Q1)

Write a program to read two integer values and print true if both the numbers end with the same digit, otherwise print false. Example: If 698 and 768 are given, program should print true as they both end with 8.
Sample Input 1 25 53 Sample Output 1 false Sample Input 2 27 77 Sample Output 2 true

```c
#include <stdio.h>
int main(){
    int a,b;
    scanf("%d %d", &a, &b);
    if (a%10 == b%10){
        printf("true");
    }
    else{
        printf("false");
    }
    return 0;
}
```

	Input	Expected	Got	
✓	25 53	false	false	✓
✓	27 77	true	true	✓

Passed all tests! ✓

Q2)

Objective

In this challenge, we're getting started with conditional statements.

Task

Given an integer, ***n***, perform the following conditional actions:

- If ***n*** is odd, print Weird
- If ***n*** is even and in the inclusive range of ***2*** to ***5***, print ***Not Weird***
- If ***n*** is even and in the inclusive range of ***6*** to ***20***, print ***Weird***
- If ***n*** is even and greater than ***20***, print ***Not Weird***

Complete the stub code provided in your editor to print whether or not ***n*** is weird.

Input Format

A single line containing a positive integer, ***n***.

Constraints

- $1 \leq n \leq 100$

Output Format

Print Weird if the number is weird; otherwise, print Not Weird.

Sample Input 0

3

Sample Output 0

Weird

Sample Input 1

24

Sample Output 1

Not Weird

Explanation

Sample Case 0: ***n = 3***

n is odd and odd numbers are weird, so we print ***Weird***.

Sample Case 1: ***n = 24***

n > 20 and ***n*** is even, so it isn't weird. Thus, we print ***Not Weird***.

```c
#include <stdio.h>
int main(){
    int n;
    scanf("%d", &n);
    if (n%2 == 0){
        if (n>=2 && n<=5){
            printf("Not Weird");
        }
        else if (n>=6 && n<=20){
            printf("Weird");
        }
        else{
            printf("Not Weird");
        }
    }
    else{
        printf("Weird");
    }
    return 0;
}
```

	Input	Expected	Got	
✓	3	Weird	Weird	✓
✓	24	Not Weird	Not Weird	✓

Passed all tests! ✓

Q3)

Three numbers form a Pythagorean triple if the sum of squares of two numbers is equal to the square of the third. For example, 3, 5 and 4 form a Pythagorean triple, since 3*3 + 4*4 = 25 = 5*5 You are given three integers, a, b, and c. They need not be given in increasing order. If they form a Pythagorean triple, then print "yes", otherwise, print "no". Please note that the output message is in small letters. Sample Input 1 3 5 4 Sample Output 1 yes Sample Input 2 5 8 2 Sample Output 2 no

```c
#include <stdio.h>
int main(){
    int a,b,c;
    scanf("%d %d %d", &a, &b, &c);
    if ((a*a +b*b) == c*c){
        printf("yes");
    }
    else if(a*a + c*c == b*b){
        printf("yes");
    }
    else if(b*b + c*c == a*a){
        printf("yes");
    }
    else{
        printf("no");
    }
    return 0;
}
```

	Input	Expected	Got	
✓	3 5 4	yes	yes	✓
✓	5 8 2	no	no	✓

Passed all tests! ✓

Q4)

Write a program that determines the name of a shape from its number of sides. Read the number of sides from the user and then report the appropriate name as part of a meaningful message. Your program should support shapes with anywhere from 3 up to (and including) 10 sides. If a number of sides outside of this range is entered then your program should display an appropriate error message.

Sample Input 1

3

Sample Output 1

Triangle

Sample Input 2

7

Sample Output 2

Heptagon

Sample Input 3

11

Sample Output 3

The number of sides is not supported.

```c
#include <stdio.h>
int main(){
    int n;
    scanf("%d", &n);
    switch (n){
        case 3:
        printf("Triangle");
        break;
        case 4:
        printf("Quadrilateral");
        break;
        case 5:
        printf("Pentagon");
        break;
        case 6:
        printf("Heaxagon");
        break;
        case 7:
        printf("Heptagon");
        break;
        case 8:
        printf("Octagon");
        break;
        case 9:
        printf("Nonagon");
        break;
        case 10:
        printf("Decagon");
        break;
        default:
        printf ("The number of sides is not supported.");
        break;
    }
    return 0;
}
```

	Input	Expected	Got	
✓	3	Triangle	Triangle	✓
✓	7	Heptagon	Heptagon	✓
✓	11	The number of sides is not supported.	The number of sides is not supported.	✓

Passed all tests! ✓

Q5)

The Chinese zodiac assigns animals to years in a 12-year cycle. One 12-year cycle is shown in the table below. The pattern repeats from there, with 2012 being another year of the Dragon, and 1999 being another year of the Hare.

Year	Animal
2000	Dragon
2001	Snake
2002	Horse
2003	Sheep
2004	Monkey
2005	Rooster
2006	Dog
2007	Pig
2008	Rat
2009	Ox
2010	Tiger
2011	Hare

Write a program that reads a year from the user and displays the animal associated with that year. Your program should work correctly for any year greater than or equal to zero, not just the ones listed in the table.

Sample Input 1

2004

Sample Output 1

Monkey

Sample Input 2

2010

Sample Output 2

Tiger

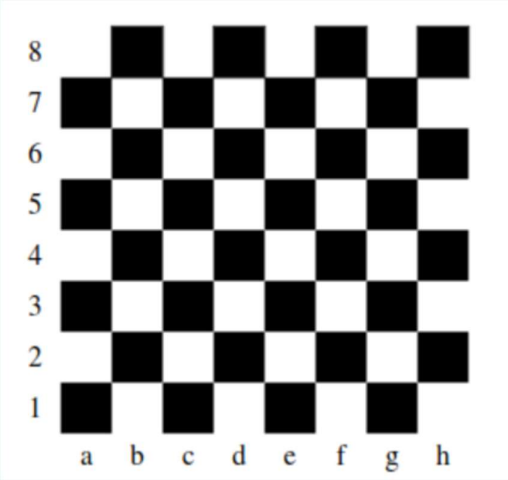
```c
#include <stdio.h>
int main(){
    int year;
    scanf("%d", &year);
    if (year%12 == 8)
    printf("Dragon");
    else if (year%12 == 9)
    printf("Snake");
    else if (year%12 == 10)
    printf("Horse");
    else if (year%12 == 11)
    printf("Sheep");
    else if (year%12 == 0)
    printf("Monkey");
    else if (year%12 == 1)
    printf("Rooster");
    else if (year%12 == 2)
    printf("Dog");
    else if (year%12 == 3)
    printf("Pig");
    else if (year%12 == 4)
    printf("Rat");
    else if (year%12 == 5)
    printf("Ox");
    else if (year%12 == 6)
    printf ("Tiger");
    else
    printf("Hare");
    return 0;

}
```

	Input	Expected	Got	
✓	2004	Monkey	Monkey	✓
✓	2010	Tiger	Tiger	✓

Passed all tests! ✓

Q6)

Positions on a chess board are identified by a letter and a number. The letter identifies the column, while the number identifies the row, as shown below:



Write a program that reads a position from the user. Use an if statement to determine if the column begins with a black square or a white square. Then use modular arithmetic to report the color of the square in that row. For example, if the user enters a1 then your program should report that the square is black. If the user enters d5 then your program should report that the square is white. Your program may assume that a valid position will always be entered. It does not need to perform any error checking.

Sample Input 1

a 1

Sample Output 1

The square is black.

Sample Input 2

d 5

Sample Output 2

The square is white.

```c
#include <stdio.h>
int main(){
    int x,y;
    char z;
    scanf("%c %d", &z, &x);
    y = x+z;
    if (y%2 == 0)
    printf("The square is black.");
    else
    printf("The square is white.");
    return 0;
}
```

	Input	Expected	Got	
✓	a 1	The square is black.	The square is black.	✓
✓	d 5	The square is white.	The square is white.	✓

Passed all tests! ✓