Sprint-4 [Day-3]

Nick and Hacks

Ended | Edit

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?

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.

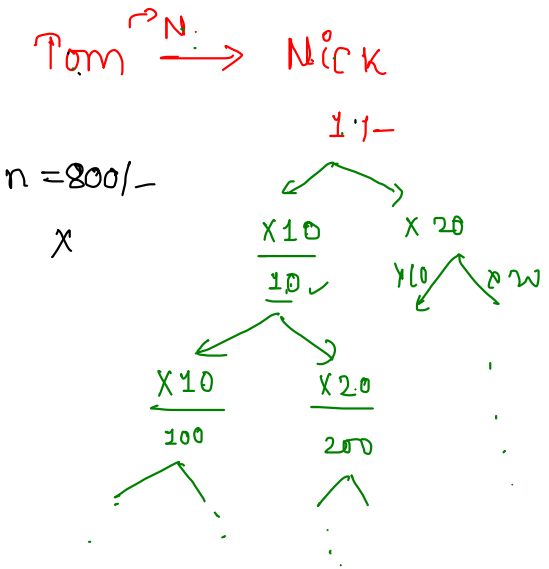
Constraints :

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.



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

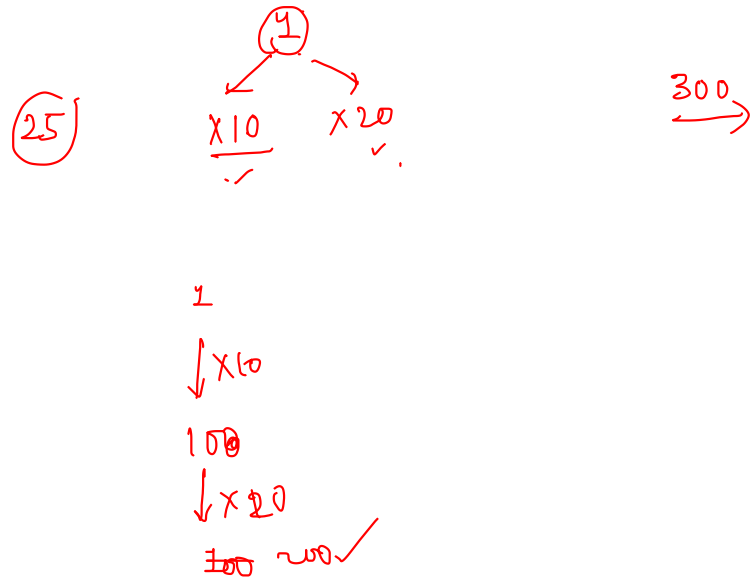
```
Yes
No
Yes
No
Yes
```

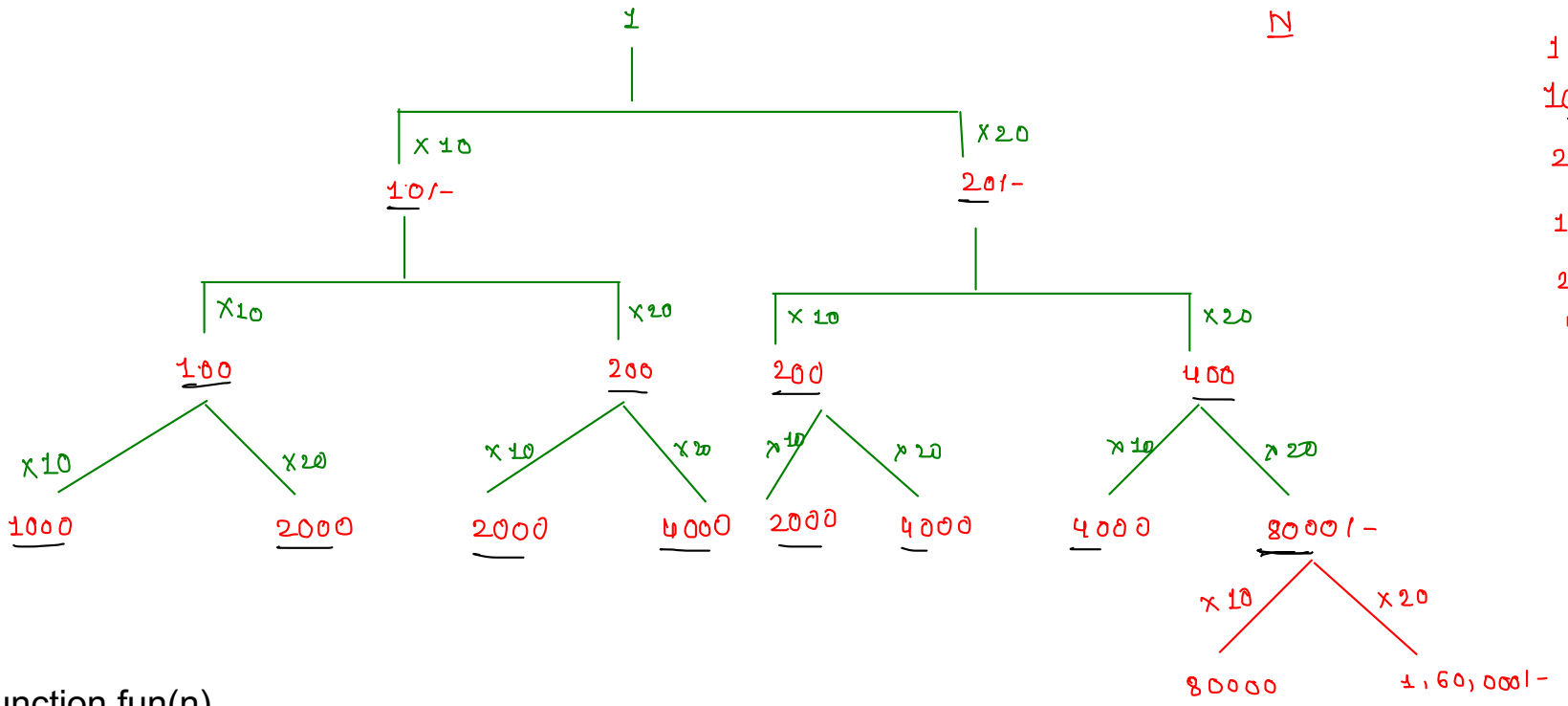
Hint

Output Explanation :

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

1 -> 10 -> 200





N

O/P

1 . True ✓

10 → 1 ✓

20 → 2

100 → 4

200 → 2

400 → 4

1000 → 1

2000 → 2

4000 → 4

8000 → 8

. . . 16

32

64

1, 2, 4, 8, 16, 32 64

```
function fun(n)
{
        if(n==1) return true;
      ✓while(n%10==0)
        {
                n=n/10;
        }
        return  ispowerOf2(n);
}
function ispowerOf2(n)
{
        // logic        ✓T/F
}
```

N = 50 → 5 → False ✓

N = 4000 ⇒ 4

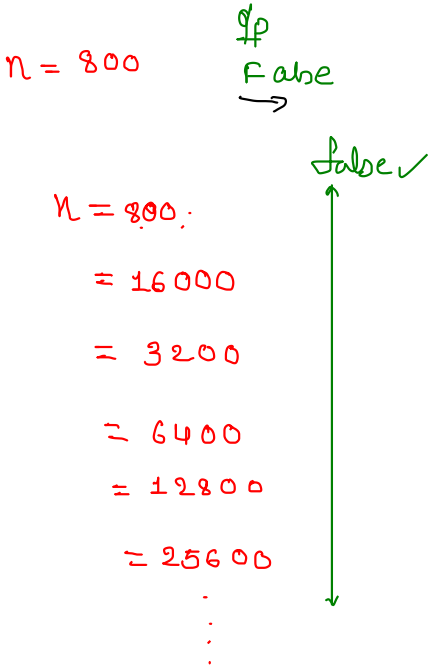
/10 → 400 → /10 → 40 → /10 → 4

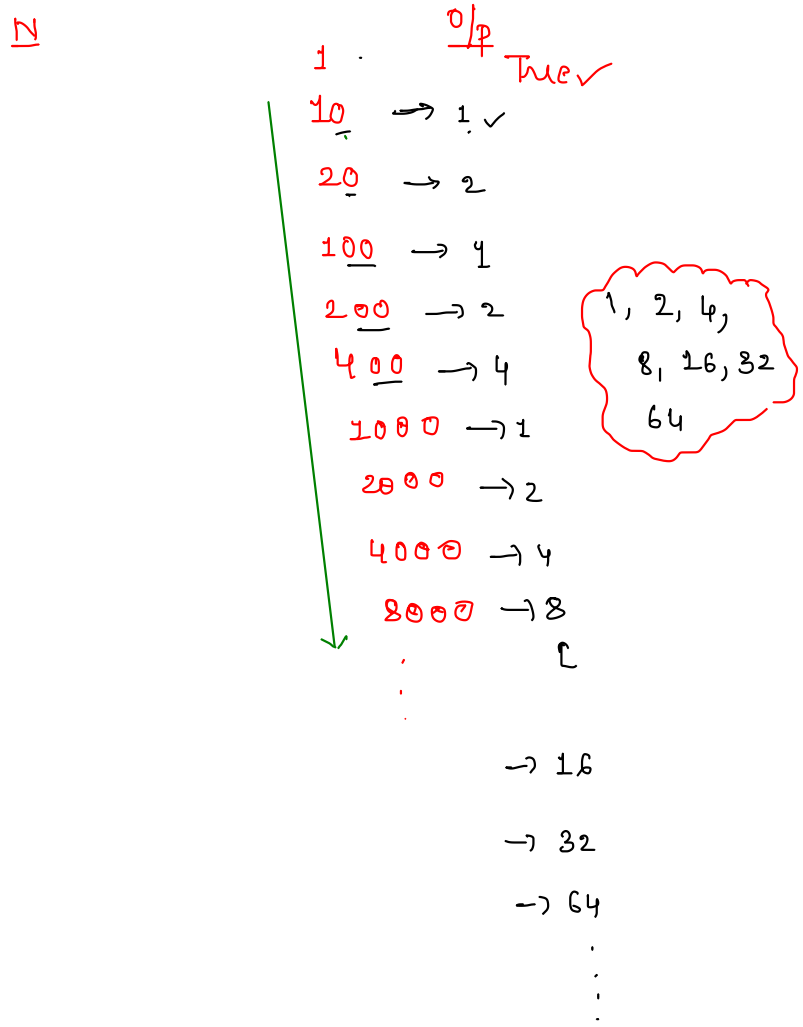
↳ (N%10==0) ✓

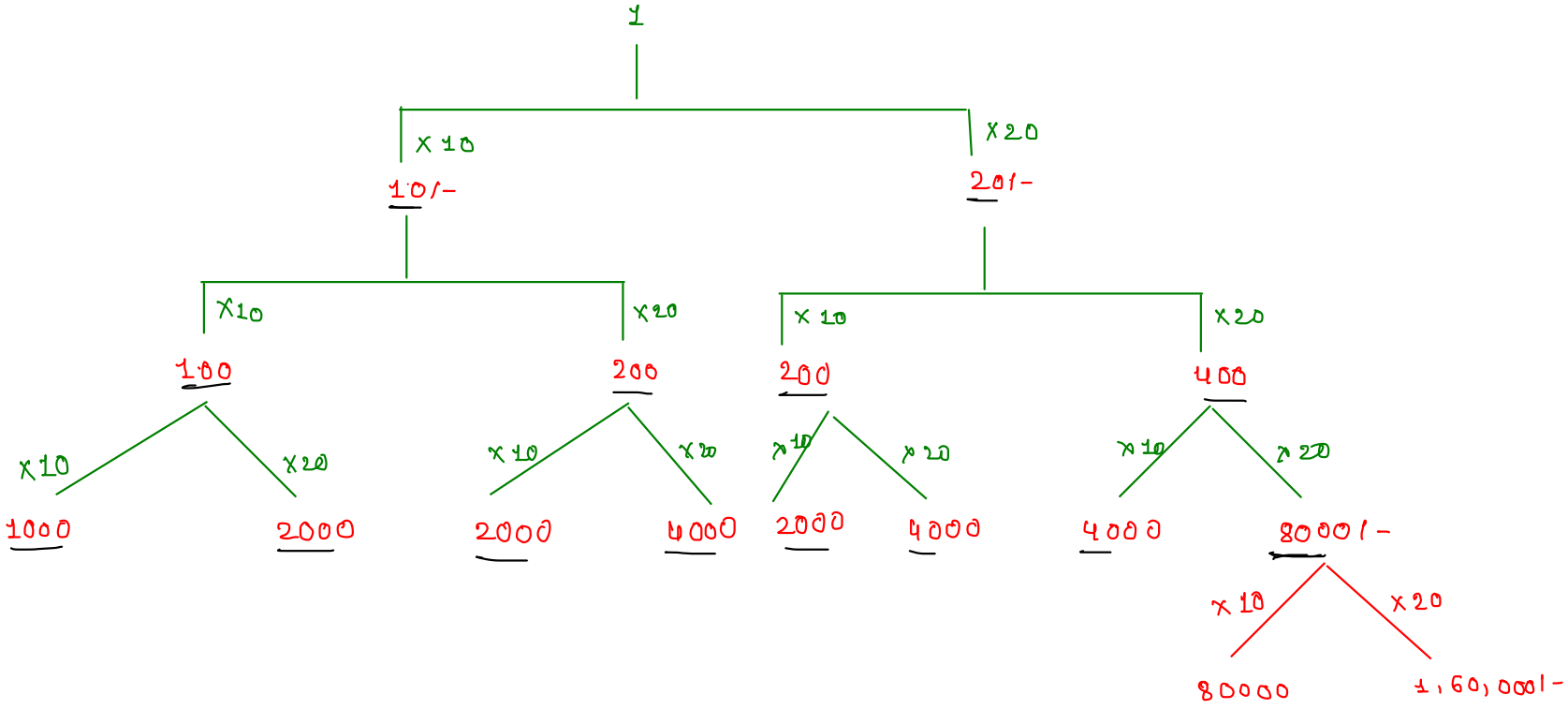
4 % 10 = 4

Logic :-

```
function fun(n)
{
        if(n==1) return true;
        while(n%10==0)
        {
                n=n/10;
        }
        return  ispowerOf2(n);
}
function ispowerOf2(n)
{
        // logic
}
```







10 → $2^0 0$ ⇒ $2^0 \cdot 10^1$

20 → $2^1 0$ ⇒ $2^1 \cdot 10^1$

100 → $2^0 00$ → $2^0 \cdot 10^2$

200 → $2^1 00$ → $2^1 \cdot 10^2$

400 → $2^2 00$ → $2^2 \cdot 10^2$

1000 → $2^0 000$ → $2^0 \cdot 10^3$

2000 → $2^1 000$ → $2^1 \cdot 10^3$

4000 → $2^2 000$ → $2^2 \cdot 10^3$

8000 → $2^3 000$ → $2^3 \cdot 10^3$

.
.
.

Accepted N values

	C_0	C_2
$10 \rightarrow 2^0 0 \Rightarrow 2^0 \cdot 10^1$	1	0
$20 \rightarrow 2^1 0 \Rightarrow 2^1 \cdot 10^1$	1	1
$100 \rightarrow 2^0 00 \rightarrow 2^0 \cdot 10^2$	2	0
$200 \rightarrow 2^1 00 \rightarrow 2^1 \cdot 10^2$	2	1
$400 \rightarrow 2^2 00 \rightarrow 2^2 \cdot 10^2$	2	2
$1000 \rightarrow 2^0 000 \rightarrow 2^0 \cdot 10^3$	3	0
$2000 \rightarrow 2^1 000 \rightarrow 2^1 \cdot 10^3$	3	1
$4000 \rightarrow 2^2 000 \rightarrow 2^1 \cdot 10^3$	3	2
$8000 \rightarrow 2^3 000 \rightarrow 2^3 \cdot 10^3$	3	3

$C_0 \geq C_2$

$C_0 < C_2$

Not Accepted

$800 \Rightarrow 2^3 \cdot 10^2$ — $C_0 = 2$, $C_2 = 3$

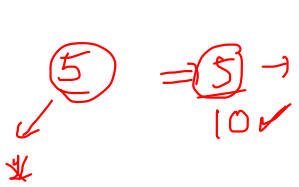
$1600 \Rightarrow 2^4 \cdot 10^2$ — $C_0 = 2$, $C_2 = 4$

$40 \Rightarrow 2^2 \cdot 10^1$ — $C_0 = 1$, $C_2 = 2$

$N = 625$

$N = 6 \cdot 25$

$N = 38$ ✓




```
function fun(n)
{
    let c0=0,c2=0

    while(n%10==0)
    {
        n=n/10;
        c0++;
    }

    while(n%2==0)
    {
        n=n/2;
        c2++;
    }

    return (n==1 && c0>=c2)
}
```

$n = 8000$ 800 80 8

$c_0 = 0 \nearrow 3$

$c_2 = 0$

⇒ Code → W.A. ⇒ Dry run

$n = 8, 4, 2, 1$

$c_0 = 3$

$c_2 = 0 \nearrow 3$

$n = 1$

Accepted

$n = 8000 = 2^3 \cdot 10^3$

$c_2 = 3$ ↵ ↳ $c_0 = 3$

$c_0 = 3$

$c_2 = 3$

$n = 1,60,000$

$c_0 = 4$

$n = 16, 8, 4, 2, 1$

$c_2 = 4$

Not Accepted.

$n = 25 \rightarrow$ False

$c_0 = 0.$

$c_2 = 0.$

Maximum Apples

-33:21:44 Edit

Description

You have some apples and a basket that can carry up to W units of w eight.

Given an integer array weight of size N where weight[i] is the weight of the ith apple, return the maximum number of apples you can put in th e basket.

Input

Input Format

The first line contains the number of apples N and the weight that the basket can carry W

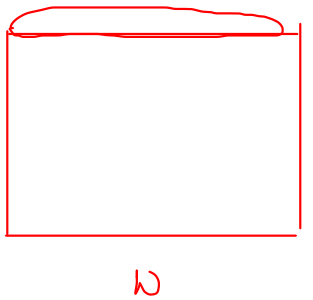
The second line contains N integers as weight of the apples.

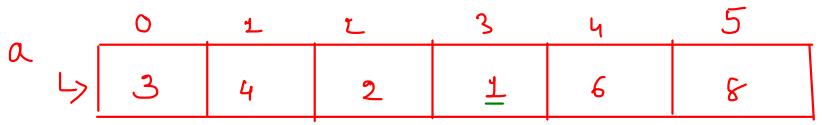
Constraints

1<=N<=10^6

1<=W<=10^9

1<=weight[i]<=1000

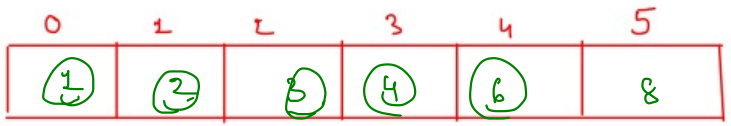




n = 6

C = 15

} ⇒ ~~max~~ - Applus more



temp = 1 + 2 + 3 + 4 + 6

temp > C

bw

cnt = 1 2 3 4

↳ ✓

Remove to Sort

-33:17:18 Edit

Description

Given an array of integers of length n, the task is to remove elements f rom the array to make array sorted. That is, remove the elements whic h do not follow an increasing order.

Input

Input Format

First line Consists of integer n.

Second line consists of n integers separated by spaces.

Constraints

1 <= n <= 10^6

Output

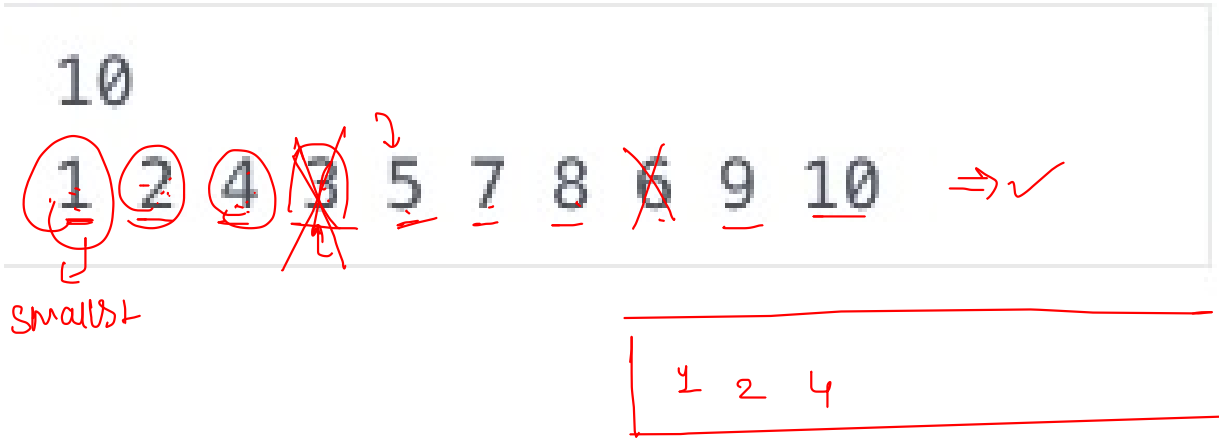
Print the sorted array.

Sample Input 1

```
10
1 2 4 3 5 7 8 6 9 10
```

Sample Output 1

```
1 2 4 5 7 8 9 10
```




```
function fun(arr,n)
{
        res=[]
        temp=-Inifinity
        for(i=0;i<n;i++)
        {
                if(arr[i]>=temp)
                {
                        res.push(arr[i])
                        temp=arr[i]
                }
        }
        return arr;
}
```

→ Variable SW ✓

→ 2ptr (⇉ ⇉)

→ S/W ⇄ 2ptr

⇒ master - class

30

V.D ✓

2ptr ⇄ SW ✓

2:30

8:00

→ Constraints (BF/optimize) ⇒ T.C - 4 ✓ Notes

Last - 8th