Java code

1. 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  is weird.

**Input Format**

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

**Constraints**

* 1<=n<=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  is even, so it isn't weird. Thus, we print Not Weird.

----------------------------------------------\*\*\*\*\*\*\*\*\*\*\*\*\*\*-----------------------------------------

2. Given an integer, N , print its first  10 multiples. Each multiple N (where 1<=i<=10) should be printed on a new line in the form: N x i = result.

**Input Format**

A single integer, .

**Constraints**

* 2<=N<=20

**Output Format**

Print  10 lines of output; each line i (where 1<=i<=10 ) contains the Nxi of  in the form:  
N x i = result.

**Sample Input**

2

**Sample Output**

2 x 1 = 2

2 x 2 = 4

2 x 3 = 6

2 x 4 = 8

2 x 5 = 10

2 x 6 = 12

2 x 7 = 14

2 x 8 = 16

2 x 9 = 18

2 x 10 = 20

-------------------------------------------\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*-------------------------------------------------------------

3. Given an input integer, you must determine which primitive data types are capable of properly storing that input.

**Input Format**

The first line contains an integer, , denoting the number of test cases.  
Each test case, , is comprised of a single line with an integer, , which can be arbitrarily large or small.

**Output Format**

For each input variable  and appropriate primitive , you must determine if the given primitives are capable of storing it. If yes, then print:

n can be fitted in:

\* dataType

If there is more than one appropriate data type, print each one on its own line and order them by size (i.e.: ).

If the number cannot be stored in one of the four aforementioned primitives, print the line:

n can't be fitted anywhere.

**Sample Input**

5

-150

150000

1500000000

213333333333333333333333333333333333

-100000000000000

**Sample Output**

-150 can be fitted in:

\* short

\* int

\* long

150000 can be fitted in:

\* int

\* long

1500000000 can be fitted in:

\* int

\* long

213333333333333333333333333333333333 can't be fitted anywhere.

-100000000000000 can be fitted in:

\* long

**Explanation**

 can be stored in a short, an int, or a long.

 is very large and is outside of the allowable range of values for the primitive data types discussed in this problem.