

# **RSA ALGORITHM**



An

Object-Oriented Programming Concepts through Java Course

Project Report in partial fulfillment of the degree

**Bachelor of Technology**

in

**Computer Science & Engineering**

**By**

G.Varshitha

2003A51073

Ch.Suchithra

2003A51293

Under the Guidance of

**Dr. T SAMPATH KUMAR**

School of Computer Science and AI

**Submitted to**

**School of CS & AI**





**DEPARTMENT OF COMPUTER SCIENCE &  
ENGINEERING**

**CERTIFICATE**

This is to certify that the **Object Oriented Programming through Java -Course Project** Report entitled “ **RSA ALGORITHM**” is a record of bonafide work carried out by the student **GAJJELA VARSHITHA(2003A51073), CHINTHAPATLA SUCHITHRA(2003A51293)** bearing Roll No(s) 2003A51073, 2003A51293 during the academic year 2022-2023 in partial fulfillment of the award of the degree of *Bachelor of Technology* in **Computer Science & Engineering** by the SR University, Hasanparthy, Warangal.

**Project Guide**

**Head of the Department**

## ACKNOWLEDGEMENT

We owe an enormous debt of gratitude to our project guide **Dr. T Sampath Kumar** , as well as Head of the CSE Department **Dr . M. Sheshikala**, Associate Professor. for guiding us from the beginning through the end of the Capstone Project with their intellectual advices and insightful suggestions. We truly value their consistent feedback on our progress, which always constructive and encouraging and ultimately drove us to the rightdirection.

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved Principal, **Dr.V.MAHESH**, for his continuous support and guidance to complete this project.

Finally, we express our thanks to all the teaching and non-teaching staff of the department for their suggestions and timely support.

## TABLE OF CONTENTS

S.NO	TITLE	PG.NO
1	ABSTRACT	1
2	OBJECTIVE	2
3	DESIGN ELEMENTS IN THE PROJECT	3-4
4	DESIGN	5
5	IMPLEMENTATION	6-9
6	RESULT SCREENS	10-11
7	CONCLUSION	12

## 1.ABSTRACT

Cryptographic technique is one of the principal means to protect information security. Not only has it to ensure the information confidential, but also provides digital signature, authentication, secret sub-storage, system security and other functions. Therefore, the encryption and decryption solution can ensure the confidentiality of the information, as well as the integrity of information and certainty, to prevent information from tampering, forgery and counterfeiting. Encryption and decryption algorithm's security depends on the algorithm while the internal structure of the rigor of mathematics, it also depends on the key confidentiality. Key in the encryption algorithm has a pivotal position, once the key was leaked, it means that anyone can be in the encryption system to encrypt and decrypt information, it means the encryption algorithm is useless. Therefore, what kind of data you choose to be a key, how to distribute the private key, and how to save both data transmission keys are very important issues in the encryption and decryption algorithm. This paper proposed an implementation of a complete and practical RSA encrypt/decrypt solution based on the study of **RSA public key algorithm**. In addition, the encrypt procedure and code implementation is provided in details.

## **2.OBJECTIVE**

Our main aim is to focus on the Encryption Algorithms which provide security to the data either in the network or in the device.

- When a message is being transferred from one device to other ,Protection from the unauthorized access and data leakage should be taken care.
- Even the data is leaked that cannot be in readable format .it should be in understandable format. So this is the,main objective of this algorithm.

So we gone through the RSA Public key Encryption Algorithm .

### 3.DEFINITIONS OF THE ELEMENTS USED IN PROJECT

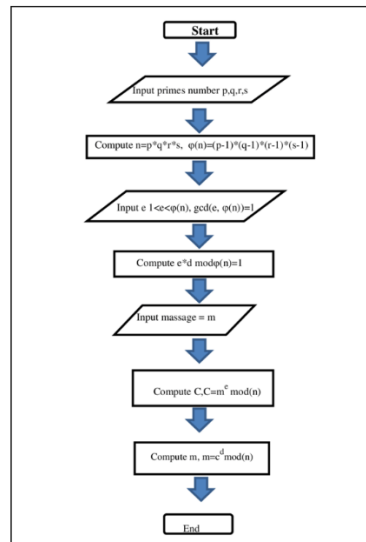
1. **TextField** : The object of a TextField class is a text component that allows the editing of a single line text. It inherits JComponent class.
2. **JLabel** : The object of JLabel class is a component for placing text in a container. It is used to display a single line of read only text. The text can be changed by an application but a user cannot edit it directly. It inherits JComponent class.
3. **JButton**: The JButton class is used to create a labeled button that has platform independent implementation. The application result in some action when the button is pushed. It inherits AbstractButton class.
4. **JFrame**: JFrame class is a type of container which inherits the java.awt.Frame class . JFrame works the main window where components like labels ,buttons, textfields are added to create a GUI.
5. **Action Listener**: The Java Action Listener is notified whenever you click on the button or menu item. It is notified against Action Event. The Action Listener interface is found in java.awt.event package. It has only one method: actionPerformed().
6. **Swings**: Java swing is a part of Java foundation classes ( JFC) that is used to create window based applications It is built on the top of Awt ( abstract window tool kit)API and entirely in java.

- 7. Exceptions:** An Exception is an **Action Event:** An action event occurs, whenever an action is performed by the user. Examples: When the user clicks a button, chooses a menu item, presses Enter in a text field. The result that an action Performed message is sent to all action listeners that are registered on the relevant component.
- 8. Files:** File is an Abstract data type . A named location used to store related Information. There are several File operations like creating a new file, getting information about file, writing in to a file, reading from a file and deleting a file.
- 9. Super():** Super Keyword is a reference variable which is used to refer immediate parent class object. Whenever you create the instance of subclass, an instance of parent class is created implicitly which is referred by super reference variable.
- 10. Exception:** An Exception is an event , which occurs during the execution of a program, that disrupts the normal flow of the programs instructions. When an error occurs within a method creates an object and hands it off to the runtime system.



## 4.DESIGN

### DATA FLOW DIAGRAM



## 5.IMPLEMENTATION

### JAVA CODE

```
import java.math.*;
import javax.swing.*;
import java.awt.Font;
import java.awt.event.*;
import java.awt.Color;
public class RSAAalgo
{
    JFrame frame = new JFrame("RSA ALGORITHM");
    public static void main(String args[]) throws Exception
    {
        new RSAAalgo();
    }
    RSAAalgo() throws Exception
    {
        frame.getContentPane().setLayout(null);
        frame.setBounds(700,700,700,700);
        JLabel lmsg = new JLabel("The Number to Encrypt &Decrypt :");
        lmsg.setBounds(70, 70, 275, 30);
        lmsg.setFont(new Font("Times New Roman", Font.BOLD, 16));
        frame.add(lmsg);

        JTextField msgt = new JTextField();
        msgt.setBounds(320, 70, 275, 30);
        msgt.setFont(new Font("Times New Roman", Font.PLAIN, 16));
        frame.add(msgt);

        JLabel lp = new JLabel("Enter the Value of P :");
        lp.setBounds(70, 140, 275, 30);
        lp.setFont(new Font("Times New Roman", Font.BOLD, 16));
        frame.add(lp);

        JTextField pp = new JTextField();
        pp.setBounds(300, 140, 275, 30);
        pp.setFont(new Font("Times New Roman", Font.PLAIN, 16));
        frame.add(pp);

        JLabel lq = new JLabel("Enter the Value of Q :");
        lq.setBounds(70, 200, 275, 30);
        lq.setFont(new Font("Times New Roman", Font.BOLD, 16));
        frame.add(lq);

        JTextField qq = new JTextField();
        qq.setBounds(300, 200, 275, 30);
```

```
qq.setFont(new Font("Times New Roman", Font.PLAIN, 16));
frame.add(qq);

JLabel pz = new JLabel("Value of Z :");
pz.setBounds(70,260, 275, 30);
pz.setFont(new Font("Times New Roman", Font.BOLD, 16));
frame.add(pz);

JTextField zz = new JTextField();
zz.setBounds(300, 260, 275, 30);
zz.setFont(new Font("Times New Roman", Font.PLAIN, 16));
frame.add(zz);
zz.setBorder(null);
zz.setEditable(false);

JLabel je = new JLabel("Value of E :");
je.setBounds(70,320, 275, 30);
je.setFont(new Font("Times New Roman", Font.BOLD, 16));
frame.add(je);

JTextField ee = new JTextField();
ee.setBounds(300,320, 275, 30);
ee.setFont(new Font("Times New Roman", Font.PLAIN, 16));
frame.add(ee);
ee.setBorder(null);
ee.setEditable(false);

JLabel jd = new JLabel("Value of D :");
jd.setBounds(70,380, 275, 30);
jd.setFont(new Font("Times New Roman", Font.BOLD, 16));
frame.add(jd);

JTextField dd = new JTextField();
dd.setBounds(300, 380, 275, 30);
dd.setFont(new Font("Times New Roman", Font.PLAIN, 16));
frame.add(dd);
dd.setBorder(null);
dd.setEditable(false);

JLabel en = new JLabel("Encrypted Message :");
en.setBounds(70,440, 275, 30);
en.setFont(new Font("Times New Roman", Font.BOLD, 16));
frame.add(en);

JTextField jen = new JTextField();
jen.setBounds(300, 440, 275, 30);
jen.setFont(new Font("Times New Roman", Font.PLAIN, 16));
```

```

frame.add(jen);
jen.setBorder(null);
jen.setEditable(false);

JLabel ed = new JLabel("Decrypted Message :");
ed.setBounds(70,500, 275, 30);
ed.setFont(new Font("Times New Roman", Font.BOLD, 16));
frame.add(ed);

JTextField jed = new JTextField();
jed.setBounds(300, 500, 275, 30);
jed.setFont(new Font("Times New Roman", Font.PLAIN, 16));
frame.add(jed);

jed.setBorder(null);
jed.setEditable(false);
JButton encr = new JButton("Encrypt");
encr.setBounds(350, 550, 150, 30);
frame.add(encr);
encr.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        int p,q,n,z,d=0,e1,i;
        int msg=Integer.parseInt(msgt.getText());
        double c;
        BigInteger msgback;
        //p=3;
        p=Integer.parseInt(pp.getText());
        q = Integer.parseInt(qq.getText());
        //q=11;
        n=p*q;
        z=(p-1)*(q-1);
        zz.setText(String.valueOf(z));
        for(e1=2;e1<z;e1++)
        {
            if(gcd(e1,z)==1)
            {
                break;
            }
        }
        ee.setText(String.valueOf(e1));
        for(i=0;i<=9;i++)
        {
            int x=1+(i*z);
            if(x%e1==0)
            {

```

```

        d=x/e1;
        break;
    }
}
System.out.println(d);
dd.setText(String.valueOf(d));
c=(Math.pow(msg, e1))%n;
jen.setText(String.valueOf(c));
BigInteger N = BigInteger.valueOf(n);
BigInteger C = BigDecimal.valueOf(c).toBigInteger();
msgback=(C.pow(d)).mod(N);
jed.setText(String.valueOf(msgback));
    }
});

JButton clr = new JButton("Reset");
clr.setBounds(150, 550, 150, 30);
frame.add(clr);
clr.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        msgt.setText("");
        pp.setText("");
        qq.setText("");
        zz.setText("");
        ee.setText("");
        dd.setText("");
        jen.setText("");
        jed.setText("");
    }
});


frame.getContentPane().setBackground(Color.PINK);
frame.setVisible(true);

}

public int gcd(int e, int z)
{
    if(e==0)
        return z;
    else
        return gcd(z%e,e);
}
}

```

## 6. RESULTS

 RSA ALGORITHM

The Number to Encrypt & Decrypt :

Enter the Value of P :

Enter the Value of Q :


Value of Z :

Value of E :

Value of D :

Encrypted Message :

Decrypted Message :

 RSA ALGORITHM

The Number to Encrypt & Decrypt :

Enter the Value of P :

Enter the Value of Q :

Value of Z :

Value of E :

Value of D :

Encrypted Message :

Decrypted Message :

# RSA ALGORITHM

The Number to Encrypt & Decrypt :	<input type="text" value="60"/>
Enter the Value of P :	<input type="text" value="113"/>
Enter the Value of Q :	<input type="text" value="151"/>
Value of Z :	<input type="text" value="16800"/>
Value of E :	<input type="text" value="11"/>
Value of D :	<input type="text" value="10691"/>
Encrypted Message :	<input type="text" value="2936.0"/>
Decrypted Message :	<input type="text" value="60"/>

# RSA ALGORITHM

The Number to Encrypt & Decrypt :	<input type="text"/>
Enter the Value of P :	<input type="text"/>
Enter the Value of Q :	<input type="text"/>
Value of Z :	<input type="text"/>
Value of E :	<input type="text"/>
Value of D :	<input type="text"/>
Encrypted Message :	<input type="text"/>
Decrypted Message :	<input type="text"/>

## 7. CONCLUSION

- Finally we want to conclude that, In this project “MODIFIED RSA ALGORITHM” Which is used to protect the information on the common channel as its implementation is simpler and its cryptographic function is faster. **RSA** provides a method to assure the **confidentiality, integrity, authenticity**. And also non-repudiation of electronic communications and data storage.

GITHUB LINK: [varshithagajjela/RSA-Algorithm-JAVA \(github.com\)](https://github.com/varshithagajjela/RSA-Algorithm-JAVA)  
[Suchithra1211/RSA-Algorithm-JAVA \(github.com\)](https://github.com/Suchithra1211/RSA-Algorithm-JAVA)