# ARMSTRONG NUMBER OR NOT:

**PROBLEM STATEMENT:** Here we given an integer input. The task is to check if this number is an Armstrong number or not.

# ARMSTRONG NUMBER:

* A number of n digits is a number that is equal to the **sum of its own digits each raised to the power *n***.
* For **3-digit numbers**, the condition becomes,

abc=a3+b3+c3

* **num = int(input())-**-> input is taken from the user from keyboard.
* The variable num stores the number you want to check.

## sum = 0 , temp = num

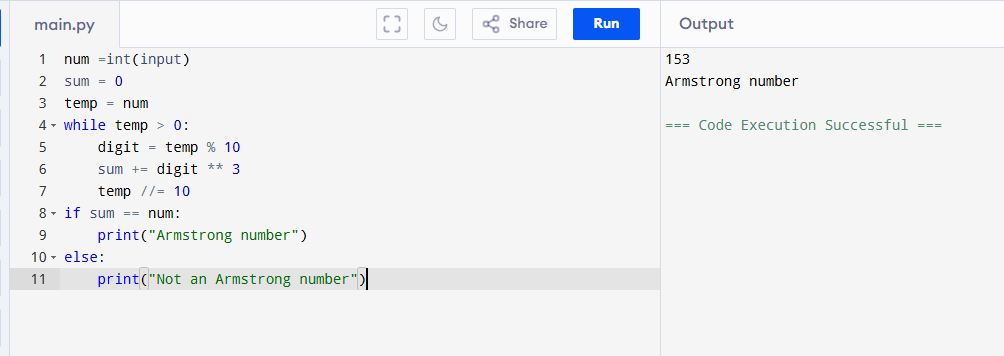
* sum is initialized to 0 — it will hold the **sum of the cubes** of the digits.
* Temp is a copy of num, we will use temp for calculations while keeping num unchanged for comparison later.
* **while temp > 0:**This loop continues as long as there are digits left in temp.
* digit = temp % 10:%gives the last digit of temp.
* **sum+=digit\*\*3**:digit\*\*3 is the **cube of the digit ,**we add this cube to the running total in sum
* **temp//=10:**// is integer division, It removes the last digit from temp.
* **if sum == num: print("Armstrong number")**

**else:**

**print("Not an Armstrong number")**

* After the loop, we compare the original number num with the sum of cubes of its digits. If they’re equal then it is Amstrong number if they’re not equal then it is not an Amstrong number.

# OUTPUT:



**REVERSED NUMBER:**

**PROBLEM STATEMENT:** Here we given an integer input. The task is to check if this number is an Reversed number or not.

**REVERSED NUMBER:** A **reversed number** means rearranging the digits of a number in the **opposite order**.

* + **n = int(input()**)--> input is taken from the user from keyboard.
  + **rev=0:**

initializes a variable rev(short for "reverse") to store the **reversed number.** This starts at 0 and will build up the reverse of the original number digit by digit.

## While n>0:

This loop runs as long as n is greater than 0.It processes the number one digit at a time from the rightmost digit.

## rem=n%10:

Extracts the **last digit** of n ,% is the modulus operator, which gives the remainder after division.

## rev=(rev\*10)+rem:

rev\*10 shifts the existing digits in rev one place to the left, +rem appends the new digit (the one we just extracted).

## Example:

Initially: rev=0

After first digit (4): rev=0\*10+4

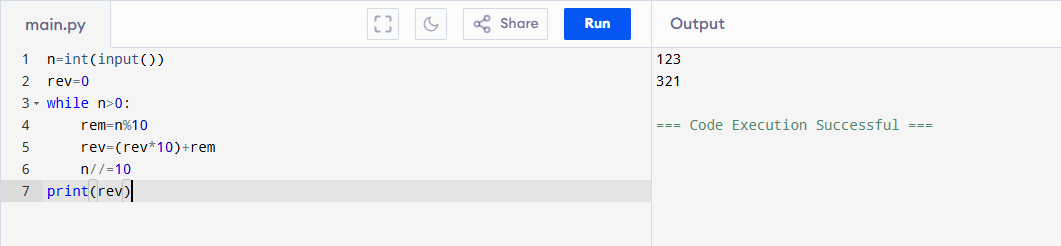
Next digit (3): rev=4\*10+3=43......

## n//=10:

Removes the **last digit** of n.// is integer division (no decimal part). Example: If n = 1234, then n = 1234 // 10 = 123

## Print(rev):

After the loop finishes (when n==0), this line prints the final reversed number.

**OUTPUT:**