1. Sum range 1: What is the output of this code?

print(sum(range(0, 7)))

1. 15
2. 21
3. 28
4. 36
5. Check even/odd-

**Problem Description**

Write a program to input an integer from user and print **1** if it is odd otherwise print **0**.

**Problem Constraints**

1 <= N <= 1000000

**Input Format**

One line containing an integer **N**.

**Output Format**

Print either 1 or 0 as per the question.

**Example Input**

Input 1:

5

Input 2:

1000

**Example Output**

Output 1:

1

Output 2:

0

1. Divisible by 2 numbers-

**Problem Description**

Take an integer **A** as input. You have to tell whether **A** is divisible by **both 5 and 11** or not.

**Problem Constraints**

1 <= **A** <= 109

**Input Format**

The input contains a single integer **A**.

**Output Format**

Print 1 if **A** is divisible by both 5 and 11, else print 0.

**Example Input**

Input 1:

55

Input 2:

22

**Example Output**

Output 1:

1

Output 2:

0

**Example Explanation**

Explanation 1:

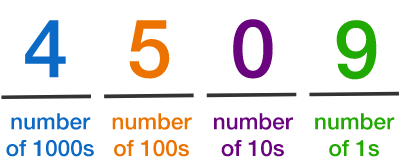
55 is divisible by both 5 (5 \* 11 = 55) and 11 (11 \* 5 = 55).

Explanation 2:

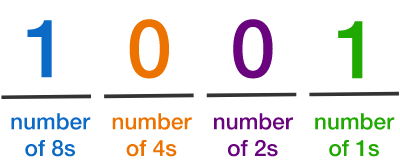
22 is divisible by 11 (11 \* 2 = 22),but it is not divisible by 5.

1. Binary to Decimal

In our base-10 system, the positions (from right to left) correspond to the "number of 1s", "number of 10s", "number of 100s", and so on, where the value of each position is **10 times** the previous one.



Binary, or base 2, uses only the digits 0 and 1. Each position corresponds to the "number of 1s", "number of 2s", "number of 4s", "number of 8s", and so on. The value of each position is **2 times** the previous one.



The number mentioned above if expressed in base 10 will give 8 + 1 = 9 since there is one 8 and one 1.

**What is the value (in base 10) of the binary number 10000?**

1. **12**
2. **14**
3. **16**
4. **20**
5. **8**
6. Larger Binary – 1:

Which binary number out of the options given below is larger?

1. 1111
2. 10000
3. Both are having the same value
4. Binary Ending in 1-

If a binary number ends in 1 (like 1001 or 111011), then it is…

1. Always Odd
2. Always Even
3. May be odd or may be even, needs to check other digits as well.
4. Binary Multiply –

What is the result (in binary) of multiplying the binary number 1100 by the binary number 10?

a. 1110

b. 1111

c. 11000

d. 11111

**Note**: Type the binary number (1110 or 1111 etc.), NOT the option (a or b etc) in the answer.

1. Binary Multiply – 1:

What is the result (in binary) of making the binary number 1001 double?

1. 1010
2. 11000
3. 10100
4. 10010
5. Larger Logarithm – Which of these two numbers is larger?
6. log10(1000)
7. log2(9)
8. Both are equal
9. Logarithm and Digits –

Here're some base-10 logarithms:

log10(1000) = 3

log10(49323) ≈ 4.69

log10(100000) = 5

log10(333333) ≈ 5.52

**Which statement is true?**

1. **The base-10 log of a number, if rounded down, is the number of digits of the number.**
2. **The base-10 log of a number, if rounded up, is 1 more than the number of digits of the number.**
3. **The base-10 log of a number, if rounded down, is 1 less than the number of digits of the number.**
4. **None of the above.**
5. Logarithm Not Integer –

**The value of** **log2​(7) is** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

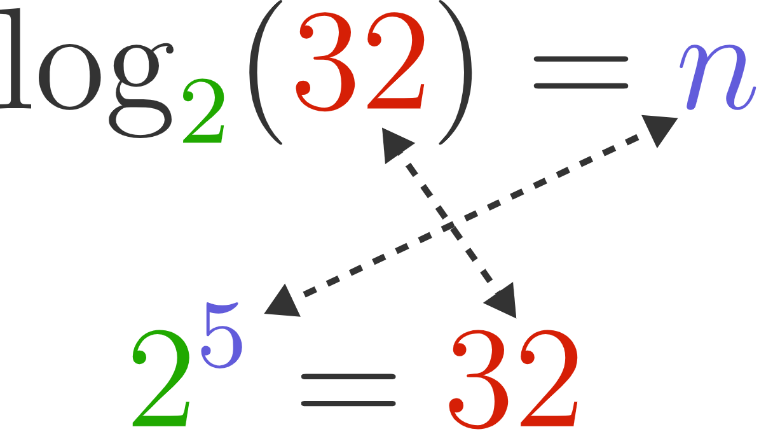
1. Between 1 & 2
2. Between 2 & 3
3. Between 3 & 4
4. Between 4 & 5
5. Logarithm 1 –

The **base** of an exponent is the number being multiplied. For example, in the power 35, 3 is the base.

Given a base and the result after raising it to a power, how do we find the power that was used?

Essentially, we would like to take the inverse of the **exponential** function.

This inverse function is a **logarithm** written as “log”:



Calculating the logarithm log2​(32) is the same as asking,

“2 raised to what power is 32?” => That is 5.

What is log6(36^2)?

1. 1
2. 2
3. 3
4. 4
5. 6

suppose, log6(36^2) = x

6 ^ x = 36 ^ 2

6 ^ x = (6 ^ 2) ^ 2

6 ^ x = 6 ^ 4

x = 4