```
public class CRC16CCITT {

  public static void main(String[] args) throws Exception {

    int crc = 0xFFFF;
    int polynomial = 0x1021;

    if (args.length == 0) {
      System.out.println("Usage: java CRC16CCITT <string>");
      return;
    }

    byte[] bytes = args[0].getBytes("ASCII");

    for (byte b : bytes) {
      for (int i = 0; i < 8; i++) {

        boolean bit = ((b >> (7 - i)) & 1) == 1;
        boolean c15 = ((crc >> 15) & 1) == 1;

        crc <<= 1;

        if (c15 ^ bit)
          crc ^= polynomial;
      }
    }

    crc &= 0xFFFF;

    System.out.println("CRC16-CCITT = " + Integer.toHexString(crc));
  }
}
import java.net.*;
import java.io.*;
```

```
import java.util.*;

public class StopWaitSender {
   public static void main(String[] args) throws Exception {

      Scanner sc = new Scanner(System.in);   // using sc
      System.out.print("Enter number of frames: ");
      int n = sc.nextInt();

      Socket socket = new Socket("localhost", 9999);

      BufferedReader in = new BufferedReader(
            new InputStreamReader(socket.getInputStream()));
      PrintStream out = new PrintStream(socket.getOutputStream());

      for (int i = 0; i < n; i++) {
         System.out.println("Frame " + i + " sent");
         out.println(i);

         String ack = in.readLine();
         if (ack != null)
            System.out.println("Acknowledgement received");

         Thread.sleep(1000);
      }

      out.println("exit");
      socket.close();
   }
}

import java.net.*;
import java.io.*;
```

```
public class StopWaitReceiver {
   public static void main(String[] args) throws Exception {

      ServerSocket server = new ServerSocket(9999);
      Socket socket = server.accept();

      BufferedReader in = new BufferedReader(
           new InputStreamReader(socket.getInputStream()));
      PrintStream out = new PrintStream(socket.getOutputStream());

      String frame;
      while (!(frame = in.readLine()).equals("exit")) {
        System.out.println("Frame " + frame + " received");
        out.println("ACK");
      }

      System.out.println("All frames received successfully");
      socket.close();
      server.close();
   }
}


package src.bin;

import java.util.Scanner;

public class Rsa {
   public static void main(String[] args) {

      Scanner in = new Scanner(System.in);   // ← changed sc → in

      int p, q, msg, n, z, e, d = 0, i, cipher = 1, plain = 1;

      System.out.println("Enter p and q:");
```

```
p = in.nextInt();
q = in.nextInt();

boolean pPrime = true, qPrime = true;

for (int k = 2; k <= p / 2; k++)
  if (p % k == 0) pPrime = false;

for (int k = 2; k <= q / 2; k++)
  if (q % k == 0) qPrime = false;

if (pPrime && qPrime) {

  System.out.println("Enter message:");
  msg = in.nextInt();

  n = p * q;
  z = (p - 1) * (q - 1);

  // choose e such that gcd(e,z)=1
  while (true) {
    System.out.println("Choose e such that gcd(z,e)=1:");
    e = in.nextInt();

    int a = e, b = z;
    while (b != 0) {
      int t = b;
      b = a % b;
      a = t;
    }

    if (a == 1) break;
    System.out.println("Invalid e. gcd != 1. Try again.");
  }
```

```
        // find d
        i = 2;
        while (true) {
          if ((i * e) % z == 1) {
            d = i;
            break;
          }
          i++;
        }

        // encryption
        cipher = 1;
        for (i = 1; i <= e; i++)
          cipher = (cipher * msg) % n;

        // decryption
        plain = 1;
        for (i = 1; i <= d; i++)
          plain = (plain * cipher) % n;

        System.out.println("Public Key: (" + e + ", " + n + ")");
        System.out.println("Private Key: (" + d + ", " + n + ")");
        System.out.println("Cipher Text: " + cipher);
        System.out.println("Plain Text: " + plain);

      } else {
        System.out.println("p or q is not prime");
      }
    }
}
```