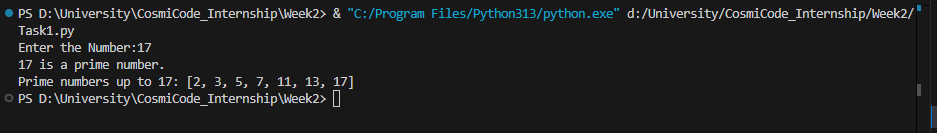
# **Week 2**

## **Task 1**

Output:



Research:

How It Works:

1. **is\_prime(n)**:
   * Checks if a number is prime by testing divisibility up to the square root of the number
   * Returns True if prime, False otherwise
2. **list\_primes\_up\_to(n)**:
   * Generates all prime numbers up to n using the is\_prime() function
   * Returns a list of primes
3. **main()**:
   * Gets user input and validates it
   * Checks if the number is prime
   * Lists all primes up to that number
   * Handles invalid input gracefully

## **Task 2**

Output:

A computer screen shot of a program

AI-generated content may be incorrect.

Research:

Approaches

1. **Iterative Approach**: Uses a loop to calculate Fibonacci numbers efficiently
2. **Recursive Approach**: Uses function calls to calculate Fibonacci numbers (less efficient but more mathematically elegant)

Iterative Approach (fibonacci\_iterative)

1. **Initialization**: Starts with the first two Fibonacci numbers [0, 1]
2. **Loop**: For each subsequent number up to n-1:
   * Calculates the next number as the sum of the previous two
   * Appends it to the sequence
3. **Return**: Returns the first n numbers

Recursive Approach (fibonacci\_recursive)

1. **Base Cases**: Directly returns 0 or 1 for n=0 or n=1
2. **Memoization**: Stores computed values to avoid redundant calculations
3. **Recursion**: For other values, recursively calculates as:
   * fib(n) = fib(n-1) + fib(n-2)
4. **Sequence Generation**: get\_recursive\_sequence creates the full sequence by calling the recursive function for each index

Main Function

1. Sets n = 30
2. Generates and prints sequences using both methods
3. Verifies both methods produce identical results

## **Task 3**

Output:

A black screen with blue text

AI-generated content may be incorrect.

Research:

* **GCD** is calculated using the **Euclidean algorithm**.
* **LCM** is calculated using the formula:

A mathematical equation with black text

AI-generated content may be incorrect.

## **Task 4**

Output:

A black screen with blue text

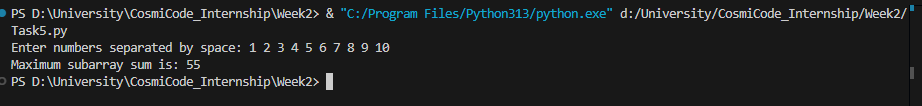
AI-generated content may be incorrect.

Research:

**Prime numbers** that multiply together to give the original number.

## **Task 5**

Output:



Research:

**Goal:**  
Find the **maximum sum of any subarray** from a list of numbers.

**Idea:**

* Go through each number one by one.
* Keep track of:
  + The **current sum** of the subarray we are building.
  + The **maximum sum** we’ve found so far.
* If adding the current number makes the sum worse, **start a new subarray** from that number.
* Keep updating the maximum whenever we find a bigger one.