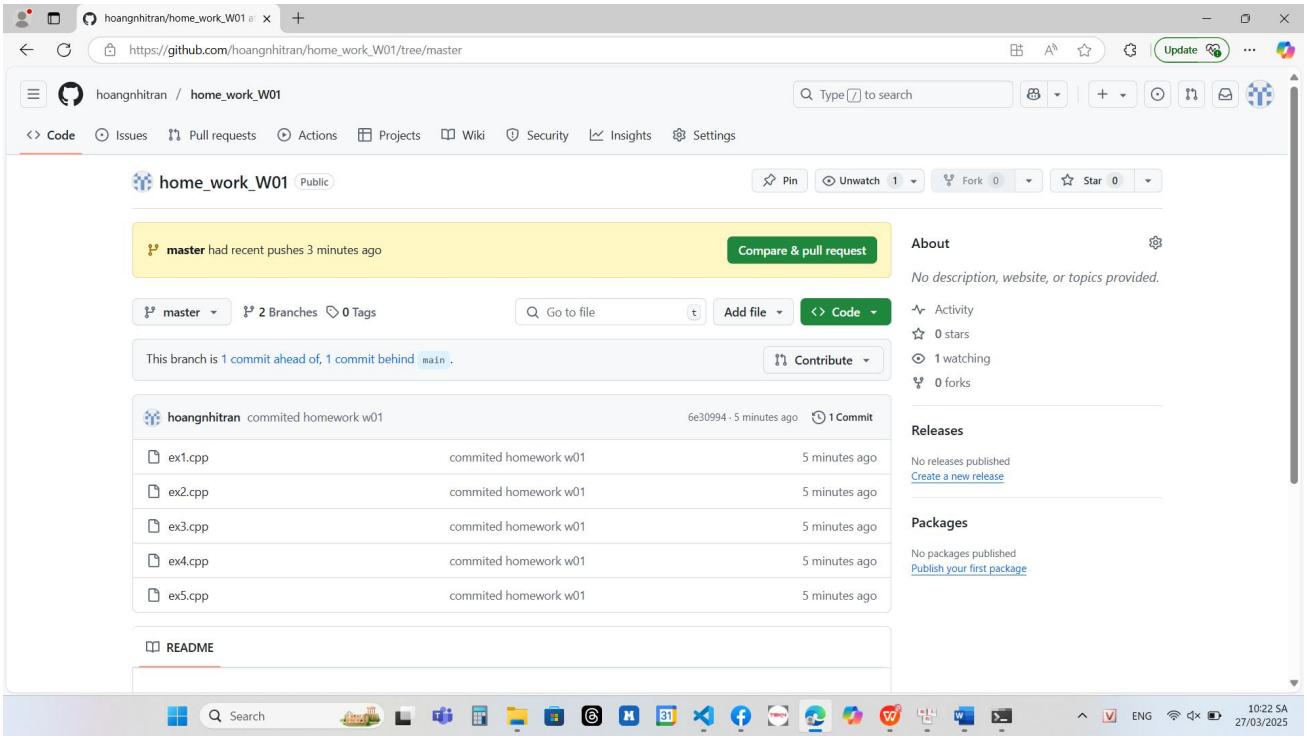
Weekly Homework 1 Report

1 Git & GitHub



2 Exercises

Exercise 1: Fibonacci Series

Problem Statement

Write a recursive function to compute the **nth Fibonacci number**. The Fibonacci series is defined as:

$$F(n) = F(n-1) + F(n-2)$$

where:

$$F(0) = 0, \qquad F(1) = 1$$

```
int fibonacci(int n);
```

C++ Function Signature

1

Example Input & Output

Input: 5 Output: 0 1 1 2 3

Algorithm Explanation

- The function fibonacci(n) is called recursively to compute the Fibonacci number at position n.
- If n is 0 or 1, it directly returns n (base case).
- If n is greater than 1, the function returns the sum of the previous two Fibonacci numbers (fibonacci(n - 1) + fibonacci(n - 2)).
- The main() function takes an integer input n and prints the Fibonacci series from 0 to n - 1.

Exercise 2: Factorial of a Number

Problem Statement

Write a recursive function to compute the **factorial** of a given number n. Factorial is defined as:

$$n! = n \times (n-1) \times (n-2) \times \cdots \times 1$$

where:

$$0! = 1$$

C++ Function Signature

```
int factorial(int n);
```

1

Example Input & Output

Input: 5 Output: 120

Factorial Algorithm

- The function factorial(n) is called recursively to compute the factorial of number n.
- If n is 0, it returns 1 (base case).

- Otherwise, it returns n * factorial(n - 1).
- The main() function reads the integer input and prints the factorial of that number.

Exercise 3: Generate All Binary Strings

Problem Statement

Write a recursive function to generate **all binary strings** of length *n*. A binary string consists only of '0's and '1's.

```
void generateBinaryStrings(int n, string str);
```

C++ Function Signature

1

Example Input & Output

Input: 3 Output:
000 001 010 011
100 101 110 111

Binary String Generation Algorithm

- The function kBits(k, s, n) is called recursively to generate all binary strings of length k.
- The base case occurs when the string reaches length k.
- Otherwise, it recursively sets the next character to 0 or 1 and continues.

Exercise 4: Towers of Hanoi puzzle

Towers of Hanoi Algorithm

- The function towerOfHanoi(n, fromRod, auxRod, toRod) recursively moves n disks.
- The base case occurs when n == 0.
- Otherwise:

1. Move n - 1 disks from source to auxiliary rod.
2. Move the nth disk from source to destination.
3. Move n - 1 disks from auxiliary to destination rod.

Exercise 5: Given an array , check whether the array is in sorted order with recursion.

Sorted Array Check Algorithm

- The function isAscending(arr, n) checks if the array is sorted in ascending order.
    - If the array has 0 or 1 elements, it is sorted (base case).
    - If the first element is greater than the second, return false.
    - Otherwise, recursively check the rest of the array.
- The function isDescending(arr, n) follows a similar method.
- The main function calls both isAscending(arr, n) and isDescending(arr, n)to check if the array is sorted.