REC-CIS

Constraints

1<=T<=1000

1<=N<=10000

Sample Input and Output

Input

3
1
6
7

Output

Yes
Yes
No

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int T,i=0,n,t;
    scanf("%d",&T);
    while(i<T)
    {
        scanf("%d",&n);
        t=n/4;
        if(t%2==0&&n%2==0)
        printf("No\n");
        else if(t%2==1&&n%2==1)
        printf("No\n");
        else
        printf("Yes\n");
        i++;
    }
    return 0;
}
```

Sample Output

2

Explanation

Add the holes count for each digit. 6, 3 and 0. Return 1 + 0 + 1 = 2.

Sample Case 1

Sample Input

1288

Sample Output

4

Explanation

Add the holes count for each digit. 1, 2, 8, 8. Return 0 + 0 + 2 + 2 = 4.

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int a,b,n=0;
    scanf("%d",&a);
    while(a>0)
    {
        b=a%10;
        if(b==0||b==6||b==9||b==4)
        n+=1;
        else if(b==8)
        n+=2;
        a=a/10;
    }
    printf("%d",n);
    return 0;
}
```

	Input	Expected	Got	
✓	630	2	2	✓
✓	1288	4	4	✓

Passed all tests! ✓

3

Explanation:

For test case 1, N=10.

According to Manish ($1, $2, $3.... $10) must be distributed.

But as per Manisha only ($1, $2, $3, $4) coins are enough to purchase any item ranging from $1 to $10. Hence minimum is 4. Likewise denominations could also be ($1, $2, $3, $5). Hence answer is still 4.

For test case 2, N=5.

According to Manish ($1, $2, $3, $4, $5) must be distributed.

But as per Manisha only ($1, $2, $3) coins are enough to purchase any item ranging from $1 to $5. Hence minimum is 3. Likewise, denominations could also be ($1, $2, $4). Hence answer is still 3.

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int n,r=0;
    scanf("%d",&n);
    while(n!=0)
    {
        n=n/2;
        r=r+1;
    }
    printf("%d",r);
    return 0;
}
```

	Input	Expected	Got	
✓	10	4	4	✓
✓	5	3	3	✓
✓	20	5	5	✓
✓	100	7	7	✓
✓	1000	10	10	✓

Passed all tests! ✓

Finish review

Output Format:

The count of numbers where the numbers are odd numbers.

Example Input / Output 1:

Input:

5 10 15 20 25 30 35 40 45 50

Output:

5

Explanation:

The numbers meeting the criteria are 5, 15, 25, 35, 45.

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int n,x=0;
    while(scanf("%d",&n)--1)
    {
        if(n%2!=0)
        {
            x++;
        }
    }
    printf("%d",x);
    return 0;
}
```

Example 2:

89 -> 68

Input: 89

Output: true

Explanation:

We get 68 after rotating *89*. 86 is a valid number and 86!=89.

Example 3:

11 -> 11

Input: 11

Output: false

Explanation:

We get 11 after rotating 11. 11 is a valid number but the value remains the same, thus 11 is not a confusing number.

Note:

1. *0 <= N <= 10^9*
2. After the rotation we can ignore leading zeros, for example if after rotation we have 0008 then this number is considered as just 8.

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    int n,x,y=1;
    scanf("%d",&n);
    while(n!=0&&y==1)
    {
        x=n%10;
        n=n/10;
        if(x==2||x==3||x==7)
        {
            y++;
        }
    }
    if(y==1)
    {
      printf("true");
    }
    else
    {
       printf("false");
    }
    return 0;
}
```

	Input	Expected	Got	
✓	6	true	true	✓
✓	89	true	true	✓
✓	26	false	false	✓

Passed all tests! ✓

Question **3**

Correct

A nutritionist is labeling all the best power foods in the market. Every food item can

2. Hence, max total is achieved by sum = 0 + 2 = 2.

Sample Case 2

Sample Input For Custom Testing

Sample Input 2

3

3

Sample Output 2

5

Explanation 2

2 + 3 = 5, is the best case for maximum nutrients.

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main()
{
    long long int n,t,i,nut=0;
    scanf("%lld %lld",&n,&t);
    for(i=1;i<=n;i++)
    {
        nut=nut+i;
        if(nut==t)
        {
            nut=nut+1;
        }
    }
    printf("%lld",nut%1000000007);
    return 0;
}
```