DecisionMakingandLooping– while and do…while, for

Ex.No.:   **Date:**

StoneGame-OneFour

ProblemStatement:

AliceandBobareplayingagamecalled"StoneGame".Stonegameisatwo-playergame. LetNbethetotalnumberofstones.Ineachturn,aplayercanremoveeitheronestoneor four stones. The player who picks the last stone, wins. They follow the "Ladies First" norm. HenceAliceisalwaystheonetomakethefirstmove.YourtaskistofindoutwhetherAlice can win, if both play the game optimally.

InputFormat
First line starts with T, which is the number of test cases. Each test case will contain N number of stones.

OutputFormat
Print"Yes"inthecaseAlicewins,elseprint"No".

Constraints1<=T<=10001<=N<=10000

SampleInput
3
1
6
7

SampleOutput
Yes
Yes
No

Program:

```c
1  #include<stdio.h>
2  int main()
3  {
4      int T,t,n,i=0;
5      scanf("%d",&T);
6      while(i<T)
7      {
8        scanf("%d",&n);
9        t=n/4;
10       if(t%2==0&&n%2==0)
11        printf("No\n");
12       else if(t%2==1&&n%2==1)
13        printf("No\n");
14       else
15        printf("Yes\n");
16       i++;
17
18
19     }
20 }
```

	Input	Expected	Got	
✓	3 1 6 7	Yes Yes No	Yes Yes No	✓

Passed all tests! ✓

Ex.No.:   **Date:**

HolesinaNumber

ProblemStatement:

Youaredesigningaposterwhichprintsoutnumberswithauniquestyleappliedtoeach   of them. The styling is based on the number of closed paths or holes present in a given number.

The number of holes that each of the digits from 0 to 9 have are equal to the number of closed paths in the digit. Their values are:

1,2,3,5,7	=0 holes.
0,4,6,9	=1 hole.
8	=2 holes.

Givenanumber,youmustdeterminethesumofthenumberofholesforallofitsdigits. For example, the number 819 has 3 holes.

Completetheprogram,itmustreturnanintegerdenotingthetotalnumberofholesin num.

Constraints
1≤num≤109

InputFormatForCustomTesting
Thereisonelineoftextcontainingasingleintegernum,thevaluetoprocess.

SampleInput
630

SampleOutput
2

Program:

```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)
         n=n+1;
         else if(b==8)
         n=n+2;
         a=a/10;
     }printf("%d",n);

}
```

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

Passed all tests! ✓

Ex.No.:   **Date:**

PhilalandCoin

ProblemStatement:

The problem solvers have found a new Island for coding and named it as Philaland. These smart people were given a task to make a purchase of items at the Island easier by distributing various coins with different values. Manish has come up with a solution that if we make coins category starting from $1 till the maximum price of the item present on Island,thenwecanpurchaseanyitemeasily.Headdedthefollowingexampletoprove his point.

Let'ssupposethemaximumpriceofanitemis5$thenwecanmakecoinsof{$1,$2,$3, $4,$5}topurchaseanyitemrangingfrom$1till$5.

Now Manisha, being a keen observer suggested that we could actually minimize thenumber of coins required and gave following distribution {$1, $2, $3}. According to himany item can be purchased one time ranging from $1 to $5. Everyone was impressed with both of them. Your task is to help Manisha come up with a minimum number of denominations for any arbitrary max price in Philaland.

InputFormat
ContainsanintegerNdenotingthemaximumpriceoftheitempresentonPhilaland.

OutputFormat
Printasinglelinedenotingtheminimumnumberofdenominationsofcoinsrequired.

Constraints
1<=T<=1001<=N<=5000

SampleInput1:
10

SampleOutput1:
4

Program

```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)
         n=n+1;
         else if(b==8)
         n=n+2;
         a=a/10;
    }printf("%d",n);

}
```

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

Passed all tests! ✓

Ex.No.:   **Date:**

NumberCount

ProblemStatement:

AsetofNnumbers(separatedbyonespace)ispassedasinputtotheprogram.The program must identify the count of numbers where the number is odd number.

InputFormat:
ThefirstlinewillcontaintheNnumbersseparatedbyonespace.

BoundaryConditions:
3<=N<=50
Thevalueof thenumbers canbefrom-99999999to 99999999

OutputFormat:
Thecountofnumberswherethenumbersareoddnumbers.

SampleInput:
5 1015 20 253035404550

SampleOutput:
5

Program:

```c
#include<stdio.h>
int main()
{
    int n,c=0;
    while(scanf("%d",&n)==1)
    {

        if(n%2!=0)
         c++;

    }printf("%d",c);
}
```

Input	Expected	Got	
5 10 15 20 25 30 35 40 45 50	5	5	✓

Passed all tests! ✓

Ex.No.:   **Date:**

ConfusingNumber

ProblemStatement:

Given a number N, return true if and only if it is a *confusing number*, which satisfies the following condition:
Wecanrotatedigitsby180degreestoformnewdigits.When0,1,6,8,9arerotated180 degrees, they become 0, 1, 9, 8, 6 respectively. When 2, 3, 4, 5 and 7 are rotated 180 degrees, they become invalid. A *confusing number* is a number that when rotated 180 degrees becomes a **different** number with each digit valid.

Example1:

Input: 6
Output: true
Explanation:Weget9afterrotating6,9isavalidnumberand9!=6.

Example2:

Input: 89
Output: true
Explanation:Weget68afterrotating89,86isavalidnumberand86!=89.

Example3:

Input: 11
Output:false
Explanation:Weget11afterrotating11,11isavalidnumberbutthevalueremainsthe same, thus 11 is not a confusing number.

Example4:

Input: 25
Output:false
Explanation:Wegetaninvalidnumberafterrotating25.
Note:
   1. 0<=N<=10^9
   2. Aftertherotationwecanignoreleadingzeros,forexampleifafterrotationwehave 0008thenthisnumberisconsideredasjust8.

Program:

```c
#include<stdio.h>
int main()
{
    int x,y=1,n=1;
    scanf("%d",&n);
    while(n!=0&&y==1)
    {
        x=n%10;n=n/10;
        if(x==2||x==3||x==4||x==7)
         y++;

    }
    if(y==1)
     printf("true");
    else
     printf("false");
}
```

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

Passed all tests! ✓

Ex.No.:   Date:

NutritionValue

ProblemStatement:

A nutritionist is labeling all the best power foods in the market. Every food item arranged in a single line, will have a value beginning from 1 and increasing by 1 for each, until all itemshaveavalue associatedwith them.An item'svalueisthesameas thenumber of macronutrients it has. For example, food item with value 1 has 1 macronutrient, food item with value 2 has 2 macronutrients, and incrementing in this fashion.

The nutritionist has to recommend the best combination to patients, i.e. maximum total of macronutrients. However, the nutritionist must avoid prescribing a particular sum ofmacronutrients(an'unhealthy'number),andthissumisknown.Thenutritionistchooses food items in the increasing order of their value. Compute the highest total of macronutrients that can be prescribed to a patient, without the sum matching the given 'unhealthy' number.

Here's an illustration: Given 4 food items (hence value:$1,2,3$and4), and the unhealthy sum being 6 macronutrients, on choosing items $1, 2, 3$-> the sum is 6,which matches the 'unhealthy' sum. Hence, one of the three needs to be skipped. Thus, the best combination is from among:
- $2+3+4=9$
- $1+3+4=8$
- $1+2+4=7$

Since $2 + 3 + 4 = 9$, allows for maximum number of macronutrients, 9 is the right answer. Complete the code in the editor below. It must return an integer that representsthe maximum total of macronutrients, modulo $1000000007 (10^9 + 7)$.
Ithasthefollowing:
*n:*anintegerthatdenotesthenumberoffooditems
*k:*anintegerthatdenotestheunhealthynumber

Constraints
- $1≤n≤2×10^9$
- $1≤k≤4×10^{15}$

InputFormatForCustomTesting
Thefirstlinecontainsaninteger,*n*,thatdenotesthenumberoffooditems.Thesecondline contains aninteger, *k*, that denotes theunhealthy number.

SampleInput0
2
2

SampleOutput0
3

Program

```
1    #include<stdio.h>
2    int main()
3    {
4        long long int n,t,i,nut=0;
5        scanf("%lld %lld ",&n,&t);
6        for(i=1;i<=n;i++)
7        {
8            nut=nut+i;
9            if(nut==t){
10           nut=nut-1;}
11
12       }printf("%lld",nut % 1000000007);
13       return 0;
14   }
15
```

	Input	Expected	Got	
✓	2 2	3	3	✓
✓	2 1	2	2	✓
✓	3 3	5	5	✓

Passed all tests! ✓