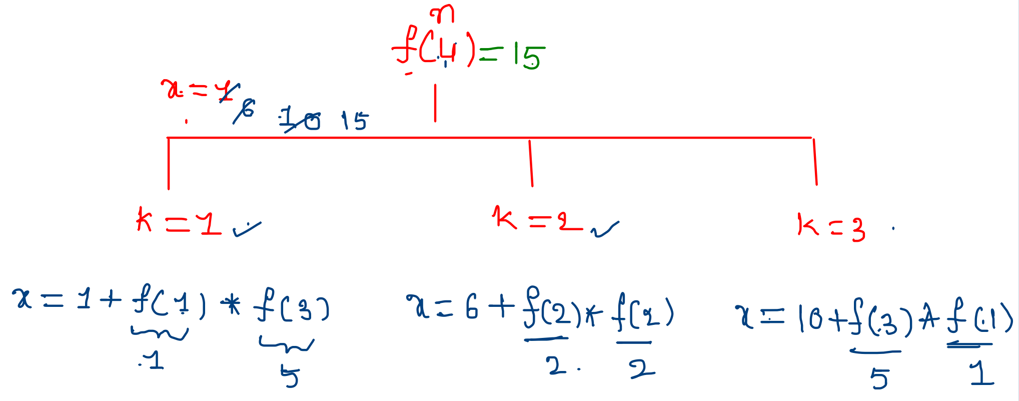
Consider the following C function.

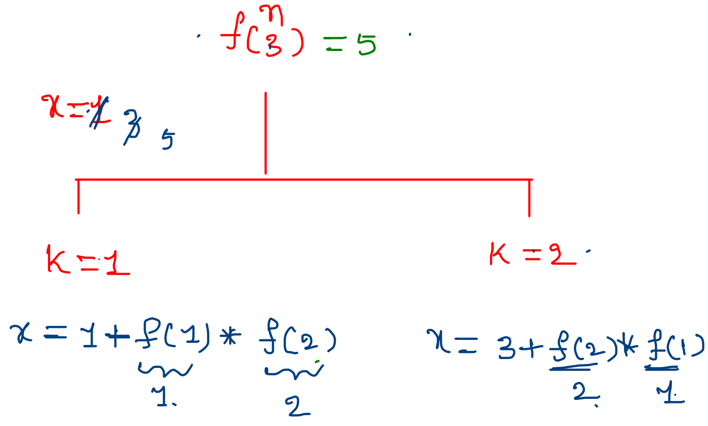
```
int fun(int n)
{
    int x = 1, k;
    if (n = = 1) return x;
    for (k =1; k<n; ++k)
        x = x + fun(k) * fun(n-k);
    return x;
}
```

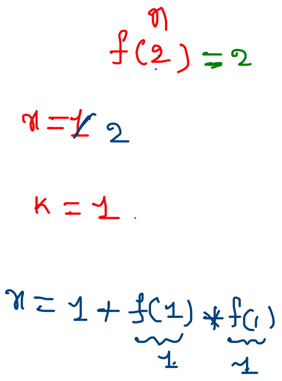
The return value of fun(5) is ____. 51

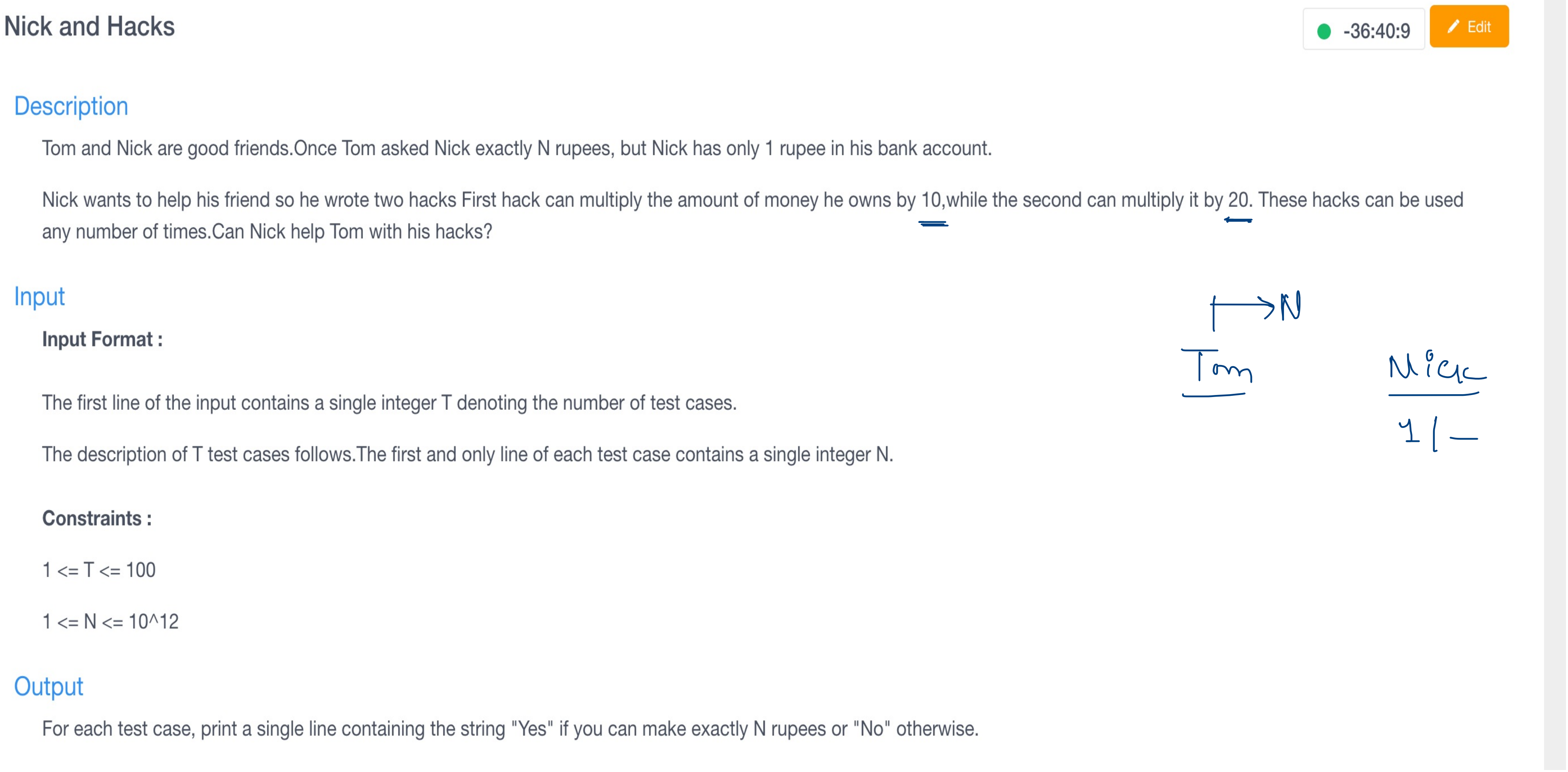












Nick and Hacks

Description

Tom and Nick are good friends.Once Tom asked Nick exactly N rupees, but Nick has only 1 rupee in his bank account.

Nick wants to help his friend so he wrote two hacks First hack can multiply the amount of money he owns by 10,while the second can multiply it by 20. These hacks can be used any number of times.Can Nick help Tom with his hacks?

→ N

Tom

Nick

1 |

Input

Input Format :

The first line of the input contains a single integer T denoting the number of test cases.

The description of T test cases follows.The first and only line of each test case contains a single integer N.

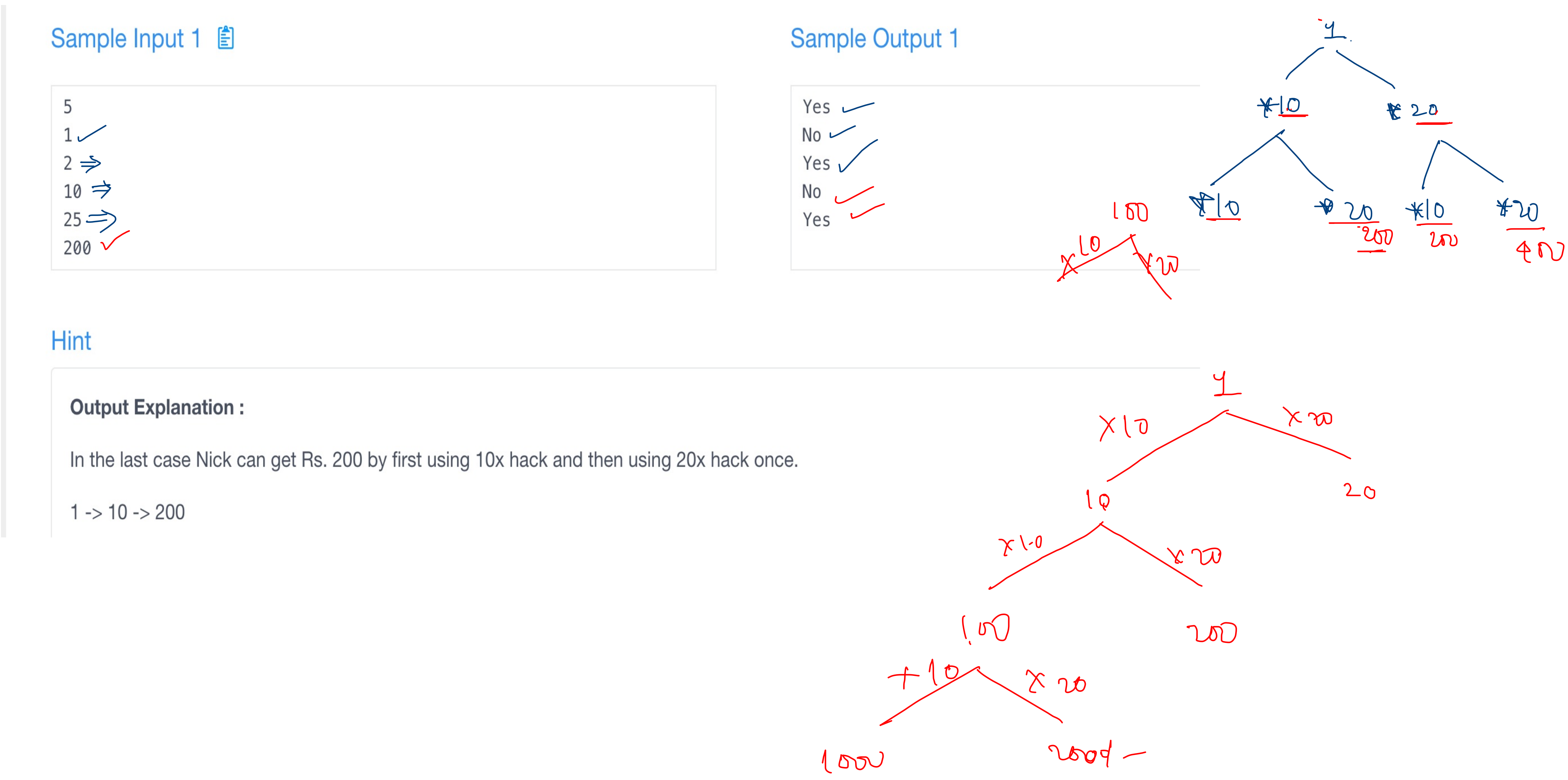
Constraints :

1 <= T <= 100

1 <= N <= 10^12

Output

For each test case, print a single line containing the string "Yes" if you can make exactly N rupees or "No" otherwise.



Sample Input 1

```
5
1
2
10
25
200
```

Sample Output 1

```
Yes
No
Yes
No
Yes
```

Hint

Output Explanation :

In the last case Nick can get Rs. 200 by first using 10x hack and then using 20x hack once.

1 -> 10 -> 200

```
function fun(n,x)
{
        if(n==x)
                return true
        if(x>n)
                return false
        return fun(n,x*10) || fun(n,x*20)
}
```

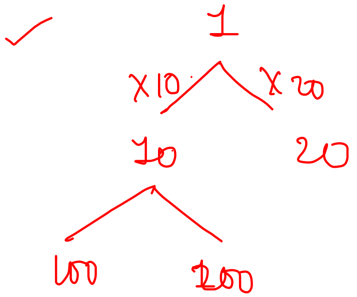
↳ logical OR. ✓

n == x ✓

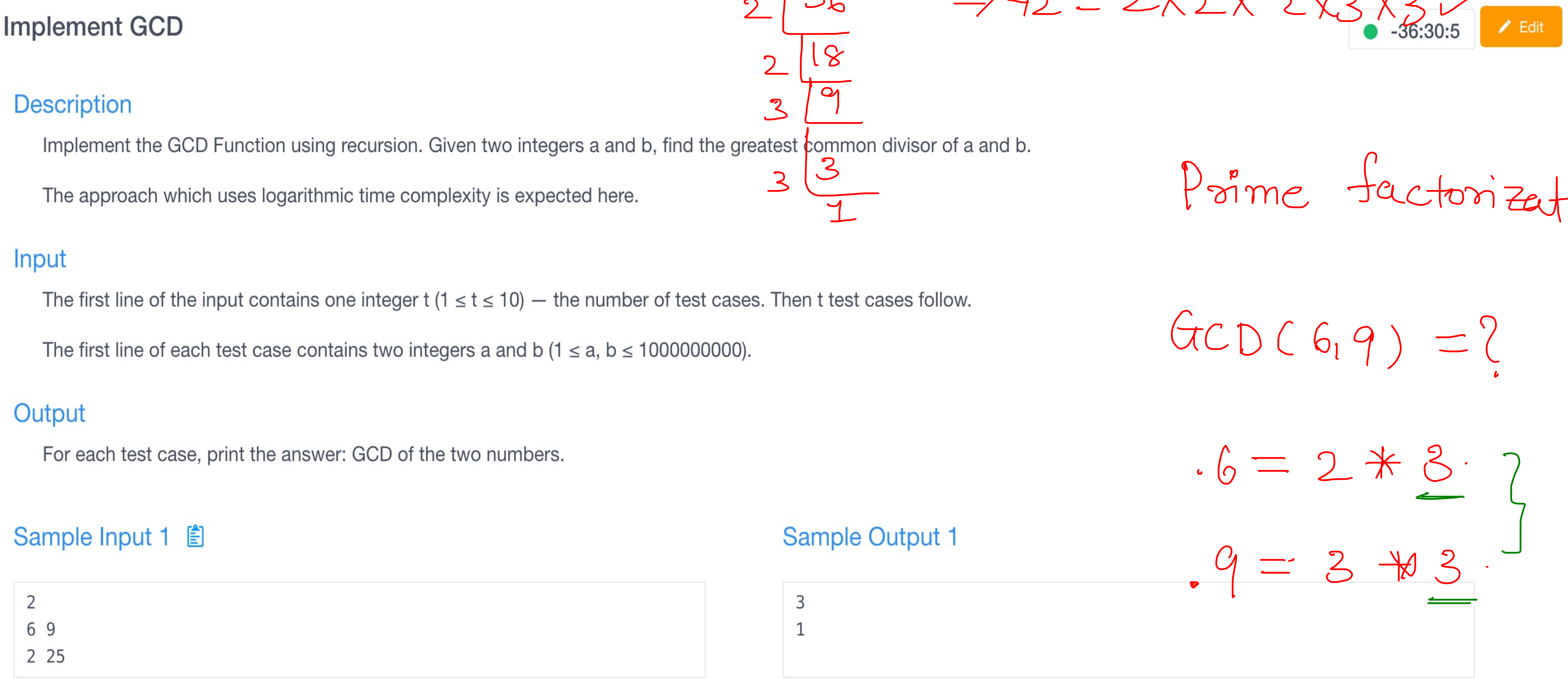
n < x

Tom   nick (X)

n ✓



A	B	A || B
F	F	F
F	T	T
T	F	T
T	T	T



Implement GCD

Description

Implement the GCD Function using recursion. Given two integers a and b, find the greatest common divisor of a and b.

The approach which uses logarithmic time complexity is expected here.

Input

The first line of the input contains one integer t (1 ≤ t ≤ 10) — the number of test cases. Then t test cases follow.

The first line of each test case contains two integers a and b (1 ≤ a, b ≤ 1000000000).

Output

For each test case, print the answer: GCD of the two numbers.

Sample Input 1

```
2
6 9
2 25
```

Sample Output 1

```
3
1
```

$$\begin{array}{r|l} 2 & 72 \\ \hline 2 & 36 \\ \hline 2 & 18 \\ \hline 3 & 9 \\ \hline 3 & 3 \\ \hline & 1 \end{array}$$

$\Rightarrow 72 = 2 \times 2 \times 2 \times 3 \times 3$ ✓

Prime factorization.

$GCD(6,9) = ?$

$6 = 2 * 3$

$9 = 3 * 3$

3 ✓

$\text{GCD}(36, 60) = ?$

$\text{GCD} \cong \text{HCF}$

$$\left.\begin{array}{l} 36 = \underline{2 \times 2 \times 3} \times 3 \\ 60 = \underline{2 \times 2 \times 3} \times 5 \end{array}\right\} 2 \times 2 \times 3 = \boxed{12} \checkmark$$

$$\begin{array}{l} 36)\ 60\ (1 \\ -\ \ 36 \\ \hline \quad 24)\ 36\ (1 \\ \quad -\ \ 24 \\ \hline \quad\quad \boxed{12})\ 24\ (2 \\ \quad\quad\quad\ \ 24 \\ \hline \quad\quad\quad\ \ \boxed{0} \end{array}$$

```
a      b
36) 60 (1
  − 36
  ------
    24) 36 (1      ⟶   GCD(36, 60) ✓
      − 24
      ------
        12) 24 (2  ⟶   GCD(24, 36) ✓
            24
            ----
            (0))12(  ⟶  GCD(12, 24) ✓
                     ⟶  GCD(0, 12)
```

a, b

GCD(36, 60) ✓

GCD(24, 36) ✓
a b

GCD(12, 24) ✓

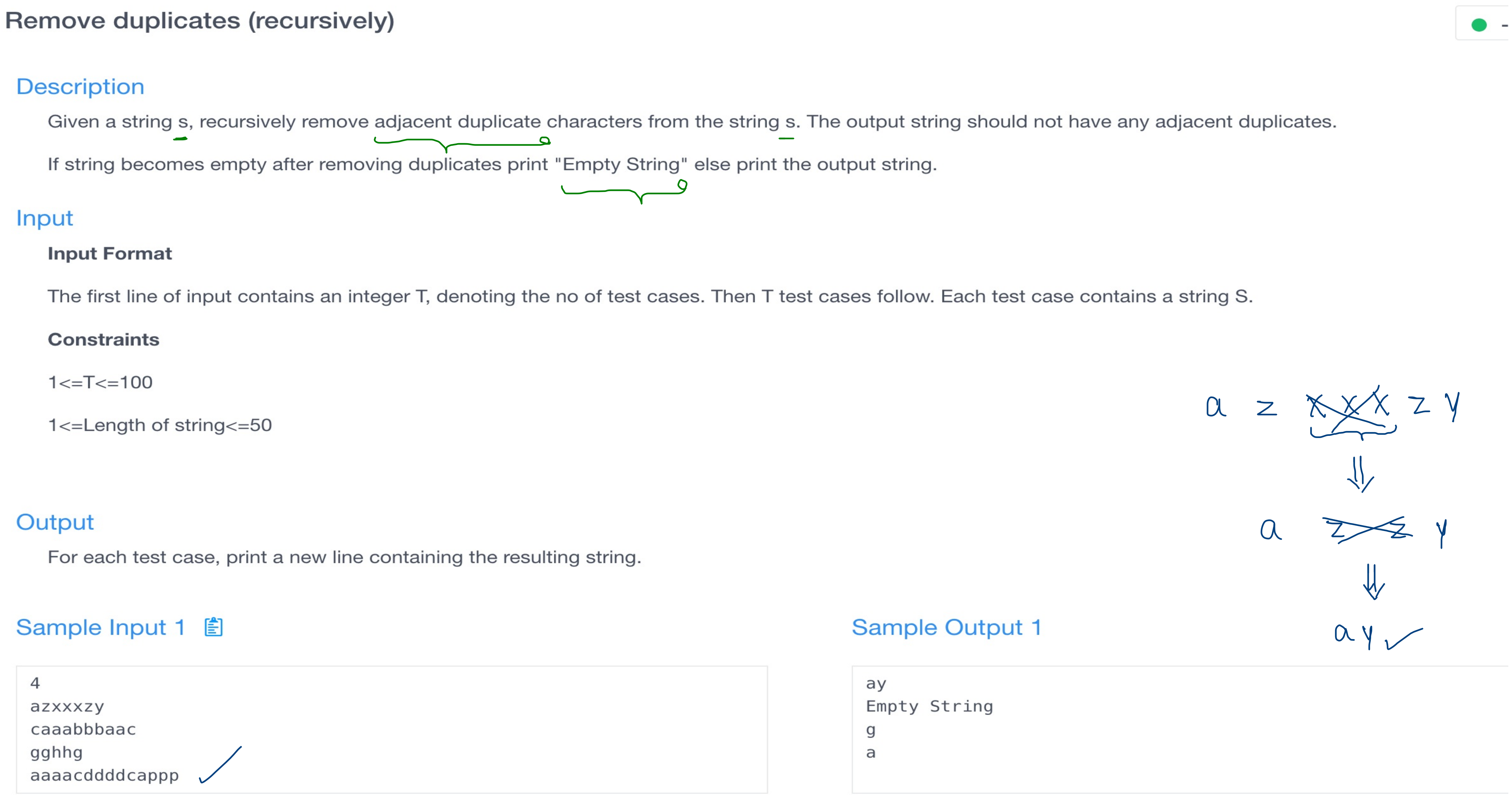
GCD(0, 12) ⇒ a = 0; b ✓

a = 36, b = 60

24? ⇒ 60%36

Euclidean Algo:-

```
function GCD(a,b)
{
    if(a == 0) return b
    else
        return GCD(b%a, a)
}
```



Remove duplicates (recursively)

Description

Given a string s, recursively remove adjacent duplicate characters from the string s. The output string should not have any adjacent duplicates.

If string becomes empty after removing duplicates print "Empty String" else print the output string.

Input

Input Format

The first line of input contains an integer T, denoting the no of test cases. Then T test cases follow. Each test case contains a string S.

Constraints

1<=T<=100

1<=Length of string<=50

$a\ z\ \underbrace{\cancel{x}\ \cancel{x}\ \cancel{x}}\ z\ y$

⇓

$a\ \cancel{z}\ \cancel{z}\ y$

⇓

a y ✓

Output

For each test case, print a new line containing the resulting string.

Sample Input 1

```
4
azxxxzy
caaabbbaac
gghhg
aaaacddddcappp
```

✓

Sample Output 1

```
ay
Empty String
g
a
```

res = d ✓

s ↳

0	1	2	3	4	5	6
a	b	b	a	c	c	d ✓
~~i~~	~~i~~	~~i~~	~~i~~	~~i~~	~~i~~	i

S = a ~~b~~ b a c c d

⇓

~~a a~~ c c d

⇓

~~c c~~ d

⇓

d o/p

s ↳

0	1	2
a	a	d

s ↳

0
d
i

✓ ✓ o/p

```
function removeDuplicates(String s)
{
    if(s.trim().isEmpty())
        return s
    is(s.length()==1)
        return s
    res=""
    for(i=0;i<=s.length();i++)
    {
        if(i==0)
            if(s[i]!=s[i+1])
                res=res+s[i]
        else if(i==s.length()-1)
            if(s[i]!=s[i-1])
                res=res+s[i]
        else
        {
            if(s[i]!=s[i-1] && s[i]!=s[i+1])
                res=res+s[i]
        }
    }
    if(s.legth()==res.length()
        return res
    else
        return removeDuplicatess(res)
}
```

s

0 1 2 3

r(A B C D)

i i i i

4

res = ABCD

4

A ↳

0	1	2
A	B	c

→ Sub-Array

no-break |A|=3

$\boxed{A}$ $\boxed{B}$ $\boxed{C}$ → 3

$\boxed{A|B}$ $\boxed{B|C}$ → 2

$\boxed{A|B|C}$ → 1

$1+2+3 = 6$

$|A|=4$ } $1+2+3+4 = 10$

$1+2+3+ \ldots \ldots +n = \frac{n(n+1)}{2}$ ✓

→ Sub-Sequence.

empty sub-seq,

we will not consider

$\therefore 2^n - 1$

→ subset.

every ele can be present
cannot be "

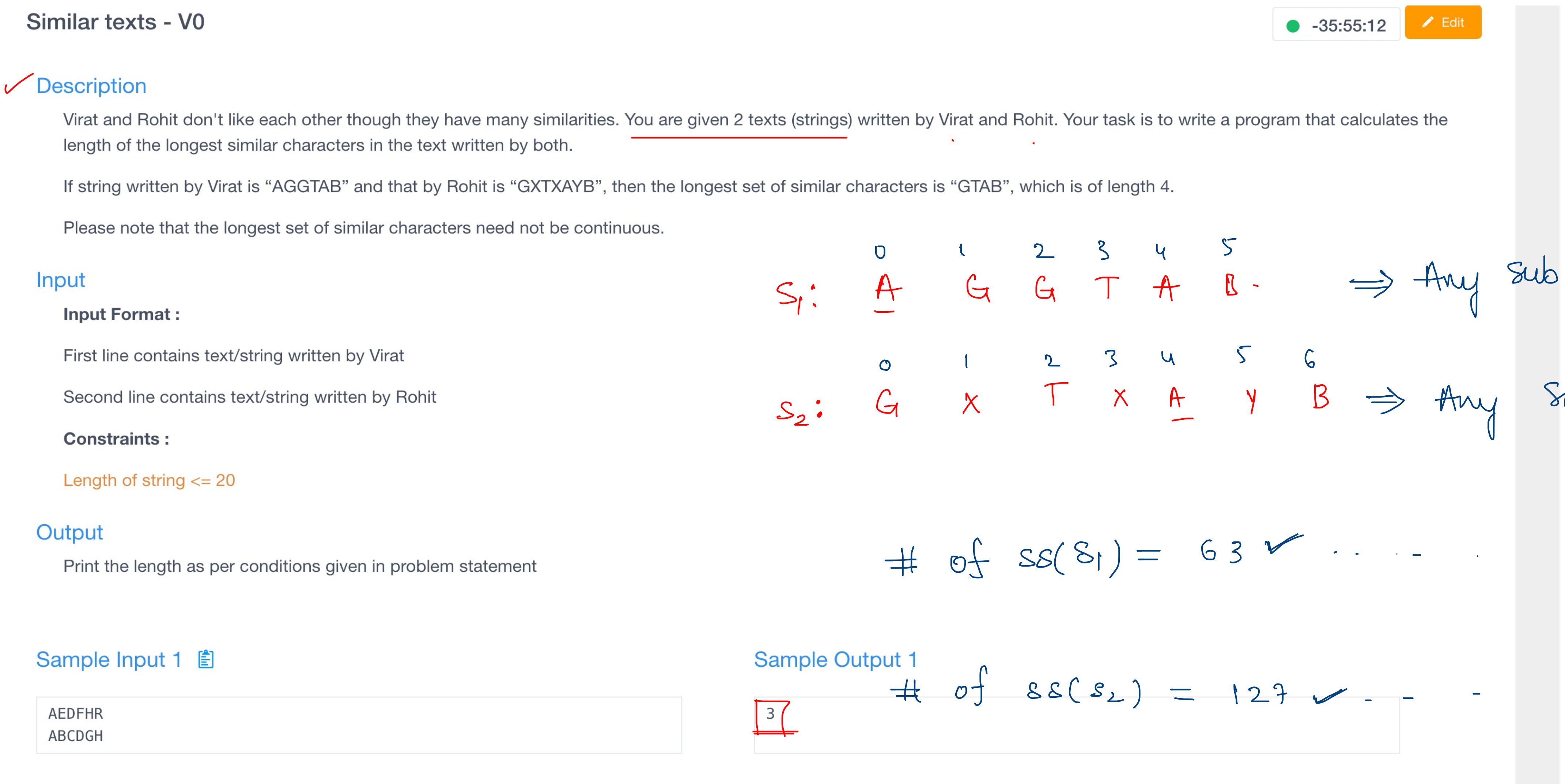
{ }, {A} {B} {C}

{A,B} {A, C}, {B,C}

{A, B, C}

$n=3 \Rightarrow 2^3 = 8$ ✓

$n \Rightarrow 2^n$ subsets.



Similar texts - V0

✓ Description

Virat and Rohit don't like each other though they have many similarities. You are given 2 texts (strings) written by Virat and Rohit. Your task is to write a program that calculates the length of the longest similar characters in the text written by both.

If string written by Virat is “AGGTAB” and that by Rohit is “GXTXAYB”, then the longest set of similar characters is “GTAB”, which is of length 4.

Please note that the longest set of similar characters need not be continuous.

Input

Input Format :

First line contains text/string written by Virat

Second line contains text/string written by Rohit

Constraints :

Length of string <= 20

Output

Print the length as per conditions given in problem statement

Sample Input 1

```
AEDFHR
ABCDGH
```

Sample Output 1

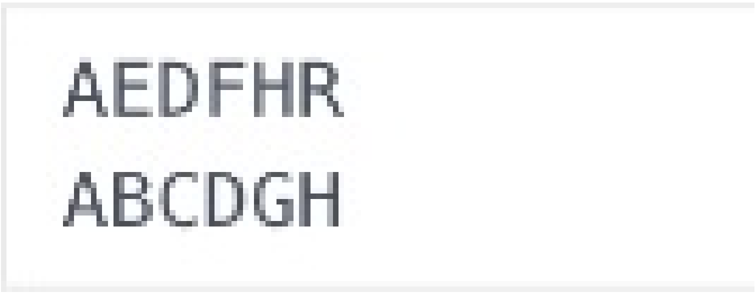
```
3
```

$$\begin{array}{ccccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ S_1: & \underline{A} & G & G & T & A & B \end{array} \Rightarrow \text{Any subseq} ✓$$

$$\begin{array}{cccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ S_2: & G & X & T & X & \underline{A} & Y & B \end{array} \Rightarrow \text{Any subseq} ✓$$

$\#$ of $SS(S_1) = 63$ ✓

$\#$ of $SS(S_2) = 127$ ✓



AEDFHR

ABCDGH

SSeq

S_1 : A E D F H R ⇒ ADH

S_2 : A B C D G H ⇒ ADH

3 ✓

S_1 : A E D F H R

S_2 : A B C D G H

Strings $|x| = m$, $|y| = n$.

LCS:- longest common sub-sequence.

(length)

$X[\ ]$: x_0 x_1 x_2 . . . . x_{m-2} $\underline{x_{m-1}}$ $\updownarrow$ [0 to m-1] ✓

$y[\ ]$: y_0 y_1 y_2 . . . . y_{n-2} $\underline{y_{n-1}}$ [0 to n-1] ✓

x_{m-1} v/s y_{n-1}

equal

$1 + LCS(x_0 \cdots x_{m-2}, y_0 \cdots y_{n-2})$

not equal.

max $\left\{ \begin{array}{l} LCS(x_0 \cdots x_{m-1}, y_0 \cdots y_{n-2}) \text{ (or)} \\ LCS(x_0, \cdots x_{m-2}, y_0 \cdots y_{n-1}) \end{array} \right.$

```
⇒function LCS(x,y,m,n) // len(x)=m, len(y)=n
  {
        if(m==0 || n==0)
              return 0
        if(x[m-1] == y[n-1])
              return 1+LCS(x,y,m-1,n-1)
        return max ( LCS(x,y,m-1,n), LCS(x,y,m,n-1) )
  }
```