

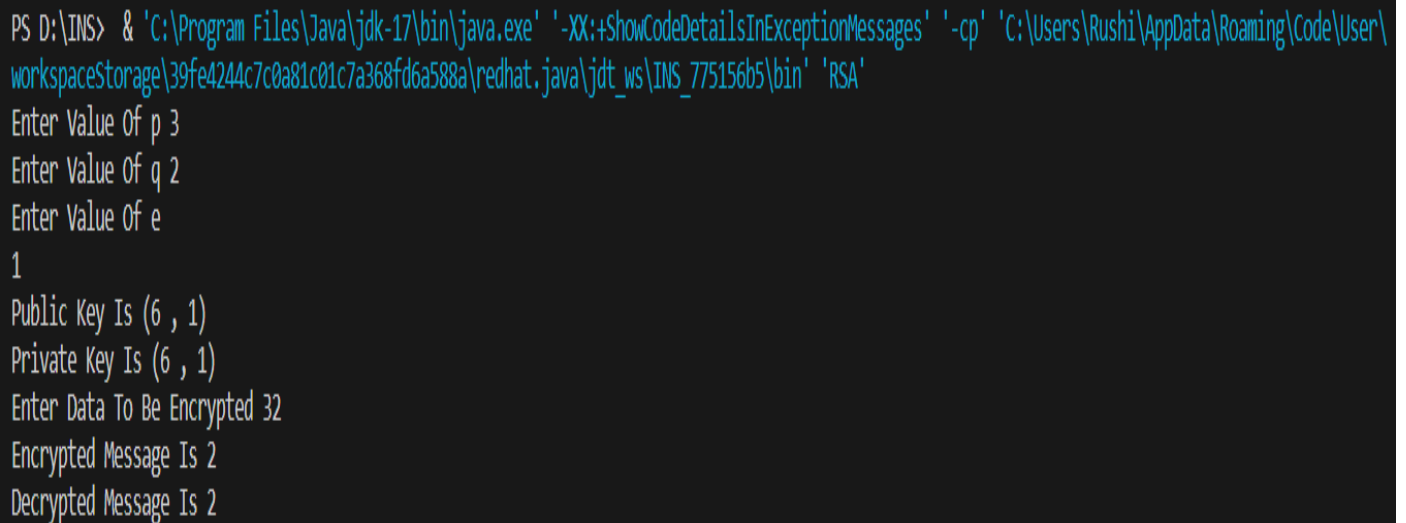
Practical No. 2**Aim: RSA Encryption and Decryption:**

- **Implement the RSA algorithm for public-key encryption and decryption, and explore its properties and security considerations.**

Code:

```
import java.io.*;
public class RSA {
    public static void main(String args[]) throws Exception {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int i = 2, flag = 0, p, q, n, z, e, d = 0, k = 1, data;
        while (true) {
            System.out.print("Enter Value Of p ");
            p = Integer.parseInt(br.readLine());
            for (i = 2; i < p; i++) {
                if ((p % i) == 0)
                {
                    System.out.println("number entered is not prime reenter it");
                    flag = 1;
                }
            }
            if (flag == 0)
                break;
            flag = 0;
        }
        while (true) {
            System.out.print("Enter Value Of q ");
            q = Integer.parseInt(br.readLine());
            for (i = 2; i < q; i++) {
                if ((q % i) == 0) {
                    flag = 1;
                    System.out.println("number entered is not prime reenter it");
                }
            }
            if (flag == 0)
                break;
            flag = 0;
        }
        n = p * q;
```

```
z = (p - 1) * (q - 1);
System.out.println("Enter Value Of e ");
e = Integer.parseInt(br.readLine());
while (true) {
    if (((d * e) % z) == 1)
        break;
    else
        d = d + 1;
}
System.out.println("Public Key Is (" + n + " , " + e + ")");
System.out.println("Private Key Is (" + n + " , " + d + ")");
System.out.print("Enter Data To Be Encrypted ");
data = Integer.parseInt(br.readLine());
long sent = (long) (Math.pow(data, e) % n);
System.out.println("Encrypted Message Is " + sent);
long rec = (long) (Math.pow(sent, d) % n);
System.out.println("Decrypted Message Is " + rec);
}
}
```

Output:

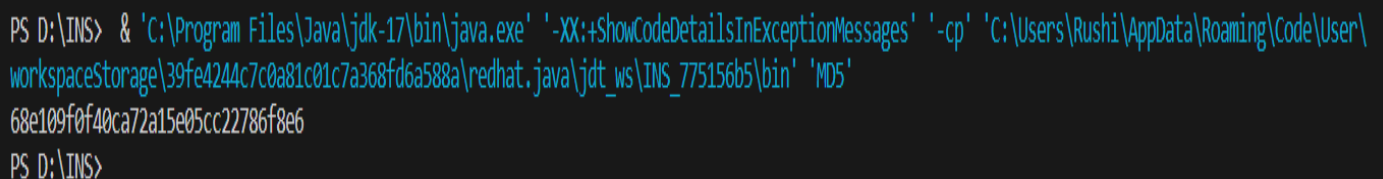
```
PS D:\INS> & 'C:\Program Files\Java\jdk-17\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Rushi\AppData\Roaming\Code\User\workspaceStorage\39fe4244c7c0a81c01c7a368fd6a588a\redhat.java\jdt_ws\INS_775156b5\bin' 'RSA'
Enter Value Of p 3
Enter Value Of q 2
Enter Value Of e
1
Public Key Is (6 , 1)
Private Key Is (6 , 1)
Enter Data To Be Encrypted 32
Encrypted Message Is 2
Decrypted Message Is 2
```

Practical No. 3**Aim: Message Authentication Codes:**

- Implement algorithms to generate and verify message authentication codes (MACs) for ensuring data integrity and authenticity.
- Implementing MD5 to compute message digest.

Code:

```
import java.io.*;
import java.math.BigInteger;
import java.security.MessageDigest;
public class MD5 {
    public static String getMD5(String input) {
        try {
            MessageDigest md = MessageDigest.getInstance("MD5");
            byte[] md1 = md.digest(input.getBytes());
            BigInteger num = new BigInteger(1, md1);
            String hashtext = num.toString(16);
            return hashtext;
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
    public static void main(String a[]) throws IOException {
        System.out.println(getMD5("HelloWorld"));
    }
}
```

Output:

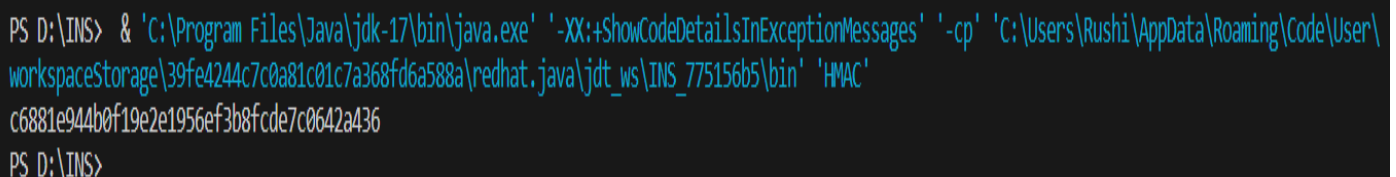
```
PS D:\INS> & 'C:\Program Files\Java\jdk-17\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Rushi\AppData\Roaming\Code\User\workspaceStorage\39fe4244c7c0a81c01c7a368fd6a588a\redhat.java\jdt_ws\INS_775156b5\bin' 'MD5'
68e109f0f40ca72a15e05cc22786f8e6
PS D:\INS>
```

Practical No. 4**Aim: Digital Signatures:**

- **Implement digital signature algorithms such as RSA-based signatures, and verify the integrity and authenticity of digitally signed messages.**
- **Implementing HMAC-SHA 1 signature.**

Code:

```
import java.security.*;
import java.io.*;
import javax.crypto.*;
import javax.crypto.spec.SecretKeySpec;
public class HMAC {
    public static void main(String[] a) throws IOException {
        System.out.println(hmacDigest("the quick brown fox jumps over the lazy dog", "key", "HmacSHA1"));
    }
    public static String hmacDigest(String msg, String keystr, String algo) {
        String digest = null;
        try {
            SecretKeySpec key = new SecretKeySpec((keystr).getBytes("UTF8"), algo);
            Mac m = Mac.getInstance(algo);
            m.init(key);
            byte[] b = m.doFinal(msg.getBytes("ASCII"));
            StringBuffer hash = new StringBuffer();
            for (int i = 0; i < b.length; i++) {
                String hex = Integer.toHexString(0xFF & b[i]);
                if (hex.length() == 1) {
                    hash.append('0');
                }
                hash.append(hex);
            }
            digest = hash.toString();
        } catch (Exception e) {
        }
        return digest;
    }
}
```

Output:

```
PS D:\INS> & 'C:\Program Files\Java\jdk-17\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Rushi\AppData\Roaming\Code\User\workspaceStorage\39fe4244c7c0a81c01c7a368fd6a588a\redhat.java\jdt_ws\INS_775156b5\bin' 'HMAC'
c6881e944b0f19e2e1956ef3b8fcde7c0642a436
PS D:\INS>
```

Practical No. 5**Aim: Key Exchange using Diffie-Hellman:**

- **Implement the Diffie-Hellman key exchange algorithm to securely exchange keys between two entities over an insecure network.**

Code:**a) Sender: Alice**

```
import java.io.*;
import java.net.*;
import java.util.*;
public class dhsend {
    DatagramSocket soc = null;
    int serverport = 9999;
    Scanner sc;
    int x, g, p, r1;
    DatagramPacket revpac;
    DatagramPacket sendpac;
    InetAddress serveradd;
    int clientport;
    byte[] inBuffer;
    byte[] outBuffer;
    public dhsend() {
        try {
            soc = new DatagramSocket();
            InetAddress server = InetAddress.getLocalHost();
            soc.connect(server, serverport);
            System.out.println("ALICE");
        } catch (Exception e) {
        }
    }
    public void KeyGen() {
        sc = new Scanner(System.in);
        System.out.print("enter p=");
        p = sc.nextInt();
        System.out.print("enter g=");
        g = sc.nextInt();
        System.out.print("enter x=");
        x = sc.nextInt();
        r1 = (int) (Math.pow(g, x)) % p;
        System.out.println("R1=" + r1);
    }
    public void send() {
        inBuffer = new byte[500];
        outBuffer = new byte[50];
        try {
            String msg = r1 + "";
            System.out.println("Message send!" + msg);
            outBuffer = msg.getBytes();
            serveradd = soc.getLocalAddress();
            sendpac = new DatagramPacket(outBuffer, outBuffer.length, serveradd, serverport);
            soc.send(sendpac);
        }
    }
}
```

```
    } catch (Exception e) {  
    }  
}  
public void receive() {  
    try {  
        revpac = new DatagramPacket(inBuffer, inBuffer.length);  
        soc.receive(revpac);  
        String msg = new String(revpac.getData(), 0, revpac.getLength());  
        System.out.println("message received" + msg);  
        int temp = (int) (Math.pow((Integer.parseInt(msg)), x));  
        temp = temp % p;  
        int k1 = temp;  
        System.out.println("k1=" + k1);  
    } catch (Exception e) {  
    }  
}  
public static void main(String a[]) {  
    dhsend ds = new dhsend();  
    ds.KeyGen();  
    ds.send();  
    ds.receive();  
}
```

Output:

Try the new cross-platform PowerShell <https://aka.ms/pscore6>

```
PS D:\INS> & 'C:\Program Files\Java\jdk-17\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Rushi\AppData\Roaming\Code\User\workspaceStorage\39fe4244c7c0a81c01c7a368fd6a588a\redhat.java\jdt_ws\INS_775156b5\bin' 'dhsend'  
ALICE  
enter p=23  
enter g=5  
enter x=6  
R1=8  
Message send!8  
message received5  
k1=8  
PS D:\INS> □
```

b) Receiver: Bob

```
import java.io.*;
import java.net.*;
import java.util.*;
public class dhrcv {
    DatagramSocket soc = null;
    int Serverport = 9999;
    Scanner sc;
    int y, g, p, r2;
    DatagramPacket revpac;
    DatagramPacket sendpac;
    InetAddress clientAdd;
    int clientPort;
    byte[] inBuffer;
    byte[] outBuffer;

    public dhrcv() {
        try {
            soc = new DatagramSocket(Serverport);
            System.out.println("BOB");
        } catch (Exception e) {
        }
    }

    public void KeyGen() {
        sc = new Scanner(System.in);
        System.out.print("enter p=");
        p = sc.nextInt();
        System.out.print("enter g=");
        g = sc.nextInt();
        System.out.print("enter y=");
        y = sc.nextInt();
        r2 = (int) (Math.pow(g, y)) % p;
        System.out.println("R2=" + r2);
    }

    public void send() {
        try {
            String msg = r2 + "";
            System.out.println("Message send :" + msg);
            outBuffer = msg.getBytes();
            System.out.println(msg);
            sendpac = new DatagramPacket(outBuffer, outBuffer.length, clientAdd, clientPort);
            soc.send(sendpac);
        } catch (Exception e) {
        }
    }

    public void receive() {
        outBuffer = new byte[500];
        inBuffer = new byte[50];
        try {
            revpac = new DatagramPacket(inBuffer, inBuffer.length);
            soc.receive(revpac);
            clientAdd = revpac.getAddress();
            clientPort = revpac.getPort();
        }
    }
}
```

```
String msg = new String(revpac.getData(), 0, revpac.getLength());
String sendData = msg + "";
System.out.println("message received" + msg);
System.out.println(sendData);
outBuffer = sendData.getBytes();
int k2 = (int) (Math.pow((Integer.parseInt(msg)), y)) % p;
System.out.println("k2=" + k2);
} catch (Exception e) {
}
}
public static void main(String a[]) {
    dhrcv dr = new dhrcv();
    dr.KeyGen();
    dr.receive();
    dr.send();
}
}
```

Output:

Try the new cross-platform PowerShell <https://aka.ms/pscore6>

```
PS D:\INS> & 'C:\Program Files\Java\jdk-17\bin\java.exe' '-XX:+ShowCodeDetailsInException
Messages' '-cp' 'C:\Users\Rushi\AppData\Roaming\Code\User\workspaceStorage\39fe4244c7c0a81
c01c7a368fd6a588a\redhat.java\jdt_ws\INS_775156b5\bin' 'dhrcv'
BOB
enter p=23
enter g=5
enter y=15
R2=5
message received8
8
k2=5
Message send!5
5
PS D:\INS> █
```