1. **RainDrop**

Your task is to convert a number into a string that contains raindrop sounds corresponding to certain potential factors. A factor is a number that evenly divides into another number, leaving no remainder.

The rules of raindrops are that if a given number:

has 3 as a factor, add 'Pling' to the result.

has 5 as a factor, add 'Plang' to the result.

has 7 as a factor, add 'Plong' to the result.

does not have any of 3, 5, or 7 as a factor, the result should be the digits of the number.

Explanation:

28 has 7 as a factor, but not 3 or 5, so the result would be "Plong".

30 has both 3 and 5 as factors, but not 7, so the result would be "PlingPlang".

34 is not factored by 3, 5, or 7, so the result would be "34".

Input Format:

Single line containing an integer N.

Output Format:

Based on the given rules of raindrop print result.

2. **Elephant and his friend**

An elephant decided to visit his friend. It turned out that the elephant's house is located at point 0 and his friend's house is located at point ***x*** (x > 0) of the coordinate line. In one step the elephant can move 1, 2, 3, 4 or 5 positions forward. Determine, what is the ***minimum number of steps*** he needs to make in order to get to his friend's house.

Note: Check out the Sample I/O for more clarity.

Input Format:
The only line of input contains ***x*** as mentioned in the problem statement.

Output Format:
Print the minimum number of steps that elephant needs to make to get from point 0 to point x.

Constraints:
$1 \leq x \leq 100000$

Sample I/O:

Input 1:
5
Output 1:
1

Input 2:
12
Output 2:
3

Input 3:
147
Output 3:
30

Input 4:
25
Output 4:
5

Input 5:
29
Output 5:
6

Explanation:
For Input 2,
Elephant can move 5 positions forwards in 1st step, 5 more positions forward in 2nd step, and 2 positions forward in 3rd step to reach point 12. So a total of 3 steps is required.

For Input 5,
Elephant can reach point 25 in just 5 steps by moving 5 positions forward in each step. From where only 1 more step (4 positions forward) is required to reach point 29.

3. Elections at Hogwarts

There are 101 students at **Hogwarts**. It is election time and 3 parties, **Gryffindor, Slytherin** and **Hufflepuff** are contesting the elections. Party **Gryffindor** receives ***X*** votes, party **Slytherin** receives ***Y*** votes and party **Hufflepuff** receives ***Z*** votes.

The constitution of **Hogwarts** requires a particular party to receive a clear majority to form the government. A party is said to have a clear marjority if it receives **strictly** greater than 50 votes.

If any party has clear majority print the winning party (**Gryffindor, Slytherin or Hufflepuff**). Othewise print **NOTA**.

Input Format:
Only line of input contains three integers ***X***, ***Y*** and ***Z***.

Output Format:
Print output according to the description.

Constraints:
$0 \leq X, Y, Z \leq 101$
$X + Y + Z = 101$

Sample I/O:

Input 1:
80 19 2
Output 1:
Gryffindor

Input 2:
20 55 26
Output 2:
Slytherin

Input 3:
50 1 50
Output 3:
NOTA

Input 4:
0 0 101
Output 4:
Hufflepuff

4. Can be a factor?

Madmax has two integers A and B (A ≤ B).

He can choose any **non-negative** integer X and add it to both A and B. Find whether it is possible to make **A a divisor of B**.

See the Sample I/O and Explanation for more clarity.

Input Format:

- The first line of input will contain a single integer T, denoting the number of test cases.
- Each test case consists of two integers A and B.

Output Format:

For each test case, output **YES** if it is possible to make A a factor of B, **NO** otherwise.

Constraints:

- 1 ≤ T ≤ 105
- 1 ≤ A ≤ B ≤ 109

Sample I/O:

Input 1:
3
3 6
4 14
9 11
Output 1:
YES
YES
NO

Input 2:
5
7 16
21 27
16 33
10 10
130 259
Output 2:
YES
NO
YES
YES
NO

Explanation:
In Input 1,
for Testcase 1, we can choose X = 0 and can add it to both 3 and 6. Now 3 is a factor of 6.
for Testcase 2, we can choose X = 1 and can add it to both 4 and 14 making them 5 and 15. 5 is a factor 15.
for Testcase 3, there not possible value of X to add such that A becomes a factor of B.

January 24 ,2024

1. Rain Drop

```
import java.io.*;
import java.util.*;
public class RainDrop
{
   public static void main(String args[])
   {
      int number;
      Scanner s=new Scanner(System.in);
      // Reading a number
      number = s.nextInt();
      // Logic to convert a number into a string contains raindrop sounds corresponding to
      //   certain potential factors and prints the corresponding result.
      if(number % 3 == 0)
            System.out.print("Pling");
     if(number % 5 == 0)
            System.out.print("Plang");
    if(number % 7 == 0)
            System.out.print("Plong");
    if(number % 3 != 0 && number % 5 != 0 && number % 7 != 0)
            System.out.println(number);
    }
}
```

Explanation:

User Input:

The program takes an integer input from the user using the Scanner class.

Factor Check:

The code then checks the input number for its divisibility by 3, 5, and 7.

If the number is divisible by 3, it appends "Pling" to the output.

If divisible by 5, it appends "Plang" to the output.

If divisible by 7, it appends "Plong" to the output.

Output Generation:

If none of the above conditions are met, meaning the number is not divisible by 3, 5, or 7, the original number is printed.

Example:

For an input of 15, the output would be "PlingPlang" since 15 is divisible by both 3 and 5.

2. Elephant and his friend

```
import java.util.Scanner;

public class ElephantAndHisFriend {

   public static void main(String args[]) {

      Scanner sc = new Scanner(System.in);

      int n = sc.nextInt();

       int res = n % 5 == 0 ? (n / 5) : (n / 5 + 1);

       System.out.println(res);

   }

}
```

Explanation:

Move Calculation:

The code calculates the minimum number of moves needed to distribute the candies in groups of 5.

The expression n % 5 == 0 ? (n / 5) : (n / 5 + 1) is a ternary operator that checks if the total number of candies is divisible evenly by 5.

If divisible by 5, it directly divides the total candies by 5 (n / 5).

If not divisible by 5, it adds 1 to the result to account for the remaining candies that cannot form a complete group of 5.

Output:

The result is stored in the variable 'res' and printed to the console using System.out.println(res).

Example:

For an input of 17 candies, the output would be 4, as it requires 4 moves to distribute them in groups of 5 (5 + 5 + 5 + 2).

3. Elections at Hogwarts

```
import java.util.*;

public class Elections
{
   public static void main(String args[])
   {
      Scanner sc=new Scanner(System.in);
      int x,y,z;
      x=sc.nextInt();
      y=sc.nextInt();
      z=sc.nextInt();
      if(x>50)
      System.out.println("Gryffindor");
      else if(y>50)
      System.out.println("Slytherin");
      else if(z>50)
      System.out.println("Hufflepuff");
      else
      System.out.println("NOTA");
   }
}
```

Explanation:

User Input:

The program takes three integers (x, y, and z) as input, representing the number of votes each party (Gryffindor, Slytherin, Hufflepuff) receives.

Clear Majority Check:

The code then checks if any party has a clear majority by comparing the number of votes (x, y, z) to 50.

If the votes for Gryffindor (x) are greater than 50, it prints "Gryffindor."

If the votes for Slytherin (y) are greater than 50, it prints "Slytherin."

If the votes for Hufflepuff (z) are greater than 50, it prints "Hufflepuff."

NOTA (None of the Above):

If none of the parties has a clear majority, it prints "NOTA."

Example:

For an input of 80 19 2, the output would be "Gryffindor" as Gryffindor has a clear majority.

4. Can be a factor:

```
import java.util.*;
public class Factor
{
   public static void main(String args[])
   {
      Scanner sc=new Scanner(System.in);
      int t=sc.nextInt();
      while(t-->0)
      {
         int a,b,z;
         a=sc.nextInt();
         b=sc.nextInt();
         z=Math.abs(a-b);
         if(z>=a||a==b)
         System.out.println("YES");
         else
         System.out.println("NO");
      }
   }
}
```

Explanation:

Test Case Iteration:

The program starts by taking the number of test cases t as input.

Input Processing:

For each test case, it takes two integers a and b as input, representing the numbers in that particular case.

Absolute Difference:

The code calculates the absolute difference z between A and B using z = Math.abs(a - b).

Factor Check:

The program checks whether it is possible to make A a factor of B.

If the absolute difference z is greater than or equal to A (z >= a), or if A is equal to B (a == b), then it prints "YES."

Otherwise, it prints "NO."

Example:

For an input of 3 6, the output would be "YES" because choosing X = 0 and adding it to both 3 and 6 makes 3 a factor of 6