```java
/* To implement Fiat-Shamir protocol for entity authentication
   using a client server program where the client is the
   claimant and the server is the verifier. */

// VERIFIER SIDE

import java.net.*;
import java.io.*;
import java.util.*;

public class Verifier
{
    // initialise socket and input stream
    private Socket       socket = null;
    private ServerSocket server = null;
    private DataInputStream inFromClaimant   = null;
    private DataInputStream input = null;
    private DataOutputStream out = null;

    static int x; // witness from claimant
    static int c; // challenge to witness
    static int y; // response from claimant to the challenge
    static int n; // public key
    static int v; // public key of the claimant
    public Verifier(int port)
    {
        // starts server and waits for a connection
        try
        {
            server = new ServerSocket(port);

            System.out.println("Waiting for a Claimant");

            socket = server.accept();
            System.out.print("Claimant accepted.");
            System.out.print("\n");

            // takes input from the client socket
            inFromClaimant = new DataInputStream(
                new BufferedInputStream(socket.getInputStream()));
```

```
// takes input from terminal
input = new DataInputStream(System.in);

// sends output to the socket
out = new DataOutputStream(socket.getOutputStream());

String line = "";
// reads message from client until "Over" is sent
while (!line.equals("Over"))
{
    try
    {
        // Announce Public Key of Verifier
        out.writeUTF(Integer.toString(n));
        System.out.print("\nVerifier:: Public key -> "+n);

        // registration of public key of the claimant
        System.out.print("\nClaimant:: ");
        line = inFromClaimant.readUTF();
        v =  Integer.parseInt(line);
        System.out.print(v+" (public key registration)");

        // input of witness from claimant
        System.out.print("\nClaimant:: ");
        line = inFromClaimant.readUTF();
        x =  Integer.parseInt(line);
        System.out.print(x+" (commitment)");

        // sending challenge to claimant
        System.out.print("\nVerifier:: ");
        Random challenge = new Random();
        c = challenge.nextInt(2);
        out.writeUTF(Integer.toString(c));
        System.out.print(c+" (challenge)");


        // input of witness from claimant
        System.out.print("\nClaimant:: ");
        line = inFromClaimant.readUTF();
```

```
                y = Integer.parseInt(line);
                System.out.print(y+" (response to the challenge)");

                // calculating y^2 and x* (v^c)
                int ySquare = 1;
                for (int i = 1; i <= 2; i++)
                {
                    ySquare = ySquare * y;
                }
                int vPowerC = 1;
                for (int i = 1; i <= c; i++)
                {
                    vPowerC = vPowerC * v;
                }
                // verifying claimant
                if(ySquare % n == (x * vPowerC) % n)
                {
                    System.out.print("\nVerifier:: ");
                    out.writeUTF("Autheticated");
                    System.out.print("Claimant Verified.");
                }
                break;
            }
            catch(IOException i)
            {
                System.out.println(i);
            }
        }
        System.out.println("\nConnection terminated.\n");
        // close connection
        socket.close();
        inFromClaimant.close();
    }
    catch(IOException i)
    {
        System.out.println(i);
    }
}

static int checkPrime(int n)
```

```
    {
        // checking prime or not
        Scanner scan2 = new Scanner(System.in);

        boolean prime = false;
        while(prime == false)
        {

            if((n%6==1 || n%6==5 || n==2 || n==3) && n!=1)
            {
                prime = true;
            }
            else
            {
                System.out.print(n+" is not prime. Enter again:: ");
                n = scan2.nextInt();
            }
        }
        return n;
    }
    static int PublicKey()
    {
        Scanner scan = new Scanner(System.in);
        // input prime numbers and store in p and q
        System.out.print("\nEnter two prime numbers p and q.");
        System.out.print("\np = ");
        int p = scan.nextInt();
        p = checkPrime(p);
        System.out.print("q = ");
        int q = scan.nextInt();
        q = checkPrime(q);
        return p*q;
    }

    public static void main(String args[])
    {
        n = PublicKey();
        Verifier verifier = new Verifier(8221);
    }
}
```