**NETWORK SECURITY ASSIGNMENT**

IMPLEMENTATION OF FIAT-SHAMIR

BINDU IIT2018105

AYUSHI GUPTA IIT2018118

SANJANA IIT2018120

TS CHINMAY IIT2018138

NILANG VASAVA IIT2018147

*V Semester B.Tech, Department of Studies in Computer Science,*

*Indian institute of information technology Allahabad,Allahabad, India*

**What is fiat shamir protocol?**

It is One of the most well-known protocols identification using zero knowledge proof protocol and is proposed by Amos Fiat and Adi Shamir. It uses the zero-knowledge interactive proofs for entity authentication.

**Zero knowledge identification proof**

A Zero knowledge proof or Zero knowledge Protocol is a method where one party can prove (prover) to another party (verifier) without revealing knowledge of the secret itself.

Zero Knowledge proofs can be used for identification. Identification schemes in general, then “traditional” secret-key and public-key schemes and finally zero-knowledge schemes. Identification schemes are methods by which a user may prove his or her identity without revealing knowledge that may be used by an eavesdropper to impersonate the user.

A Zero Knowledge protocol must satisfy the three conditions:

* **Completeness:** If a statement is true and both parties follow the same protocol correctly, then the verifier naturally becomes convinced
* **Soundness:** If the statement is false, the verifier will almost certainly not be convinced (Probabilistically Checkable Proof constructions rely on repetition until probability of falsehood or plain coin flip luck approaches zero).
* **Zero-knowledge:** If the statement is true, no verifier learns anything other than the fact that the statement is true.
* Alice wants to prove she knows a secret without revealing any info about it.
* Bob must verify that Alice knows the secret.
* But Bob gains no info about the secret.
* Process is probabilistic .
* Bob can verify that Alice knows the secret to an arbitrarily high probability.
* It is an “interactive Proof System”.

**Fiat shamir explained further:**

There are several variants of the Fiat–Shamir identification protocol:

1. To classify these is based on the number of secrets.

2. To distinguish between identity-based and public-key based one.

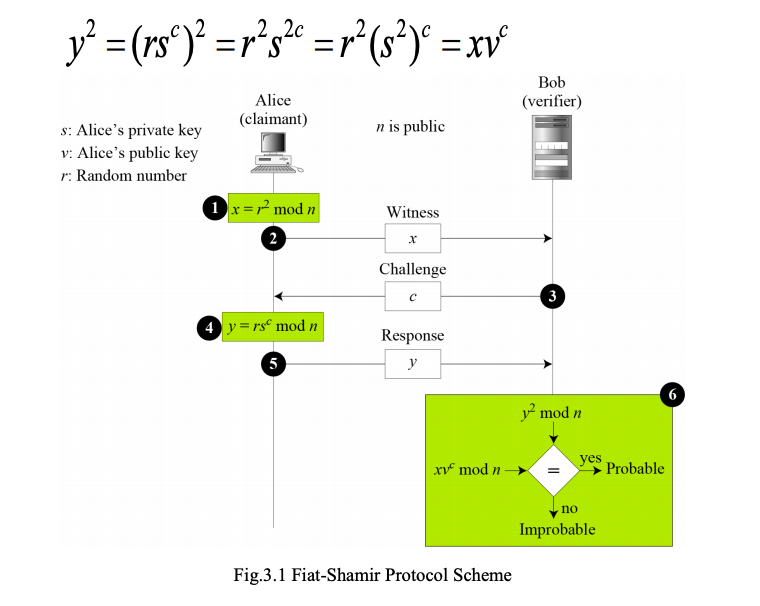
In both, a trusted center made public n = p ⋅ q such that p and q are secret primes only known to the centre.

**The Protocol:**

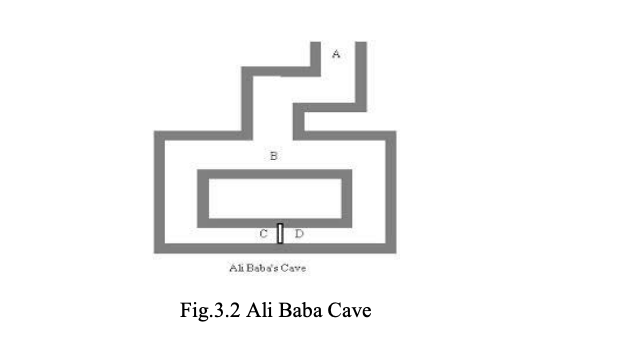
Repeat the protocol t times:

1. Alice chooses uniformly random r and computes z := r mod n and sends t to Bob.
2. Bob sends Alice bits ei ( ≤ i ≤ k).
3. Alice sends α :=r ∗ ∏ei=1 siei mod n.
4. Bob verifies that α2 =∏ei=1 xi=z

If all the verifications are correct, Bob accepts.



**Cave Example :**



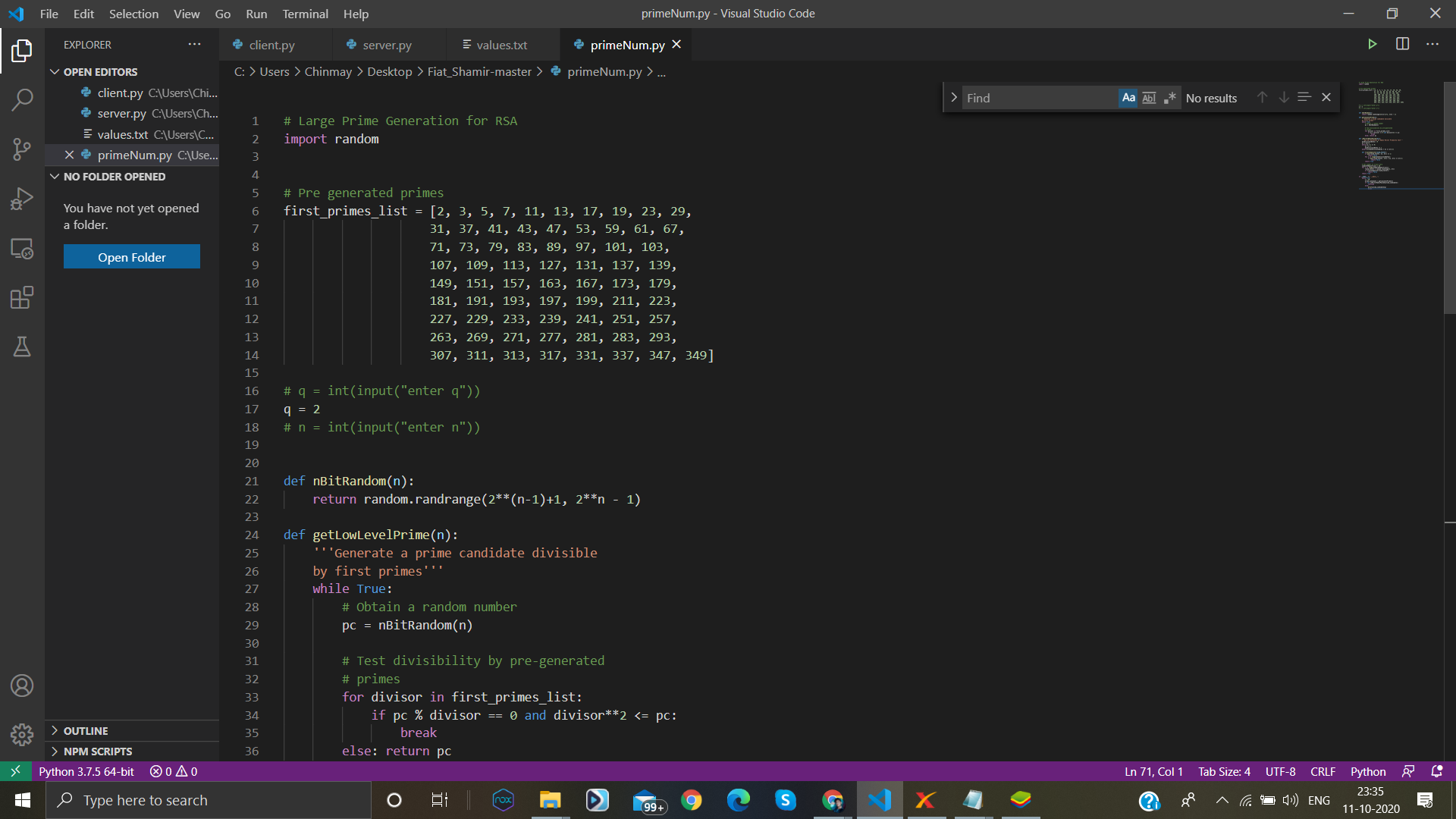
Suppose there is an underground cave like the above cave with a door at both ends which will open with some magic word. Alice says that she knows the word and thus can open the door. Suppose both Alice and Bob are standing at the entrance point A, then Alice goes to B, where Bob can’t see Alice.

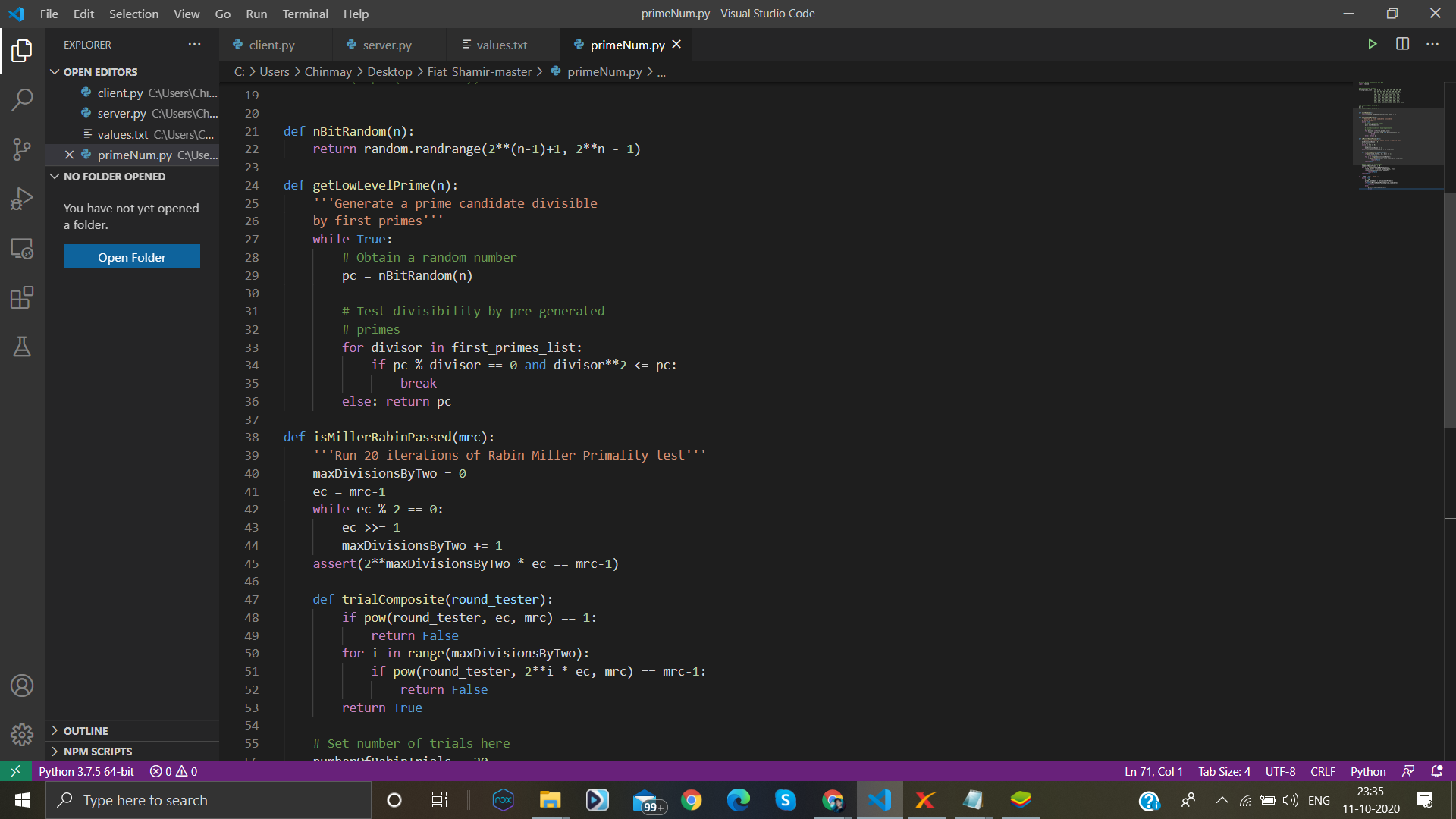
* Alice can move to either right or left and it can be corresponded to the witness.
* Then Bob comes to B and asks Alice to come from left or right and thus it corresponds to challenge.
* If Alice knows the word, she can come from any side which is requested. Because if she is on the right side, she can come easily and if on the wrong side, she can use the magic word to come from the requested side.
* This game can be repeated many times, Alice can win the game if she can pass the test all the time. The probability of her winning the game is very less if she doesn't have the private key (word).

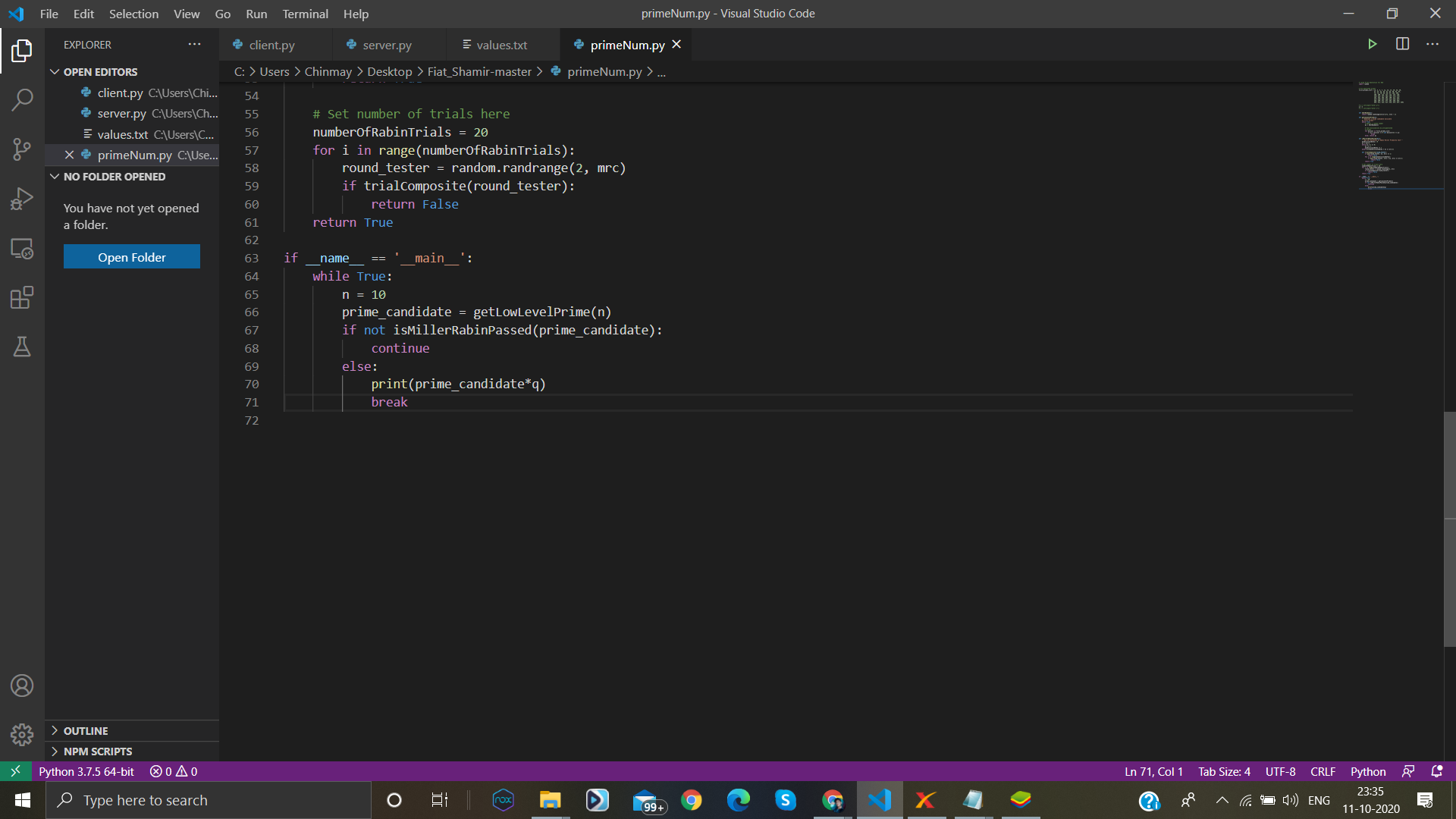
So if the game is run N times and as each time Alice has ½ chance of winnig.so the chance of Alice winning the game is (1/2)N .

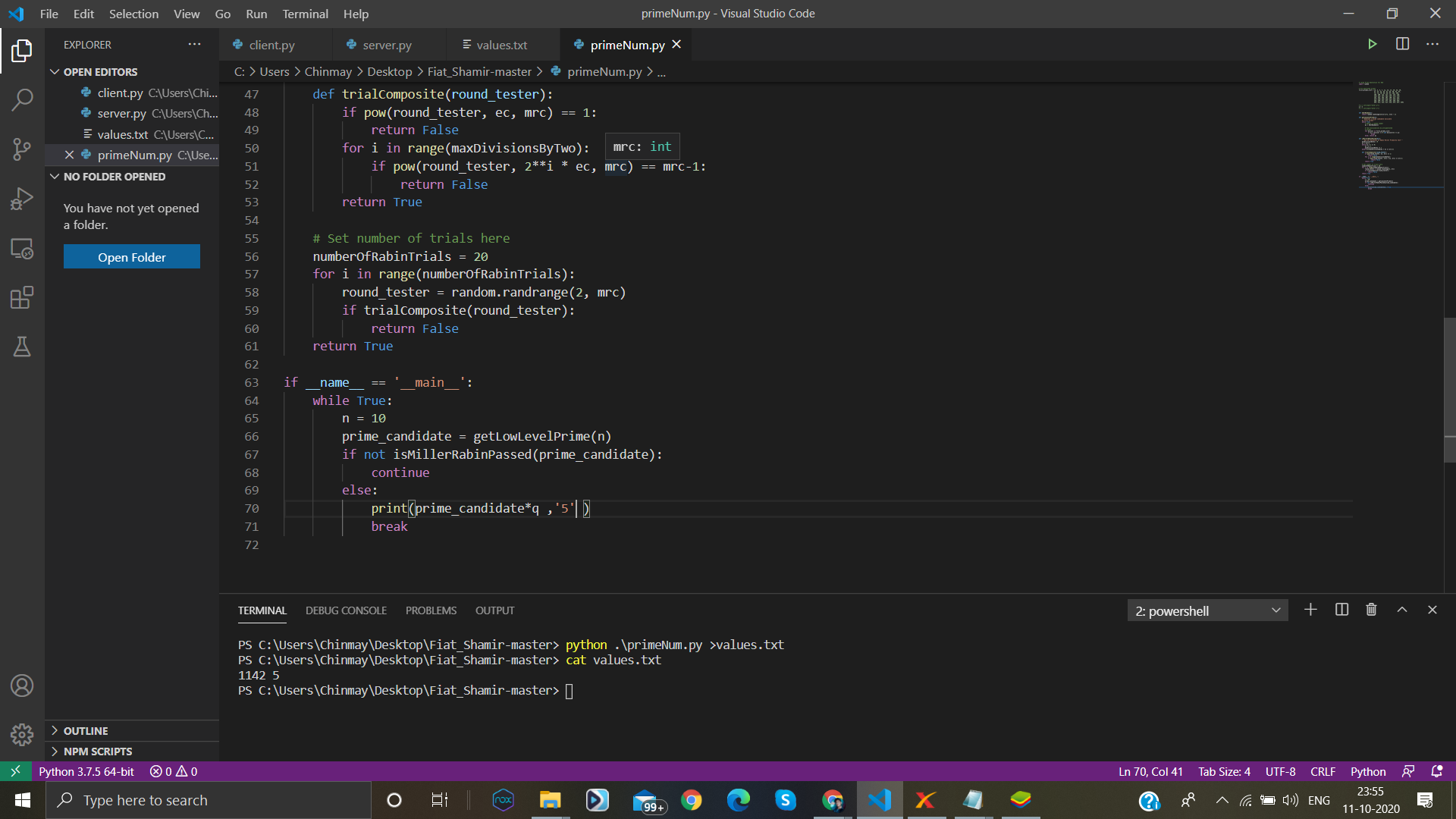
**IMPLEMENTATION :**

**CODE TO GENERATE LARGE PRIME VALUES FOR N**







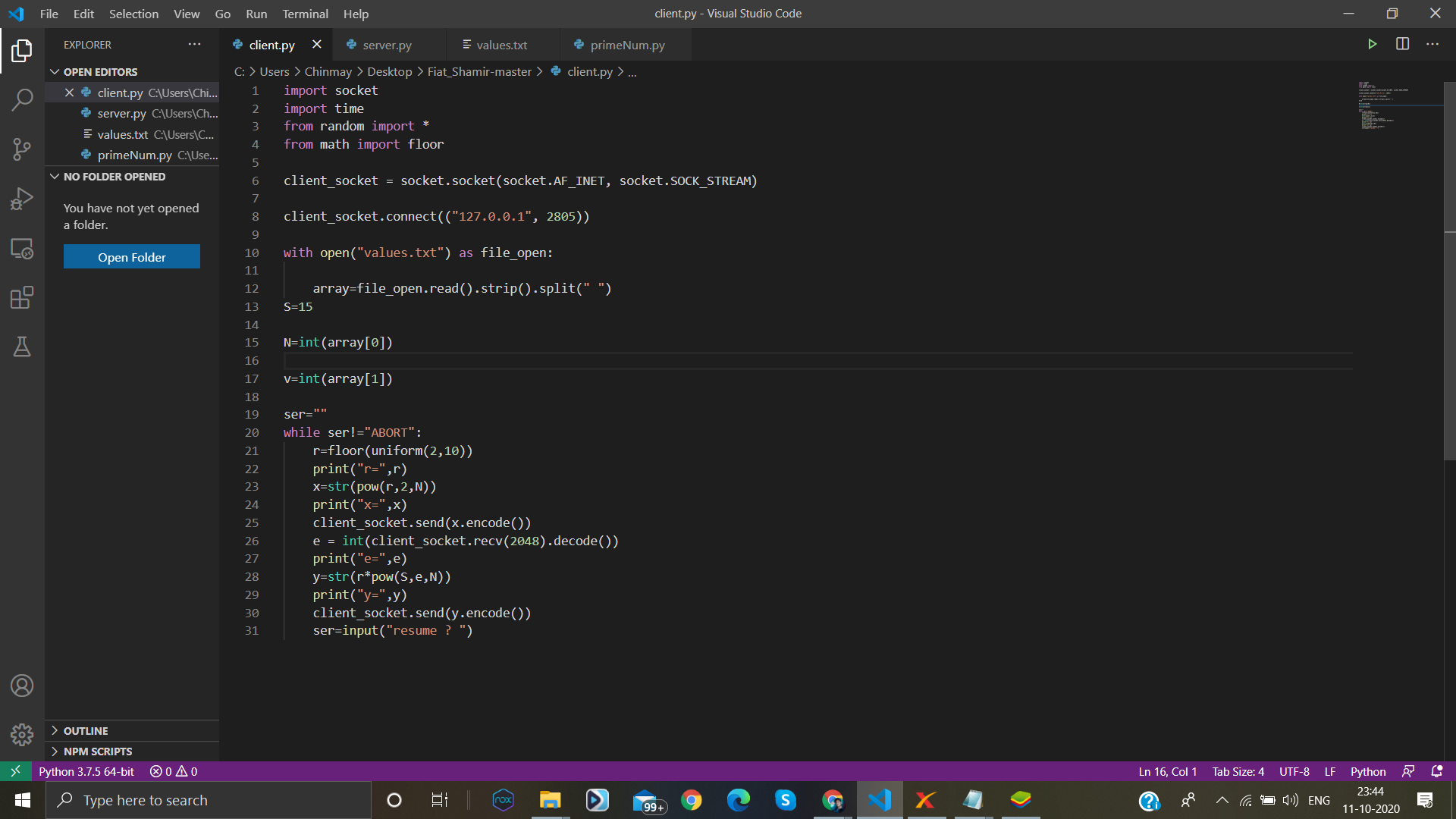


WHERE N = p\*q ( p will be randomly generated large prime

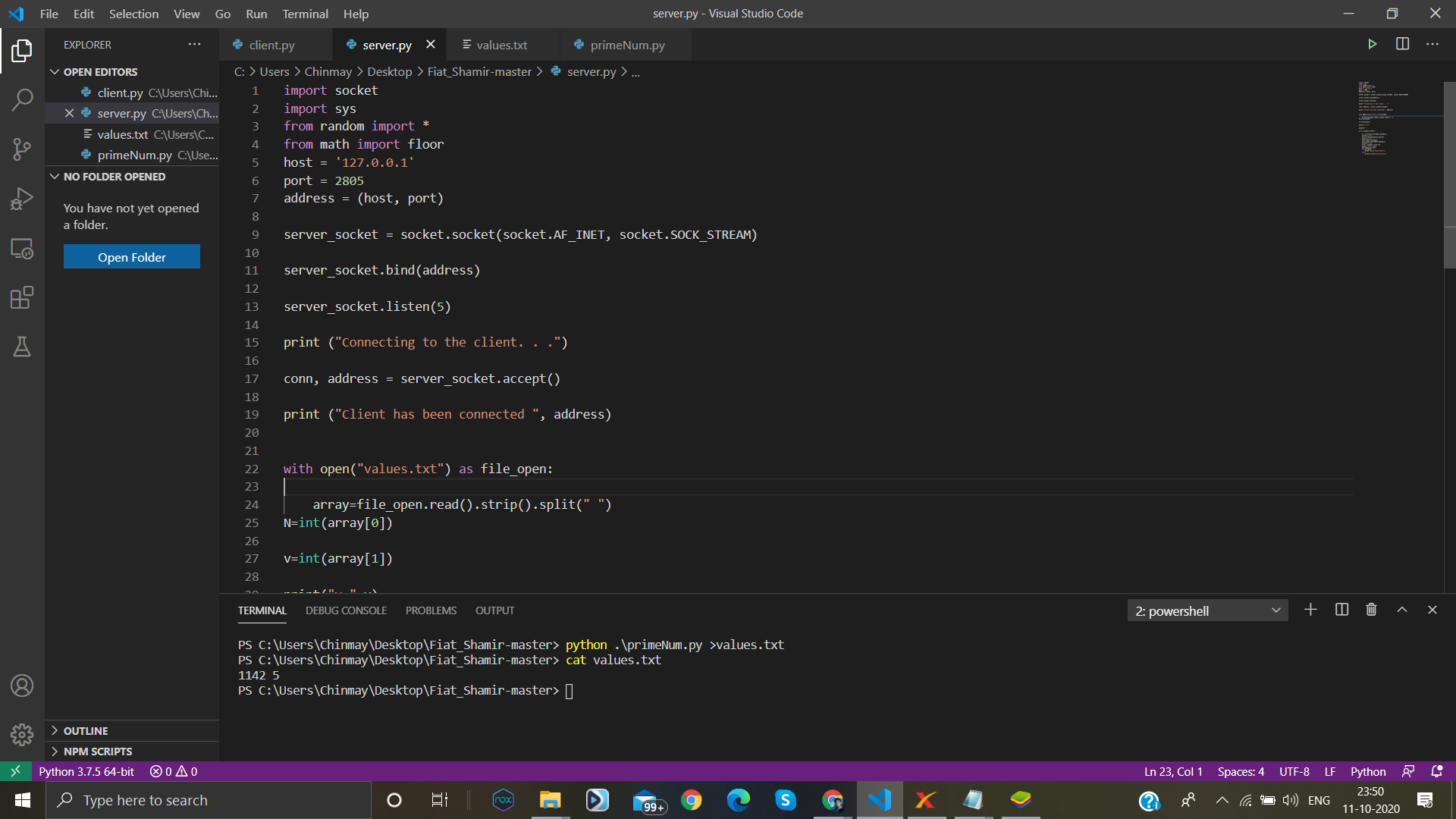
Where as q would be any small prime ) eg .2

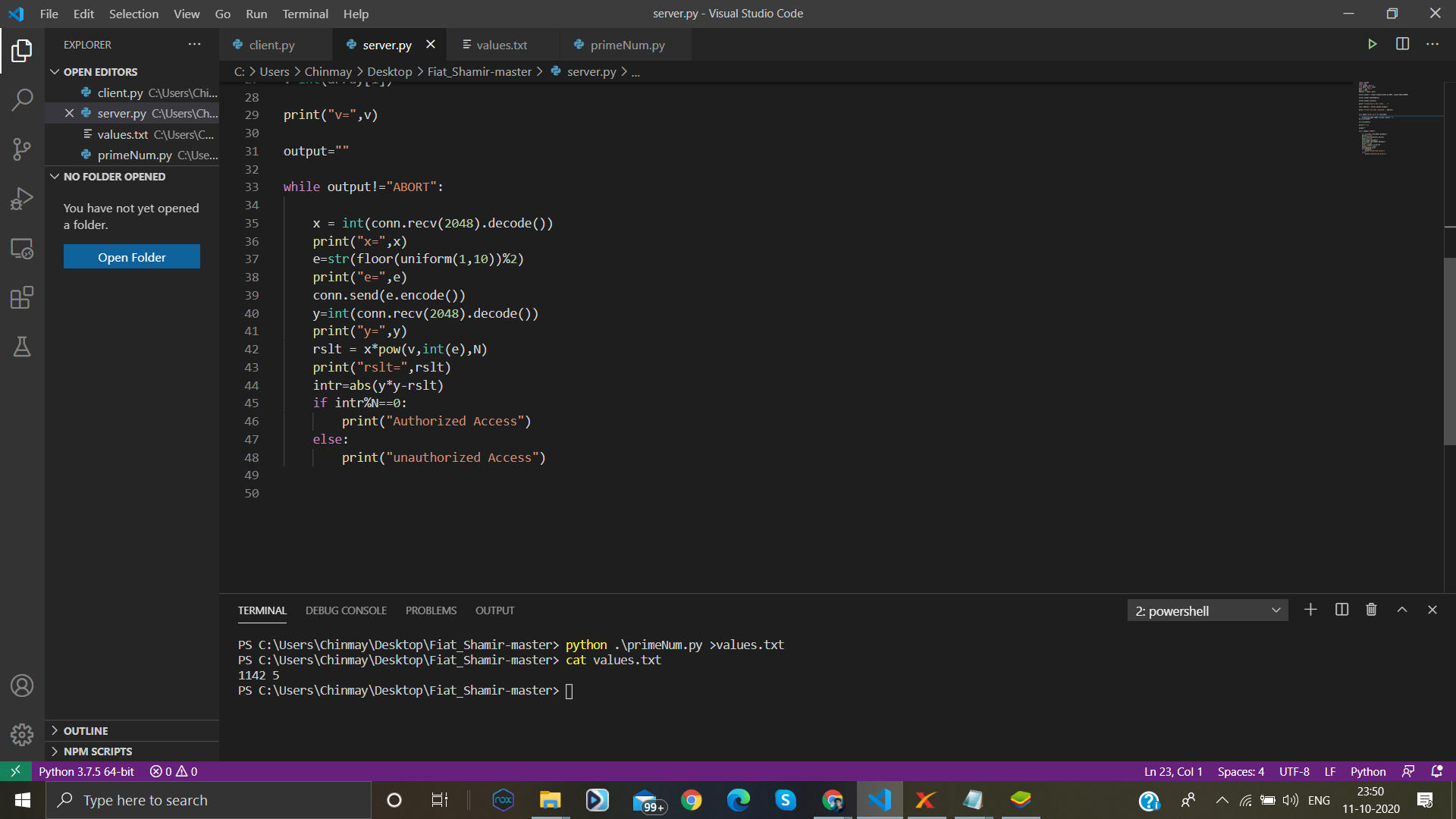
The client will take the values of N from the file values.txt.

**client.py**

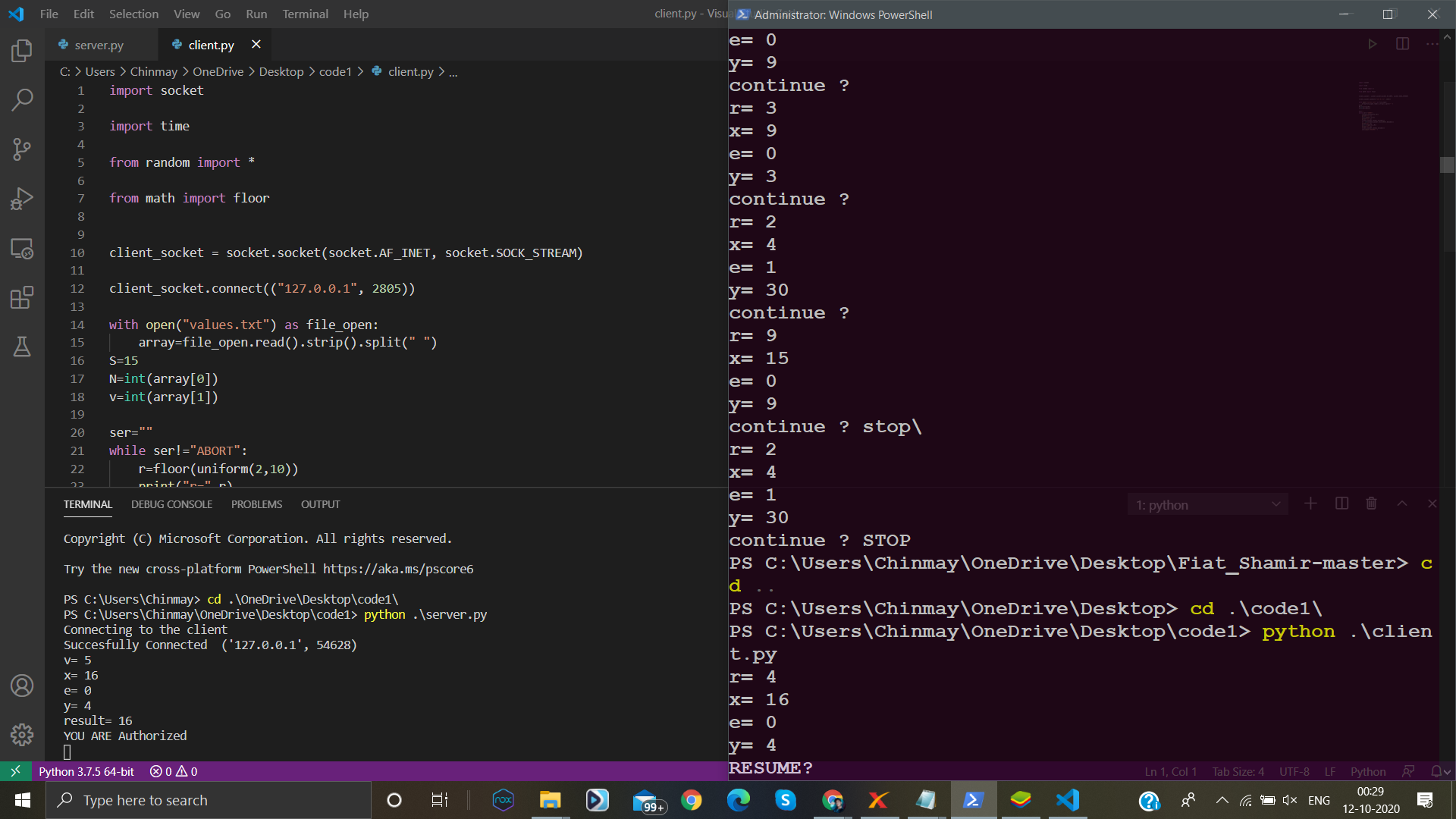


**server.py**





**OUTPUT-**

****