Armstrong Number -

n = 153
digits = 3

$1^3 + 5^3 + 3^3$
$\Rightarrow 1 + 125 + 27 \Rightarrow 153$ ✓ Yes, true AN.

n = 100
digits = 3

$1^3 + 0^3 + 0^3$
$\Rightarrow 1$, false, Not AN

① no. of digits

② power

In-built function to find length of a number.

int n = String.valueOf(number).length();

n = 1234 → "1234" → 4

```
class Main {
    public static void main(String[] args) {
        int n = 1234567;
        int len = String.valueOf(n).length();
        System.out.println(len);
    }
}
```

Output

```
7

=== Code Exe
```

① power

② is Armstrong.

n = 153 → (3) digits (1)

int rem = n%10

rem = 3

rem = 5

rem = 1

(2)

int ans = 0

ans = (3)^3 = 3x3x3

ans = (3)^3 + (5)^3

ans = (3)^3 + (5)^3 + (1)^3

ans = (3)^3 + (5)^3 + (1)^3 = 153

if (n == ans) returns true;

else returns false.

(3)

n = n/10

n = 15 (>0)

n = 1 (>0)

n = 0 (>0) X

ans = ans + (rem x rem x rem)

```java
public static boolean isArmstrong(int number){
    int original = number;
    int result = 0;
    int n = String.valueOf(number).length();

    while(original != 0){
        int rem = original % 10;
        result += power(rem, n);
        original /= 10;
    }
    return result == number;
}
```

```java
public static int power(int base, int expo){
    int result = 1;
    for(int i = 1; i <= expo; i++){
        result *= base;
    }
    return result;
}
```

```java
Scanner s = new Scanner(System.in);
int x = s.nextInt();
int y = s.nextInt();

for(int i = x; i <= y; i++){
    if(isArmstrong(i)){
        System.out.println(i);
    }
}
}
```

i=300 false;

0 = 300 → n = 3

n = 0 300 != 0 T

rem = 300 % 10 → 0

n = 0 + 0 = 0

300 / 10 → 30

30 % 10 = 0

0 + 0 = 0

30 / 10 → 3

result = 1

i = 1 ≤ 3T 2≤3T 3≤3T

1 x 0 x 0 x 0 = 0

3 % 10 = 3

0 + 27

= 27

1 x 3 x 3 x 3

3 / 10 = 0

[27 == 300] → false

Find GCD.

```java
public static int gcd(int x, int y){
    while(y != 0){
        int temp = y;
        y = x % y;
        x = temp;
    }
    retun x;
}
```

Print all unique prime factors

```java
public static void main(String[] args) {
    /* Enter your code here. Read input from STI
}

public static boolean isprime(int n){

}
public static void factors(int n){
    for(int i = 2; i <= n;i++){
        if(n % i == 0 && isprime(i)){
            Syso(i);
            while(n % i == 0){
                n /= i;
            }
        }
    }
```

Print the array elements linewise

```java
Scanner s = new Scanner(System.in);
// size of array
int n = s.nextInt();

// array declaration
int arr[] = new int[n];

// take input in array
for(int i = 0; i < n; i++){
    arr[i] = s.nextInt();
}

// array output
for(int i = 0; i < n; i++){
    System.out.println(arr[i]);
}
}
```

Identical →

arr1 → | 1 | 2 | 3 | — 3 ✓

arr2 → | 1 | 2 | 3 | 4 | — 4 ✓

arr1 → | 1 | 2 | 3 | 4 | — 4

(0 1 2 3)

arr2 → | 1 | 2 | 3 | 5 | — 4

(0 1 2 4)

arr1[i] == arr2[i]

arr[0] == arr2[0]

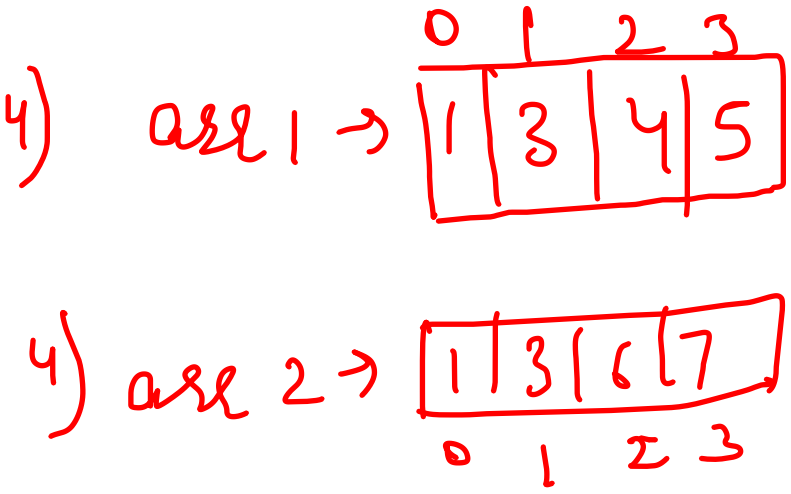
1 == 1 ✓, T

① length equal

② elements equal.

Check if two arrays are identical?

```
    Scanner s = new Scanner(System.in);
    int n = s.nextInt();
    int arr1[] = new int[n];
    for(int i = 0; i < n;i++){
        arr1[i] = s.nextInt();
    }

    int m = s.nextInt();
    int arr2[] = new int[m];
    for(int i = 0; i < m;i++){
        arr2[i] = s.nextInt();
    }

    boolean ans = arraysEqual(arr1, arr2);
    System.out.println(ans ? "true" : "false");

}

public static boolean arraysEqual(int arr1[], int arr2[]){
    if(arr1.length != arr2.length){
        return false;
    }

    // compare each elements
    for(int i = 0; i < arr1.length; i++){
        if(arr1[i] != arr2[i]){
            return false;
        }
    }
    return true;
}
```



i = 0 < 4

arr1[0] != arr2[0]

1 != 1

i = 1 < 4

arr1[1] != arr2[1]

3 != 3

i = 2 < 4

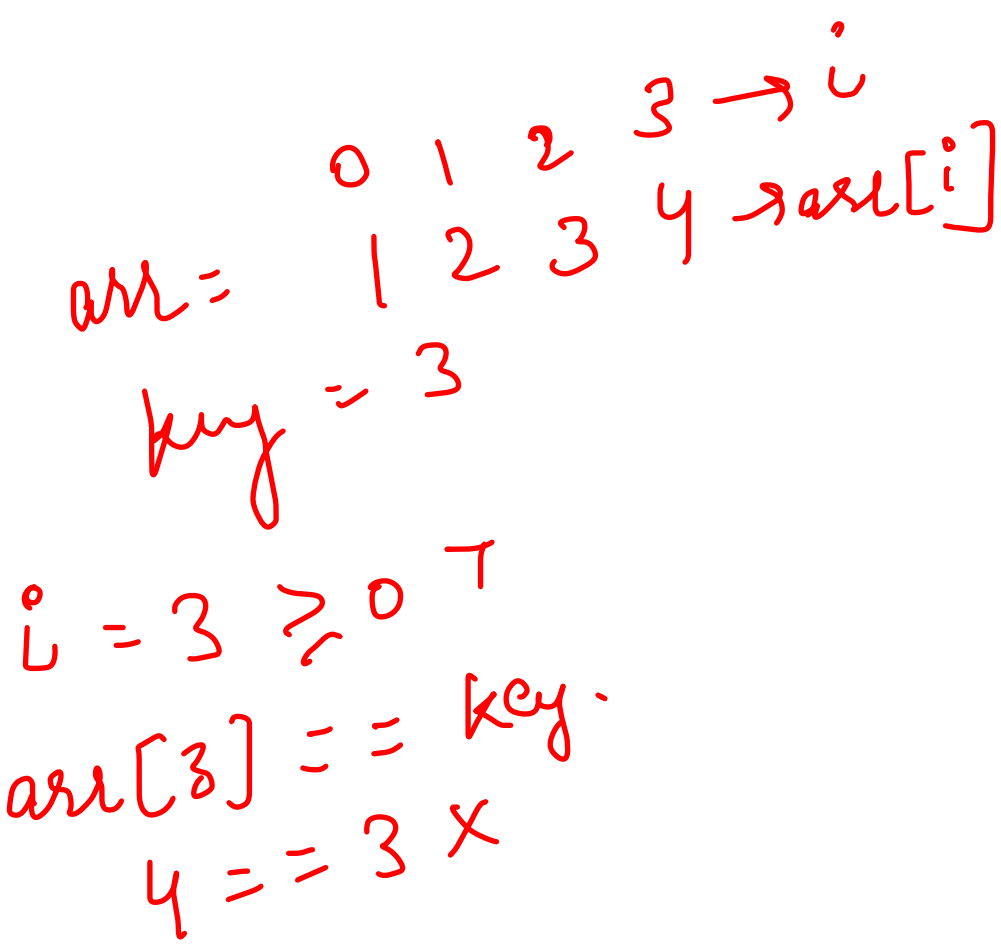
arr1[2] != arr2[2]

4 != 6 T — false

hw_Print last index of x in array

```
    // take input arr

   // take key i/p
}
public static int search(int arr[], int key){
    for(int i = arr.length - 1; i >= 0; i--){
        if(arr[i] == key){
            return i;
        }
    }
    return -1;
}
```

o/p → 2.



Print First NON MATCHING NUMBER

```
    // arr1
    // arr2

    int index = -1;
    for(int i = 0; < n; i++){
        if(arr1[i] != arr2[i]){
            index = i;
            break;
        }
    }
    Syso(index);
}
```