Question 1: Find the Duplicate Number

Given an array of integers `nums` containing `n + 1` integers where each integer is in the range `[1, n]`, return the repeated number. There is only **one** repeated number in the array.

Examples:

```
Input: nums = [1,3,4,2,2]
Output: 2

Input: nums = [3,1,3,4,2]
Output: 3

Input: nums = [3,3,3,3,3]
Output: 3
```

Time Complexity:

O(n).

Question 2: Butterfly Pattern

n=5

```
*         *
**       **
***     ***
****   ****
**********
****   ****
***     ***
**       **
*         *
```

n=3

```
*     *
**   **
*****
**   **
*     *
```

Question 3: Perfect Number

Description:

A **perfect number** is a positive integer that is equal to the sum of its **positive divisors**, excluding the number itself.

Given an integer `num`, return `true` if `num` is a perfect number, otherwise return `false`.

Examples:

```
Input: num = 28
Output: true
Explanation: 1 + 2 + 4 + 7 + 14 = 28

Input: num = 7
Output: false
```

Time Complexity:

O(sqrt(n))

Space Complexity:

O(1)

Question 4: Sqrt(x)

Description:

Given a non-negative integer `x`, return the square root of `x` **rounded down to the nearest integer**. You must not use any built-in exponent function or operator.

Examples:

```
Input: x = 4
Output: 2

Input: x = 8
Output: 2
Explanation: sqrt(8) = 2.828..., round down to 2
```

Time Complexity:

O(log x).

Space Complexity:

O(1)

Question 5: Prime Number

Description:

Check whether given number is prime or not !?

Examples:

```
Input: x = 7
Output: true

Input: x = 10
Output: false


Input: x = 29
Output:true
```

Time Complexity:

Less than O(n)

Space Complexity:

O(1)

Question 6: Factorial Trailing Zeroes:

Given an integer n, return the number of trailing zeroes in n!. Note that n! = n * (n - 1) * (n - 2) * ... * 3 * 2 * 1.

Examples:

```
Input: n = 3
Output: 0 cause 3!=6

Input: n = 5
Output: 1 cause 5! = 120, one trailing zero


Input: n = 0
Output: 0
```

Time Complexity:

Less than O(n)

Space Complexity:

O(1)

Question 7: Next Greater Element

Given an array arr[] of integers, the task is to find the next greater element for each element of the array in order of their appearance in the array.

Examples:

```
Input: arr[] = [1, 3, 2, 4]

Output:[3, 4, 4, -1]
```

```
Input: arr[] = [6, 8, 0, 1, 3]

Output:[8,-1,1,3,-1]
```