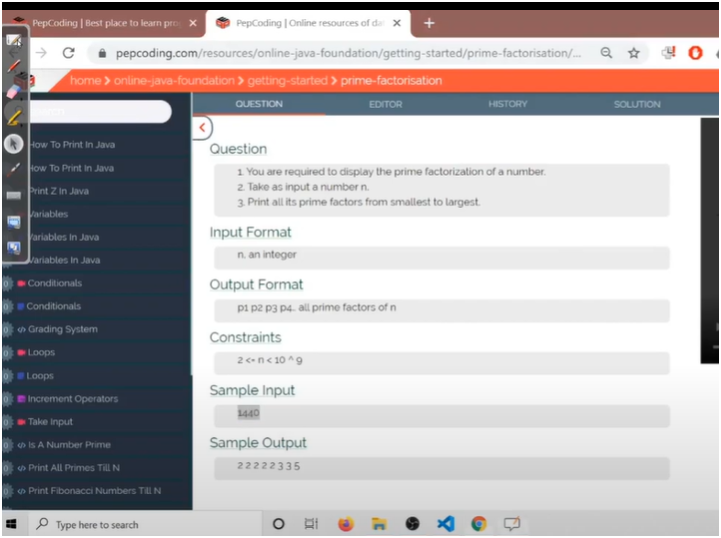
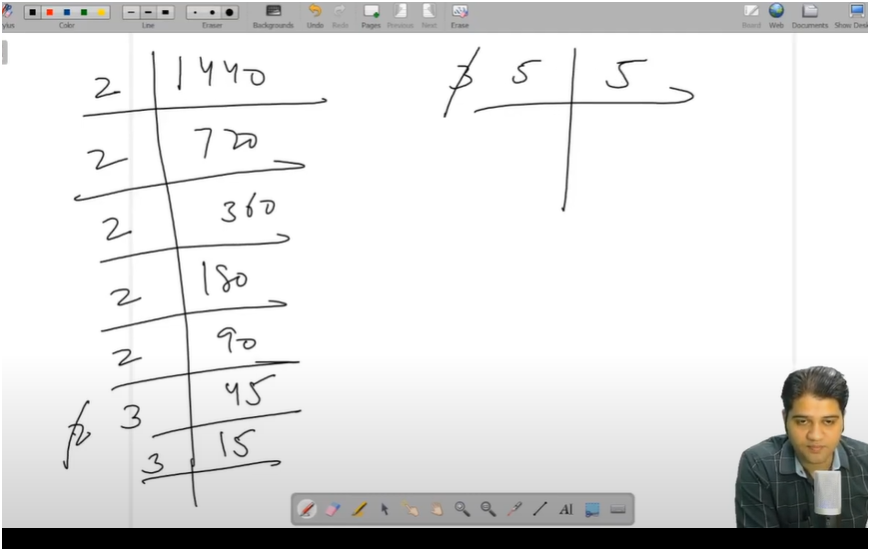
PRIME FACTORIZATION

DIV|QUOTIENT





HERE WE DIVIDE THE INPUT NUMBER MULTIPLE TIMES UNTIL THE QUOTIENT GET EQL TO 1

AND THE NUMBER THAT DIVIDE THE INPUT NUMBER START FROM 2 IF THE NUMBER DOESN’T GETS DIVIDE INCREMENT THE NUMBER BY 1 SUH AS 3,4,5… < ROOT OF INPUT NUMBER

WE INCREASE THE DIV ONLY WHEN THE NUMBER DOESNOT GET FULLY DIVIDE BY DIV

IF THE NUMBER DOESN’T GET DIVIDE TILL ROOT OF INPUT NUMBER THEN THE NUMBER ITSELF A FACTOR

**ALGO:-**

**\* 📌 SUMMARY: PRIME FACTORIZATION**

**\* This program prints the prime factorization of a given number.**

**\***

**\* ✅ HOW IT WORKS:**

**\* 1. Start dividing the input number `n` from 2 (smallest prime).**

**\* 2. If `n` is divisible by the current divisor `div`, print it and divide `n` by `div` AGAIN.**

**\* 3. Keep dividing by the same `div` until it no longer divides `n`.**

**\* 4. Then increase `div` to check for the next possible prime factor.**

**\* 5. Loop continues while div\*div <= n (we don’t need to check beyond √n).**

**\* 6. If after the loop, `n` is still > 1, it means `n` itself is a prime number and should be printed (this handles cases like 13, 17, 29, etc).**

**\* 🧠 Example:**

**Input: 60**

**Output: 2 2 3 5**

**(Because 60 = 2 × 2 × 3 × 5)**