

OBJECTIVE (Title 1) Write a program to implement Additive Cipher (Z26) with the following conditions: Plaintext should be in lowercase. Ciphertext should be in uppercase. Brute force attack.

CODE –

```
import java.util.Scanner;
public class BruteForce
{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int i, j, l, k = 97, key = 0, flag = 0, index = 0, keyVal;
        String pt;
        char[] ct1 = new char[10];
        char[] pt1 = new char[10];
        char temp;
        System.out.println("ENTER PLAIN TEXT");
        pt = sc.next();
        System.out.println("ENTER KEY VALUE :");
        key = sc.nextInt();
        for (i = 0; i < pt.length(); i++) {
            for (j = 0; j < 26; j++) {
                if (pt.charAt(i) == ' ') {
                    flag = 0;
                    break;
                }
                temp = (char) (j + k);
                if (pt.charAt(i) == temp) {
```

```

        flag = 1;
        index = j;
        break;
    }
}
if (flag == 1) {

    char c = (char) (((index + key) % 26) + 97);
    ct1[i] = c;

}

}
System.out.println("ENCRYPTED DATA:");
for (i = 0; i < pt.length(); i++) {
    System.out.print(ct1[i]);
}
System.out.println("\n" + "DECRYPTION OF DATA USING BRUTE-FORCE
ATTACK :");
key = 1;
while (key <= 26) {
    for (i = 0; i < pt.length(); i++) {
        for (j = 0; j < 26; j++) {
            if (ct1[i] == ' ') {
                flag = 0;
                break;
            }
            temp = (char) (j + k);
            if (ct1[i] == temp) {
                flag = 1;
                index = j;
                break;
            }
        }
        keyVal = index - key;
        if (flag == 1 & keyVal > 0) {
            pt1[i] = (char) ((keyVal % 26) + 97);

        } else if (flag == 1) {
            pt1[i] = (char) ((26 + keyVal) + 97);
        }

    }
    System.out.print("\n" + "DECRYPTED DATA:");
    for (i = 0; i < pt.length(); i++) {
        System.out.print(pt1[i]);
    }
    key++;
}
}
}

```

Output –

```

"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2022.2.1\lib\idea_rt.jar=52856:C:\Program Files\JetBrains\IntelliJ IDEA 202
ENTER PLAIN TEXT
vszcc
ENTER KEY VALUE :
13
ENCRYPTED DATA:
wtaad
DECRYPTION OF DATA USING BRUTE-FORCE ATTACK :

DECRYPTED DATA:vszcc
DECRYPTED DATA:uryyb
DECRYPTED DATA:txxx{

```

DECRYPTED DATA:spwwz

DECRYPTED DATA:rovvy

DECRYPTED DATA:gnuux

DECRYPTED DATA:pmttw

DECRYPTED DATA:olssv

DECRYPTED DATA:nkrvu

DECRYPTED DATA:mjqqt

DECRYPTED DATA:lipps

DECRYPTED DATA:khoor

DECRYPTED DATA:jgnnq

DECRYPTED DATA:ifmmp

DECRYPTED DATA:hello

OBJECTIVE - Title: 2. Write a program to implement Multiplicative Cipher. Plaintext should be in lowercase. Ciphertext should be uppercase. Brute force attack.

CODE –

```
import java.util.*;

class MultiplicativeCipher
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int shift,i,n;
        String str;
        String str1="";
        String str2="";
        System.out.println("Enter the plaintext");
        str=sc.nextLine();
        str=str.toLowerCase();
        n=str.length();
        char ch1[]=str.toCharArray();
        char ch3;
        char ch4;
        System.out.println("Enter the value by which each letter of the
string is to be shifted");
        shift=sc.nextInt();

        System.out.println();
        System.out.println("Encrypted text is");

        for(i=0;i<n;i++)
        {
```

```

        if(Character.isLetter(ch1[i]))
        {
            ch3=(char)((int)ch1[i]*shift-97)%26+97);
            str1=str1+ch3;
        }
        else if(ch1[i]==' ')
        {
            str1=str1+ch1[i];
        }
    }
    System.out.println(str1);

    //Caclulation of multiplicative inverse
    int q=0,flag=0;
    for(i=0;i<26;i++)
    {
        if((i*26+1)%shift==0)
        {
            q=(i*26+1)/shift;
            break;
        }
    }

    System.out.println();
    System.out.println("Decrypted text is");
    char ch2[]=str1.toCharArray();
    for(i=0;i<str1.length();i++)
    {
        if(Character.isLetter(ch2[i]))
        {
            ch4=(char)((int)ch2[i]*q-97)%26+97);
            str2=str2+ch4;
        }

        else if(ch2[i]==' ')
        {
            str2=str2+ch2[i];
        }
    }
    System.out.println(str2);
}
}

```

OUTPUT –

```

MultiplicativeCipher
"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2022.2.1\lib\idea_rt.jar=53448:C:\Program Files\JetBrains\IntelliJ IDEA
Enter the plaintext
hello
Enter the value by which each letter of the string is to be shifted
3
Encrypted text is
hyttc
Decrypted text is
hello

```

TITLE 3 : Write a program to implement Affine Cipher. Plaintext should be in lowercase. Ciphertext should be uppercase. Brute force attack.

Encryption Program:

A screenshot of an IDE window titled 'Main.java'. The code implements a Caesar cipher encryption. It imports java.util.*, defines a Main class with static variables a=17 and b=20, and a Message method that iterates through a character array, shifting each letter by 'a' positions and then adding 'b'. The main method uses a Scanner to read a message and prints the encrypted result.

```
1- import java.util.*;
2
3 public class Main
4 {
5     static int a = 17;
6     static int b = 20;
7
8     static String Message(char[] msg)
9     {
10         String cipher = "";
11         for (int i = 0; i < msg.length; i++)
12         {
13             if (msg[i] != ' ')
14             {
15                 cipher = cipher
16                     + (char) (((a * (msg[i] - 'A')) + b) % 26) + 'A';
17             }
18             else
19             {
20                 cipher += msg[i];
21             }
22         }
23         return cipher;
24     }
25     public static void main(String[] args)
26     {
27         Scanner sc = new Scanner(System.in);
28         System.out.print("Enter the message : ");
29         String msg = sc.nextLine();
30         String cipherText = Message(msg.toCharArray());
31         System.out.println("Encrypted Message is : " + cipherText);
32     }
33 }
34
35
```

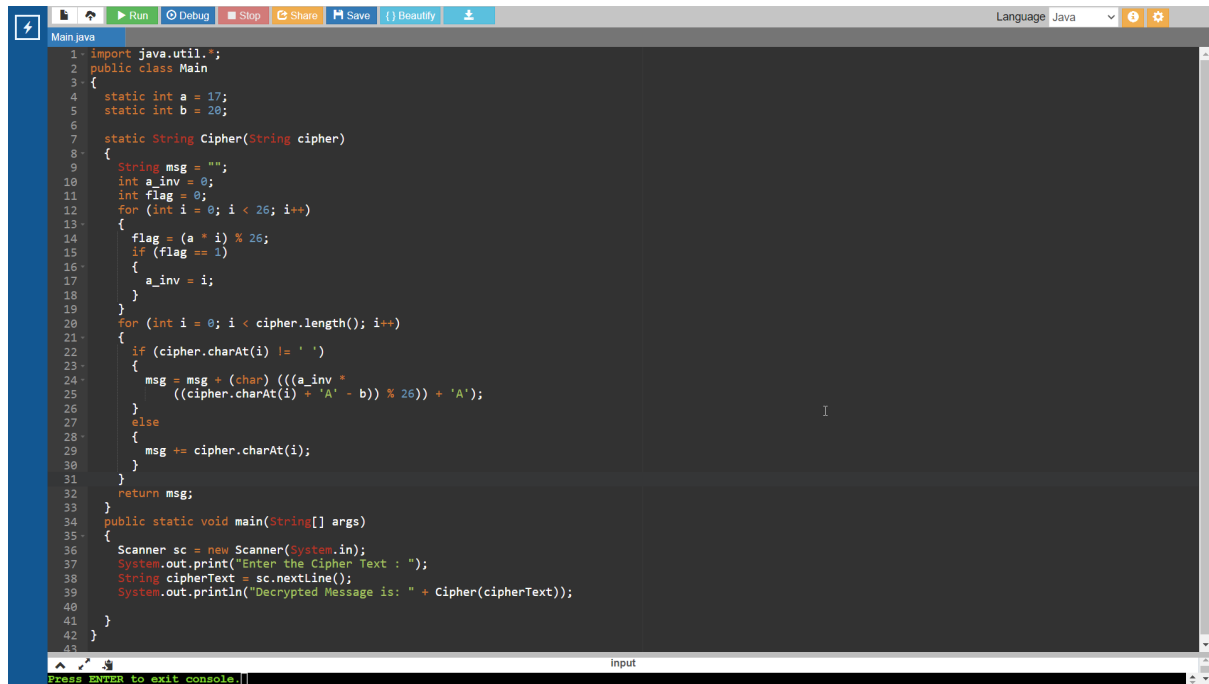
Encryption Output:

A screenshot of a console window titled 'input'. It shows the program's execution: it prompts for a message, receives 'HELLO', and outputs the encrypted message 'JKZZY'. It then shows the program finished with exit code 0 and prompts for the user to press ENTER to exit the console.

```
input
Enter the message : HELLO
Encrypted Message is : JKZZY

...Program finished with exit code 0
Press ENTER to exit console.
```

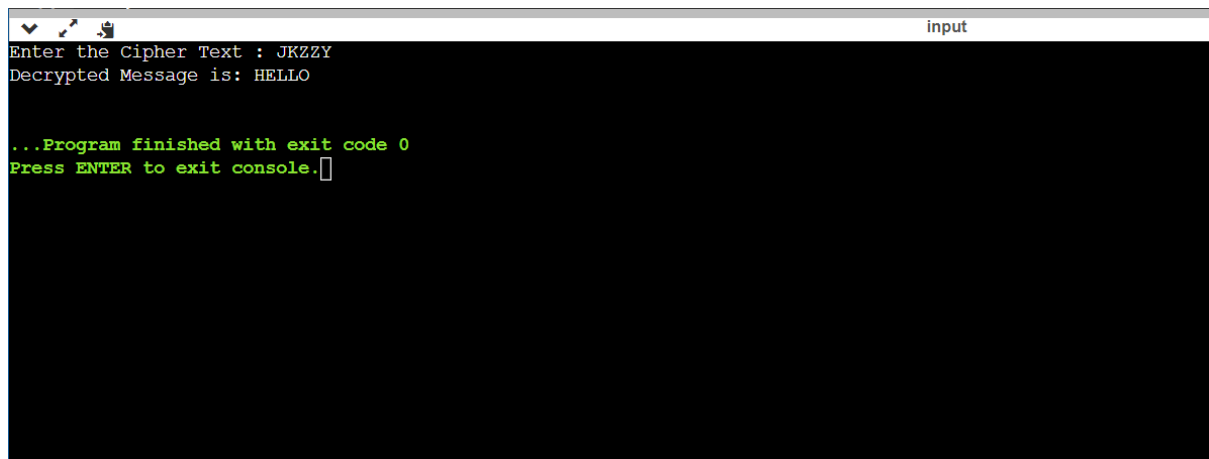
Decryption Program:



The screenshot shows an IDE window titled 'Main.java' with a Java program for decrypting a message. The code defines a static method `Cipher` that takes a string and returns a decrypted message. It uses a Vigenere cipher with a key 'HELLO'. The `main` method prompts the user to enter a cipher text, which is then decrypted and printed.

```
1: import java.util.*;
2: public class Main
3: {
4:     static int a = 17;
5:     static int b = 20;
6:
7:     static String Cipher(String cipher)
8:     {
9:         String msg = "";
10:        int a_inv = 0;
11:        int flag = 0;
12:        for (int i = 0; i < 26; i++)
13:        {
14:            flag = (a * i) % 26;
15:            if (flag == 1)
16:            {
17:                a_inv = i;
18:            }
19:        }
20:        for (int i = 0; i < cipher.length(); i++)
21:        {
22:            if (cipher.charAt(i) != ' ')
23:            {
24:                msg = msg + (char) (((a_inv *
25:                    ((cipher.charAt(i) + 'A' - b)) % 26)) + 'A');
26:            }
27:            else
28:            {
29:                msg += cipher.charAt(i);
30:            }
31:        }
32:        return msg;
33:    }
34:    public static void main(String[] args)
35:    {
36:        Scanner sc = new Scanner(System.in);
37:        System.out.print("Enter the Cipher Text : ");
38:        String cipherText = sc.nextLine();
39:        System.out.println("Decrypted Message is: " + Cipher(cipherText));
40:    }
41: }
42:
43:
Input
Press ENTER to exit console. |
```

Decrypted Output:



The screenshot shows a terminal window titled 'input' with the following output:

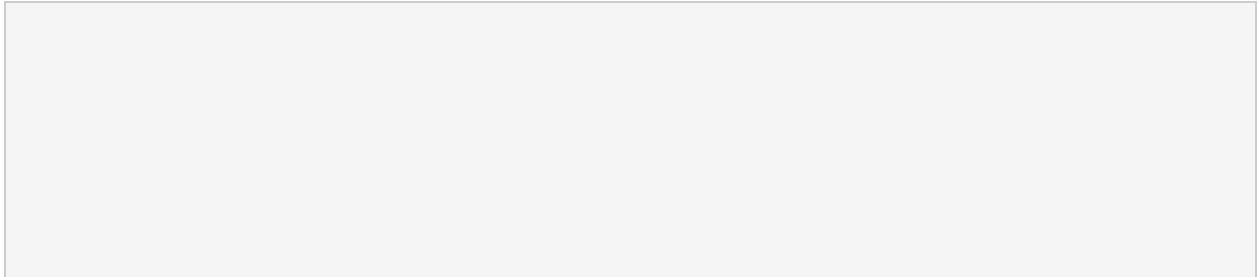
```
Enter the Cipher Text : JKZZY
Decrypted Message is: HELLO

...Program finished with exit code 0
Press ENTER to exit console. |
```


Name-Nikhil Srivastava

Sec-c

U roll num—191500505



Student Corner | El Gamal example with random | Vigenère Cipher - GeeksforGeeks | Firewall Authentication Keepalive | Online Java Compiler - online | New Tab

onlinegdb.com/online_java_compiler

OnlineGDB beta
online compiler and debugger for c/c++
code compile run debug share

IDE
My Projects
Classroom **new**
Learn Programming
Programming Questions
Sign Up
Login

Main.java

```
18 class Main
19 {
20 // This function generates the key in
21 // a cyclic manner until it's length isn't
22 // equal to the length of original text
23 static String generateKey(String str, String key)
24 {
25     int x = str.length();
26     for (int i = 0; i++)
27     {
28         if (x == i)
29             i = 0;
30         if (key.length() == str.length())
31             break;
32         key += (key.charAt(i));
33     }
34     return key;
35 }
36 // This function returns the encrypted text
37 // generated with the help of the key
38 static String cipherText(String str, String key)
39 {
40     String cipher_text="";
41     for (int i = 0; i < str.length(); i++)
42     {
43         // converting in range 0-25
44         int x = (str.charAt(i) + key.charAt(i)) % 26;
45         // convert into alphabets(ASCII)
46         x += 'A';
47         cipher_text += (char)(x);
48     }
49     return cipher_text;
50 }
51 // This function decrypts the encrypted text
52 // and returns the original text
53 static String originalText(String cipher_text, String key)
54 {
55     String orig_text="";
56     for (int i = 0; i < cipher_text.length() &&
57         i < key.length(); i++)
58     {
59         // converting in range 0-25
60         int x = (cipher_text.charAt(i) -
61             key.charAt(i) + 26) % 26;
62         // convert into alphabets(ASCII)
63         x += 'A';
64         orig_text += (char)(x);
65     }
66     return orig_text;
67 }
```

Input

Ciphertext : GCYCZFMYLLEIM

Original/Decrypted Text : GEEKSFORGEEKS

...Program finished with exit code 0
Press ENTER to exit console.

Student Corner | El Gamal example with random | Vigenère Cipher - GeeksforGeeks | Firewall Authentication Keepalive | Online Java Compiler - online | New Tab

onlinegdb.com/online_java_compiler

OnlineGDB beta
online compiler and debugger for c/c++
code compile run debug share

IDE
My Projects
Classroom **new**
Learn Programming
Programming Questions
Sign Up
Login

Main.java

```
47 {
48 // converting in range 0-25
49 int x = (str.charAt(i) + key.charAt(i)) % 26;
50 // convert into alphabets(ASCII)
51 x += 'A';
52 cipher_text += (char)(x);
53 }
54 return cipher_text;
55 }
56 // This function decrypts the encrypted text
57 // and returns the original text
58 static String originalText(String cipher_text, String key)
59 {
60     String orig_text="";
61     for (int i = 0; i < cipher_text.length() &&
62         i < key.length(); i++)
63     {
64         // converting in range 0-25
65         int x = (cipher_text.charAt(i) -
66             key.charAt(i) + 26) % 26;
67         // convert into alphabets(ASCII)
68         x += 'A';
69         orig_text += (char)(x);
70     }
71     return orig_text;
72 }
```

Input

Ciphertext : GCYCZFMYLLEIM

Original/Decrypted Text : GEEKSFORGEEKS

...Program finished with exit code 0
Press ENTER to exit console.

Student Corner | El Gamal example with random | Vigenère Cipher - GeeksforGeeks | Firewall Authentication Keepalive | Online Java Compiler - online | New Tab

onlinegdb.com/online_java_compiler

OnlineGDB beta
online compiler and debugger for c/c++
code compile run debug share

IDE
My Projects
Classroom **new**
Learn Programming
Programming Questions
Sign Up
Login

Main.java

```
47 {
48 // converting in range 0-25
49 int x = (str.charAt(i) + key.charAt(i)) % 26;
50 // convert into alphabets(ASCII)
51 x += 'A';
52 cipher_text += (char)(x);
53 }
54 return cipher_text;
55 }
56 // This function decrypts the encrypted text
57 // and returns the original text
58 static String originalText(String cipher_text, String key)
59 {
60     String orig_text="";
61     for (int i = 0; i < cipher_text.length() &&
62         i < key.length(); i++)
63     {
64         // converting in range 0-25
65         int x = (cipher_text.charAt(i) -
66             key.charAt(i) + 26) % 26;
67         // convert into alphabets(ASCII)
68         x += 'A';
69         orig_text += (char)(x);
70     }
71     return orig_text;
72 }
```

Input

Ciphertext : GCYCZFMYLLEIM

Original/Decrypted Text : GEEKSFORGEEKS

...Program finished with exit code 0
Press ENTER to exit console.

Student Corner

El Gamal example with random i

Vigenère Cipher - GeeksforGeeks

Firewall Authentication Keepalive

Online Java Compiler - online

New Tab

onlinegdb.com/online_java_compiler

Language Java

OnlineGDB beta
online compiler and debugger for c/c++
code, compile, run, debug, share.
IDE
My Projects
Classroom **new**
Learn Programming
Programming Questions
Sign Up
Login

```
76 } return orig_text;
77 }
78
79 // This function will convert the lower case character to Upper case
80 static String LowerToUpper(String s)
81 {
82     StringBuffer str =new StringBuffer(s);
83     for(int i = 0; i < s.length(); i++)
84     {
85         if(Character.isLowerCase(s.charAt(i)))
86         {
87             str.setCharAt(i, Character.toUpperCase(s.charAt(i)));
88         }
89     }
90     s = str.toString();
91     return s;
92 }
93
94 // Driver code
95 public static void main(String[] args)
96 {
97     String Str = "GEEKSFORGEEKS";
98     String Keyword = "AYUSH";
99
100     String str = LowerToUpper(Str);
101     String keyword = LowerToUpper(Keyword);
102
103     String key = generateKey(str, keyword);
104     String cipher_text = cipherText(str, key);
105
106     System.out.println("CipherText : "
107         + cipher_text + "\n");
108
109     System.out.println("Original/Decrypted Text : "
110         + originalText(cipher_text, key));
111 }
112 }
```

Input

Ciphertext : GCYCZFMYLLEIM

Original/Decrypted Text : GEEKSFORGEEKS

...Program finished with exit code 0
Press ENTER to exit console.

Student Corner

El Gamal example with random i

Vigenère Cipher - GeeksforGeeks

Firewall Authentication Keepalive

Online Java Compiler - online

New Tab

onlinegdb.com/online_java_compiler

Language Java

OnlineGDB beta
online compiler and debugger for c/c++
code, compile, run, debug, share.
IDE
My Projects
Classroom **new**
Learn Programming
Programming Questions
Sign Up
Login

```
83 for(int i = 0; i < s.length(); i++)
84 {
85     if(Character.isLowerCase(s.charAt(i)))
86     {
87         str.setCharAt(i, Character.toUpperCase(s.charAt(i)));
88     }
89 }
90 s = str.toString();
91 return s;
92 }
93
94 // Driver code
95 public static void main(String[] args)
96 {
97     String Str = "GEEKSFORGEEKS";
98     String Keyword = "AYUSH";
99
100     String str = LowerToUpper(Str);
101     String keyword = LowerToUpper(Keyword);
102
103     String key = generateKey(str, keyword);
104     String cipher_text = cipherText(str, key);
105
106     System.out.println("CipherText : "
107         + cipher_text + "\n");
108
109     System.out.println("Original/Decrypted Text : "
110         + originalText(cipher_text, key));
111 }
112 }
```

Input

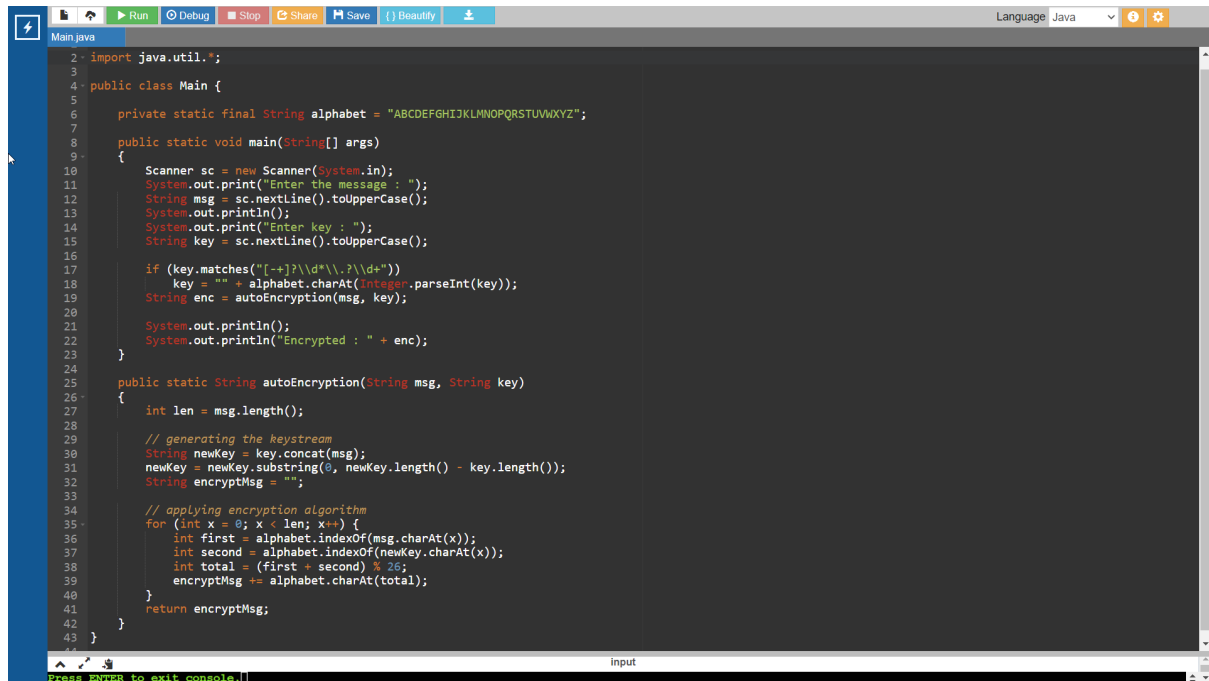
Ciphertext : GCYCZFMYLLEIM

Original/Decrypted Text : GEEKSFORGEEKS

...Program finished with exit code 0
Press ENTER to exit console.

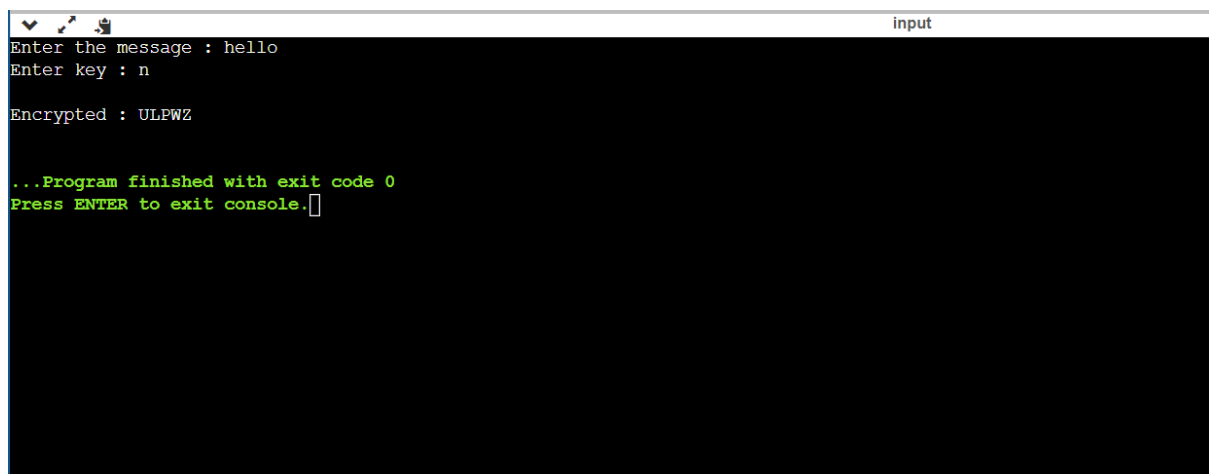
TITLE 5 : Write a program to implement Autokey Cipher. Plaintext should be in lowercase. Ciphertext should be uppercase. Brute force attack.

Encryption Program:

A screenshot of an IDE window titled 'Main.java'. The code is a Java program for encryption. It imports java.util.* and defines a public class Main. Inside the class, there is a private static final String alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ;". The main method takes a String[] args and uses a Scanner to get a message and a key from the user. The message is converted to uppercase. The key is also converted to uppercase and validated against the alphabet. The program then calls a static method autoEncryption(msg, key). This method generates a keystream by concatenating the message and key, then truncating it to the length of the message. It then applies a Vigenere cipher algorithm by adding the message characters to the keystream characters modulo 26. The result is the encrypted message. The console at the bottom shows the input 'hello' and 'n', resulting in the encrypted output 'ULPWZ'.

```
2 import java.util.*;
3
4 public class Main {
5
6     private static final String alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ;";
7
8     public static void main(String[] args)
9     {
10         Scanner sc = new Scanner(System.in);
11         System.out.print("Enter the message : ");
12         String msg = sc.nextLine().toUpperCase();
13         System.out.println();
14         System.out.print("Enter key : ");
15         String key = sc.nextLine().toUpperCase();
16
17         if (key.matches("[a-zA-Z\\d*\\.\\?\\d+]"))
18             key = "" + alphabet.charAt(Integer.parseInt(key));
19         String enc = autoEncryption(msg, key);
20
21         System.out.println();
22         System.out.println("Encrypted : " + enc);
23     }
24
25     public static String autoEncryption(String msg, String key)
26     {
27         int len = msg.length();
28
29         // generating the keystream
30         String newKey = key.concat(msg);
31         newKey = newKey.substring(0, newKey.length() - key.length());
32         String encryptMsg = "";
33
34         // applying encryption algorithm
35         for (int x = 0; x < len; x++) {
36             int first = alphabet.indexOf(msg.charAt(x));
37             int second = alphabet.indexOf(newKey.charAt(x));
38             int total = (first + second) % 26;
39             encryptMsg += alphabet.charAt(total);
40         }
41         return encryptMsg;
42     }
43 }
```

Encryption Output:

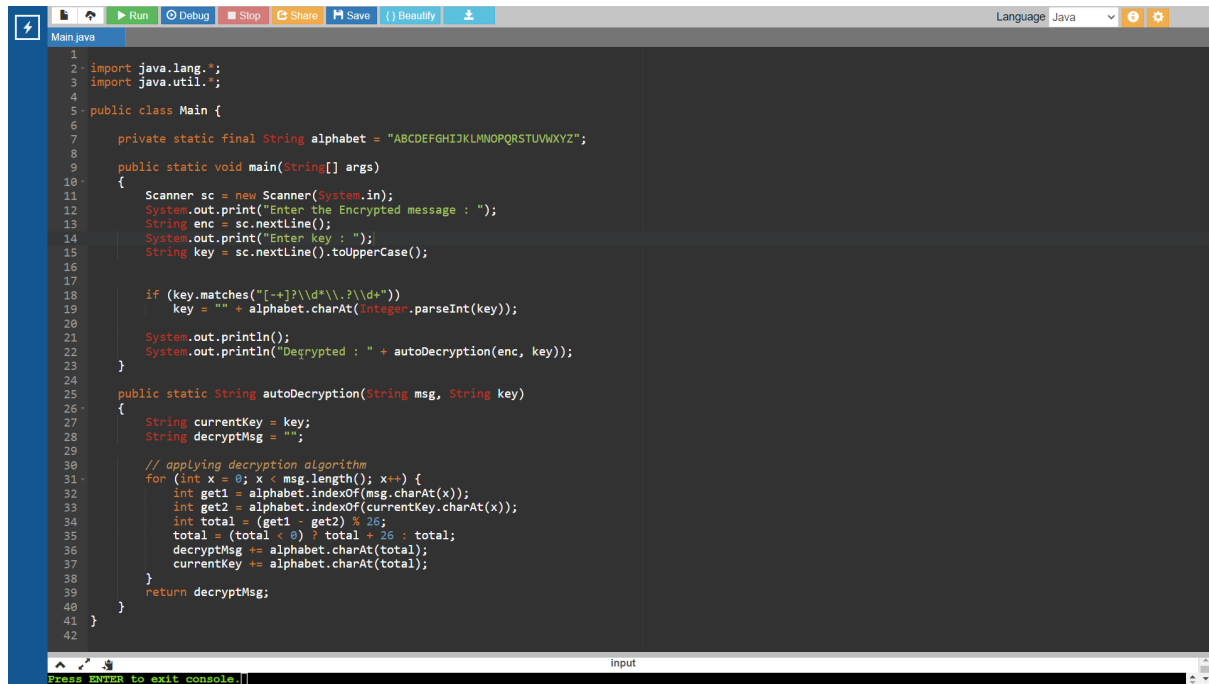
A screenshot of a console window titled 'input'. It shows the execution of the encryption program. The user enters 'hello' for the message and 'n' for the key. The program outputs 'ULPWZ' as the encrypted message. At the end, it shows '...Program finished with exit code 0' and 'Press ENTER to exit console.'.

```
Enter the message : hello
Enter key : n

Encrypted : ULPWZ

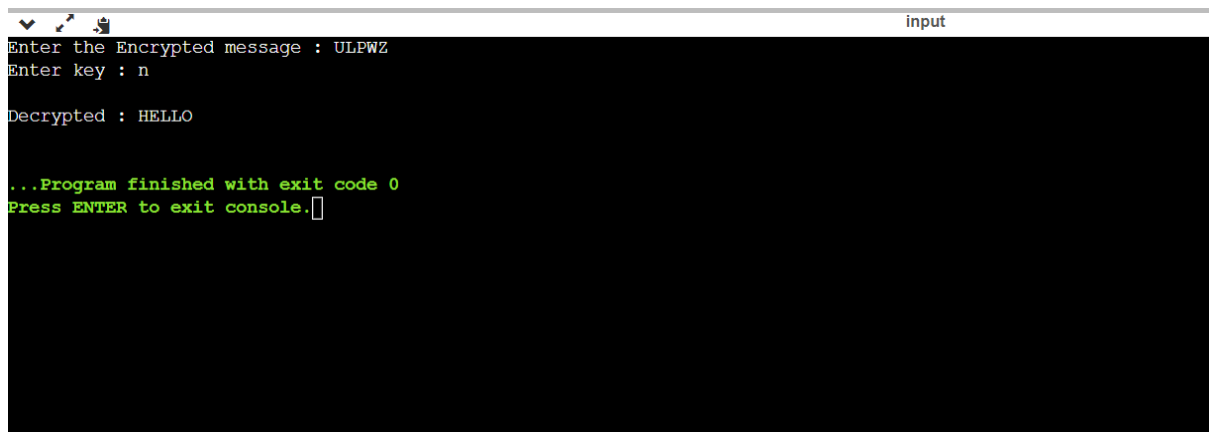
...Program finished with exit code 0
Press ENTER to exit console.
```

Decryption Program:



```
1
2 import java.lang.*;
3 import java.util.*;
4
5 public class Main {
6
7     private static final String alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
8
9     public static void main(String[] args)
10    {
11        Scanner sc = new Scanner(System.in);
12        System.out.print("Enter the Encrypted message : ");
13        String enc = sc.nextLine();
14        System.out.print("Enter key : ");
15        String key = sc.nextLine().toUpperCase();
16
17        if (key.matches("[~+]?\\d*\\.?\\d+"))
18            key = "" + alphabet.charAt(Integer.parseInt(key));
19
20        System.out.println();
21        System.out.println("Decrypted : " + autoDecryption(enc, key));
22    }
23
24    public static String autoDecryption(String msg, String key)
25    {
26        String currentKey = key;
27        String decryptMsg = "";
28
29        // applying decryption algorithm
30        for (int x = 0; x < msg.length(); x++) {
31            int get1 = alphabet.indexOf(msg.charAt(x));
32            int get2 = alphabet.indexOf(currentKey.charAt(x));
33            int total = (get1 - get2) % 26;
34            total = (total < 0) ? total + 26 : total;
35            decryptMsg += alphabet.charAt(total);
36            currentKey += alphabet.charAt(total);
37        }
38        return decryptMsg;
39    }
40
41 }
42
```

Decrypted Output:



```
input
Enter the Encrypted message : ULPWZ
Enter key : n

Decrypted : HELLO

...Program finished with exit code 0
Press ENTER to exit console.
```

OBJECTIVE (Title 6) - Write a program to implement Playfair Cipher to encrypt & decrypt the given message where the key matrix can be formed by using a given keyword.

CODE: - Encryption using shift key

```
import java.io.*;
import java.util.*;

public class Encryption { //to keep track of index
    public static final String alpha = "abcdefghijklmnopqrstuvwxyz";

    public static String encrypt(String message, int shiftKey) {
        message = message.toLowerCase();
        String cipherText = "";
        for (int i = 0; i < message.length(); i++) {
            int charPosition = alpha.indexOf(message.charAt(i));
            int keyVal = (shiftKey + charPosition) % 26;
            char replaceVal = alpha.charAt(keyVal);
            cipherText += replaceVal;
        }
        return cipherText;
    }

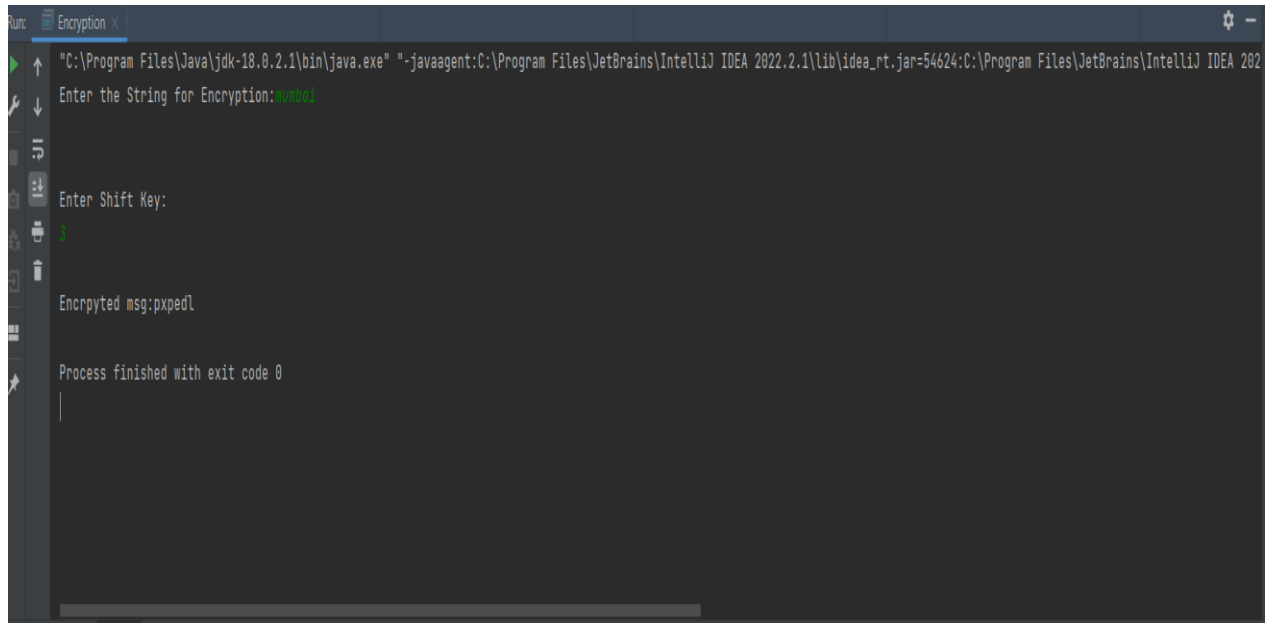
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String message = new String();
        int key = 0;
        System.out.print("Enter the String for Encryption:");
        message = sc.next();
    }
}
```

```

        System.out.println("\n\nEnter Shift Key:");
        key = sc.nextInt();
        System.out.println("\nEncrypted msg:" + encrypt(message, key));
    } //main method ends
} //Main Class End

```

Output: -



```

C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2022.2.1\lib\idea_rt.jar=54624:C:\Program Files\JetBrains\IntelliJ IDEA 202
Enter the String for Encryption:xxxxx
Enter Shift Key:
Encrypted msg:pxpedl
Process finished with exit code 0

```

Decryption using shift key: -

CODE –

```

import java.io.*;
import java.util.*;
public class Decryption { //to keep track of index
    public static final String ALPHABET = "abcdefghijklmnopqrstuvwxyz";

    public static String decrypt(String cipherText, int shiftKey) {
        cipherText = cipherText.toLowerCase();
        String message = "";
        for (int i = 0; i < cipherText.length(); i++) {
            int charPosition = ALPHABET.indexOf(cipherText.charAt(i));
            int keyVal = (charPosition - shiftKey) % 26;
            if (keyVal < 0) {
                keyVal = ALPHABET.length() + keyVal;
            }
            char replaceVal = ALPHABET.charAt(keyVal);
            message += replaceVal;
        }
        return message;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
    }
}

```



```

String message = new String();
int key = 0;
System.out.print("Enter the String for Decryption:");
message = sc.next();

System.out.println("\n\nEnter Shift Key:");
key = sc.nextInt();
// System.out.println("\nEncrpyted msg:"+encrypt(message, key));
System.out.println("\nDecrypted Message:" + decrypt(message, key));

}
}

```

Output –



```

Decryption X
"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2022.2.1\lib\idea_rt.jar=54704:C:\Program Files\JetBrains\IntelliJ IDEA 202
Enter the String for Decryption:mumbai
Enter Shift Key:
3
Decrypted Message:mumbai
Process finished with exit code 0

```

Name-Nikhil Srivastava

Sec-c

U roll num—191500505

```
1. import java.util.ArrayList;
2. import java.util.Scanner;
3. public class HillCipherExample {
4.     //method to accept key matrix
5.     private static int[][] getKeyMatrix() {
6.         Scanner sc = new Scanner(System.in);
7.         System.out.println("Enter key matrix:");
8.         String key = sc.nextLine();
9.         //int len = key.length();
10.        double sq = Math.sqrt(key.length());
11.        if (sq != (long) sq) {
12.            System.out.println("Cannot Form a square matrix");
13.        }
14.        int len = (int) sq;
15.        int[][] keyMatrix = new int[len][len];
16.        int k = 0;
17.        for (int i = 0; i < len; i++)
18.        {
19.            for (int j = 0; j < len; j++)
20.            {
21.                keyMatrix[i][j] = ((int) key.charAt(k)) - 97;
22.                k++;
23.            }
24.        }
25.        return keyMatrix;
```

```

26. }
27. // Below method checks whether the key matrix is valid (det=0)
28. private static void isValidMatrix(int[][] keyMatrix) {
29.     int det = keyMatrix[0][0] * keyMatrix[1][1] - keyMatrix[0][1] * keyMatrix[1]
        [0];
30.     // If det=0, throw exception and terminate
31.     if(det == 0) {
32.         throw new java.lang.Error("Det equals to zero, invalid key matrix!");
33.     }
34. }
35. // This method checks if the reverse key matrix is valid (matrix mod26 = (1,0
    ,0,1)
36. private static void isValidReverseMatrix(int[][] keyMatrix, int[][] reverse
    Matrix) {
37.     int[][] product = new int[2][2];
38.     // Find the product matrix of key matrix times reverse key matrix
39.     product[0][0] = (keyMatrix[0][0]*reverseMatrix[0][0] + keyMatrix[0][1] * re
        verseMatrix[1][0]) % 26;
40.     product[0][1] = (keyMatrix[0][0]*reverseMatrix[0][1] + keyMatrix[0][1] * re
        verseMatrix[1][1]) % 26;
41.     product[1][0] = (keyMatrix[1][0]*reverseMatrix[0][0] + keyMatrix[1][1] * re
        verseMatrix[1][0]) % 26;
42.     product[1][1] = (keyMatrix[1][0]*reverseMatrix[0][1] + keyMatrix[1][1] * re
        verseMatrix[1][1]) % 26;
43.     // Check if a=1 and b=0 and c=0 and d=1
44.     // If not, throw exception and terminate
45.     if(product[0][0] != 1 || product[0][1] != 0 || product[1][0] != 0 || product[1]
        [1] != 1) {
46.         throw new java.lang.Error("Invalid reverse matrix found!");
47.     }
48. }
49. // This method calculates the reverse key matrix
50. private static int[][] reverseMatrix(int[][] keyMatrix) {
51.     int detmod26 = (keyMatrix[0][0] * keyMatrix[1][1] - keyMatrix[0][1] * key
        Matrix[1][0]) % 26; // Calc det
52.     int factor;
53.     int[][] reverseMatrix = new int[2][2];

```

```

54. // Find the factor for which is true that
55. // factor*det = 1 mod 26
56. for(factor=1; factor < 26; factor++)
57. {
58.     if((detmod26 * factor) % 26 == 1)
59.     {
60.         break;
61.     }
62. }
63. // Calculate the reverse key matrix elements using the factor found
64. reverseMatrix[0][0] = keyMatrix[1][1] * factor % 26;
65. reverseMatrix[0][1] = (26 - keyMatrix[0][1]) * factor % 26;
66. reverseMatrix[1][0] = (26 - keyMatrix[1][0]) * factor % 26;
67. reverseMatrix[1][1] = keyMatrix[0][0] * factor % 26;
68. return reverseMatrix;
69. }
70. // This method echoes the result of encrypt/decrypt
71. private static void echoResult(String label, int adder, ArrayList<Integer> p
    hrase) {
72.     int i;
73.     System.out.print(label);
74.     // Loop for each pair
75.     for(i=0; i < phrase.size(); i += 2) {
76.         System.out.print(Character.toChars(phrase.get(i) + (64 + adder)));
77.         System.out.print(Character.toChars(phrase.get(i+1) + (64 + adder)));
78.         if(i+2 < phrase.size()) {
79.             System.out.print("-");
80.         }
81.     }
82.     System.out.println();
83. }
84. // This method makes the actual encryption
85. public static void encrypt(String phrase, boolean alphaZero)
86. {
87.     int i;
88.     int adder = alphaZero ? 1 : 0; // For calculations depending on the alpha
    bet

```

```

89.     int[][] keyMatrix;
90.     ArrayList<Integer> phraseToNum = new ArrayList<>();
91.     ArrayList<Integer> phraseEncoded = new ArrayList<>();
92.     // Delete all non-english characters, and convert phrase to upper case
93.     phrase = phrase.replaceAll("[^a-zA-Z]", "").toUpperCase();
94.
95.     // If phrase length is not an even number, add "Q" to make it even
96.     if(phrase.length() % 2 == 1) {
97.         phrase += "Q";
98.     }
99.     // Get the 2x2 key matrix from sc
100.        keyMatrix = getKeyMatrix();
101.        // Check if the matrix is valid (det != 0)
102.        isValidMatrix(keyMatrix);
103.        // Convert characters to numbers according to their
104.        // place in ASCII table minus 64 positions (A=65 in ASCII table)
105.        // If we use A=0 alphabet, subtract one more (adder)
106.        for(i=0; i < phrase.length(); i++) {
107.            phraseToNum.add(phrase.charAt(i) - (64 + adder));
108.        }
109.        // Find the product per pair of the phrase with the key matrix mod
        ulo 26
110.        // If we use A=1 alphabet and result is 0, replace it with 26 (Z)
111.        for(i=0; i < phraseToNum.size(); i += 2) {
112.            int x = (keyMatrix[0][0] * phraseToNum.get(i) + keyMatrix[0][1]
        * phraseToNum.get(i+1)) % 26;
113.            int y = (keyMatrix[1][0] * phraseToNum.get(i) + keyMatrix[1][1]
        * phraseToNum.get(i+1)) % 26;
114.            phraseEncoded.add(alphaZero ? x : (x == 0 ? 26 : x));
115.            phraseEncoded.add(alphaZero ? y : (y == 0 ? 26 : y));
116.        }
117.        // Print the result
118.        echoResult("Encoded phrase: ", adder, phraseEncoded);
119.    }
120.    // This method makes the actual decryption
121.    public static void decrypt(String phrase, boolean alphaZero)
122.    {

```

```

123.         int i, adder = alphaZero ? 1 : 0;
124.         int[][] keyMatrix, revKeyMatrix;
125.         ArrayList<Integer> phraseToNum = new ArrayList<>();
126.         ArrayList<Integer> phraseDecoded = new ArrayList<>();
127.         // Delete all non-
            english characters, and convert phrase to upper case
128.         phrase = phrase.replaceAll("[^a-zA-Z]", "").toUpperCase();
129.
130.         // Get the 2x2 key matrix from sc
131.         keyMatrix = getKeyMatrix();
132.         // Check if the matrix is valid (det != 0)
133.         isValidMatrix(keyMatrix);
134.         // Convert numbers to characters according to their
135.         // place in ASCII table minus 64 positions (A=65 in ASCII table)
136.         // If we use A=0 alphabet, subtract one more (adder)
137.         for(i=0; i < phrase.length(); i++) {
138.             phraseToNum.add(phrase.charAt(i) - (64 + adder));
139.         }
140.         // Find the reverse key matrix
141.         revKeyMatrix = reverseMatrix(keyMatrix);
142.         // Check if the reverse key matrix is valid (product = 1,0,0,1)
143.         isValidReverseMatrix(keyMatrix, revKeyMatrix);
144.         // Find the product per pair of the phrase with the reverse key mat
            rix modulo 26
145.         for(i=0; i < phraseToNum.size(); i += 2) {
146.             phraseDecoded.add((revKeyMatrix[0][0] * phraseToNum.get(i) +
                revKeyMatrix[0][1] * phraseToNum.get(i+1)) % 26);
147.             phraseDecoded.add((revKeyMatrix[1][0] * phraseToNum.get(i) +
                revKeyMatrix[1][1] * phraseToNum.get(i+1)) % 26);
148.         }
149.         // Print the result
150.         echoResult("Decoded phrase: ", adder, phraseDecoded);
151.     }
152.     //main method
153.     public static void main(String[] args) {
154.         String opt, phrase;
155.         byte[] p;

```

```

156. Scanner sc = new Scanner(System.in);
157. System.out.println("Hill Cipher Implementation (2x2)");
158. System.out.println("-----");
159. System.out.println("1. Encrypt text (A=0,B=1,...Z=25)");
160. System.out.println("2. Decrypt text (A=0,B=1,...Z=25)");
161. System.out.println("3. Encrypt text (A=1,B=2,...Z=26)");
162. System.out.println("4. Decrypt text (A=1,B=2,...Z=26)");
163. System.out.println();
164. System.out.println("Type any other character to exit");
165. System.out.println();
166. System.out.print("Select your choice: ");
167. opt = sc.nextLine();
168. switch (opt)
169. {
170.     case "1":
171.         System.out.print("Enter phrase to encrypt: ");
172.         phrase = sc.nextLine();
173.         encrypt(phrase, true);
174.         break;
175.     case "2":
176.         System.out.print("Enter phrase to decrypt: ");
177.         phrase = sc.nextLine();
178.         decrypt(phrase, true);
179.         break;
180.     case "3":
181.         System.out.print("Enter phrase to encrypt: ");
182.         phrase = sc.nextLine();
183.         encrypt(phrase, false);
184.         break;
185.     case "4":
186.         System.out.print("Enter phrase to decrypt: ");
187.         phrase = sc.nextLine();
188.         decrypt(phrase, false);
189.         break;
190. }
191. }
192. }

```

```
C:\Users\Anurati\Desktop\abcDemo>javac HillCipherExample.java
```

```
C:\Users\Anurati\Desktop\abcDemo>java HillCipherExample  
Hill Cipher Implementation (2x2)
```

```
-----
```

1. Encrypt text (A=0,B=1,...Z=25)
2. Decrypt text (A=0,B=1,...Z=25)
3. Encrypt text (A=1,B=2,...Z=26)
4. Decrypt text (A=1,B=2,...Z=26)

```
Type any other character to exit
```

```
Select your choice: 1
```

```
Enter phrase to encrypt: hillcipheralgorithm
```

```
Enter key matrix:
```

```
nwkc
```

```
Encoded phrase: HI-VC-UK-LI-KW-IW-WK-HE-LW-OW
```


Name-Nikhil Srivastava

Sec-c

U roll num—191500505

```
1.
   package com.sanfoundry.setandstring;
2.
3. public class TranspositionCipher
4. {
5.     public static String selectedKey;
6.     public static char   sortedKey[];
7.     public static int    sortedKeyPos[];
8.
9.     // default constructor define the default key
10.    public TranspositionCipher()
11.    {
12.        selectedKey = "megabuck";
13.        sortedKeyPos = new int[selectedKey.length()];
14.        sortedKey = selectedKey.toCharArray();
15.    }
16.
17.    // Parameterized constructor define the custom key
18.    public TranspositionCipher(String myKey)
19.    {
20.        selectedKey = myKey;
21.        sortedKeyPos = new int[selectedKey.length()];
22.        sortedKey = selectedKey.toCharArray();
23.    }
24.
25.    // To reorder data do the sorting on selected key
26.    public static void doProcessOnKey()
27.    {
28.        // Find position of each character in selected key and arrange it on
29.        // alphabetical order
30.        int min, i, j;
31.        char originalKey[] = selectedKey.toCharArray();
32.        char temp;
33.        // First Sort the array of selected key
34.        for (i = 0; i < selectedKey.length(); i++)
35.        {
36.            min = i;
```

```

37.         for (j = i; j < selectedKey.length(); j++)
38.         {
39.             if (sortedKey[min] > sortedKey[j])
40.             {
41.                 min = j;
42.             }
43.         }
44.         if (min != i)
45.         {
46.             temp = sortedKey[i];
47.             sortedKey[i] = sortedKey[min];
48.             sortedKey[min] = temp;
49.         }
50.     }
51.     // Fill the position of array according to alphabetical order
52.     for (i = 0; i < selectedKey.length(); i++)
53.     {
54.         for (j = 0; j < selectedKey.length(); j++)
55.         {
56.             if (originalKey[i] == sortedKey[j])
57.                 sortedKeyPos[i] = j;
58.         }
59.     }
60. }
61.
62. // to encrypt the targeted string
63. public static String doEncryption(String plainText)
64. {
65.     int min, i, j;
66.     char originalKey[] = selectedKey.toCharArray();
67.     char temp;
68.     doProcessOnKey();
69.     // Generate encrypted message by doing encryption using Transposition
70.     // Cipher
71.     int row = plainText.length() / selectedKey.length();
72.     int extrabit = plainText.length() % selectedKey.length();
73.     int exrow = (extrabit == 0) ? 0 : 1;
74.     int rowtemp = -1, coltemp = -1;
75.     int totallen = (row + exrow) * selectedKey.length();
76.     char pmat[][] = new char[(row + exrow)][selectedKey.length()];
77.     char encry[] = new char[totallen];
78.     int tempcnt = -1;
79.     row = 0;
80.     for (i = 0; i < totallen; i++)
81.     {
82.         coltemp++;

```

```

83.         if (i < plainText.length())
84.         {
85.             if (coltemp == (selectedKey.length()))
86.             {
87.                 row++;
88.                 coltemp = 0;
89.             }
90.             pmat[row][coltemp] = plainText.charAt(i);
91.         }
92.         else
93.         { // do the padding ...
94.             pmat[row][coltemp] = '*';
95.         }
96.     }
97.     int len = -1, k;
98.     for (i = 0; i < selectedKey.length(); i++)
99.     {
100.         for (k = 0; k < selectedKey.length(); k++)
101.         {
102.             if (i == sortedKeyPos[k])
103.             {
104.                 break;
105.             }
106.         }
107.         for (j = 0; j <= row; j++)
108.         {
109.             len++;
110.             encry[len] = pmat[j][k];
111.         }
112.     }
113.     String p1 = new String(encry);
114.     return (new String(p1));
115. }
116.
117. // to decrypt the targeted string
118. public static String doDecryption(String s)
119. {
120.     int min, i, j, k;
121.     char key[] = selectedKey.toCharArray();
122.     char encry[] = s.toCharArray();
123.     char temp;
124.     doProcessOnKey();
125.     // Now generating plain message
126.     int row = s.length() / selectedKey.length();
127.     char pmat[][] = new char[row][(selectedKey.length())];
128.     int tempcnt = -1;

```

```

129.         for (i = 0; i < selectedKey.length(); i++)
130.         {
131.             for (k = 0; k < selectedKey.length(); k++)
132.             {
133.                 if (i == sortedKeyPos[k])
134.                 {
135.                     break;
136.                 }
137.             }
138.             for (j = 0; j < row; j++)
139.             {
140.                 tempcnt++;
141.                 pmat[j][k] = encry[tempcnt];
142.             }
143.         }
144.         // store matrix character in to a single string
145.         char p1[] = new char[row * selectedKey.length()];
146.         k = 0;
147.         for (i = 0; i < row; i++)
148.         {
149.             for (j = 0; j < selectedKey.length(); j++)
150.             {
151.                 if (pmat[i][j] != '*')
152.                 {
153.                     p1[k++] = pmat[i][j];
154.                 }
155.             }
156.         }
157.         p1[k++] = '\0';
158.         return (new String(p1));
159.     }
160.
161.     @SuppressWarnings("static-access")
162.     public static void main(String[] args)
163.     {
164.         TranspositionCipher tc = new TranspositionCipher();
165.         System.out.println("Encrypted Message is: "
166.             + tc.doEncryption("Sanfoundry"));
167.         System.out.println("Decrypted Message is: "
168.             + tc.doDecryption(tc.doEncryption("Sanfoundry")));
169.     }
170. }

```

Output:

```
$ javac TranspositionCipher.java
```

```
$ java TranspositionCipher
```

```
Encrypted Message is: f*o*n*ayn*d*Sru*
```

```
Decrypted Message is: Sanfoundry
```

The screenshot displays the IntelliJ IDEA IDE interface. The top toolbar includes standard development tools like Run, Debug, and Build. The left sidebar shows the project structure for 'Cryptography'. The main editor window contains the following Java code:

```

1  Scanner scan = new Scanner(System.in);
2  Scanner scan = new Scanner(System.in);
3  System.out.print("Enter r1 value : ");
4  int a = scan.nextInt();
5  System.out.print("Enter r2 value : ");
6  int b = scan.nextInt();
7  int s;
8  s = a + b;
9  System.out.println("Sum" + a + " + " + b + " = " + s);
10 }
11 }

```

The bottom output window shows the execution results:

```

Enter r1 value : 1700
Enter r2 value : 2940
Sum(1700 , 2940) = 4640
Process finished with exit code 0

```

Name – Nikhil Srivastava

Section – C (2)

Roll No. – 29

Q - Write a program to find out the Multiplicative inverse of a given number by using Extended Euclidean algorithm.

```
import java.util.Scanner;

public class FindMultiplicativeInverse {

    static int modInverse(int A, int M)

    {

        for (int X = 1; X < M; X++)

            if (((A % M) * (X % M)) % M == 1)

                return X;

        return 1;

    }

    public static void main(String[] args) {

        Scanner scan = new Scanner(System.in);

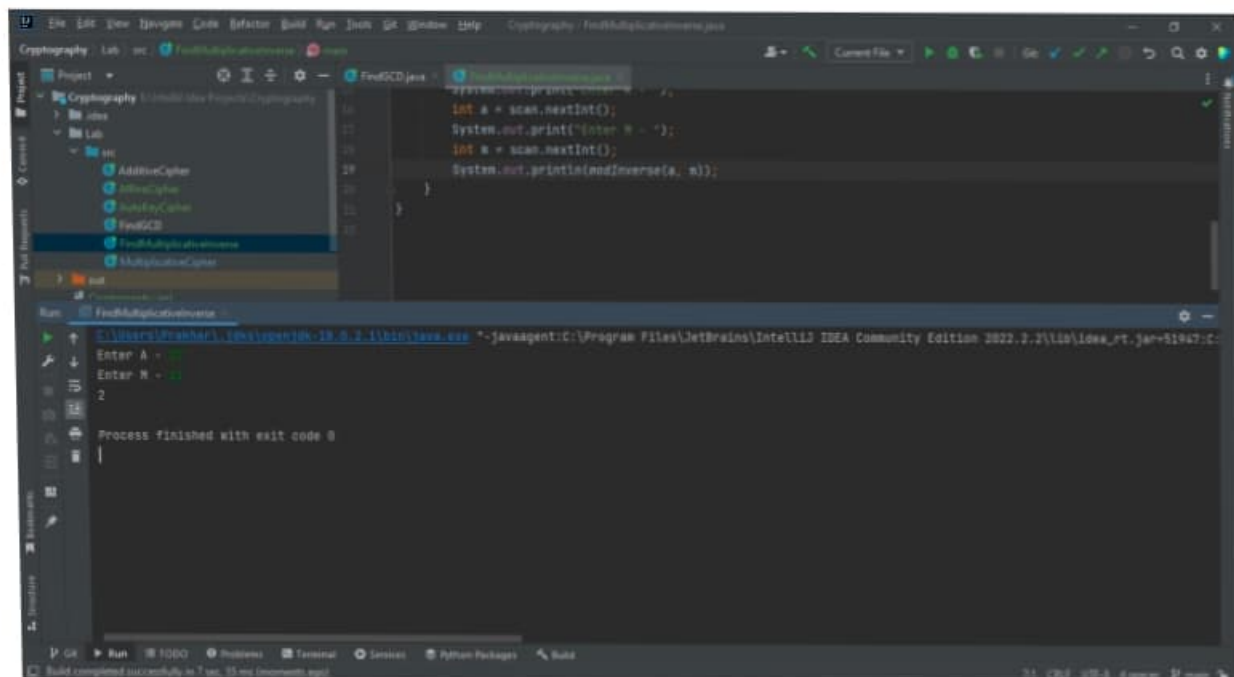
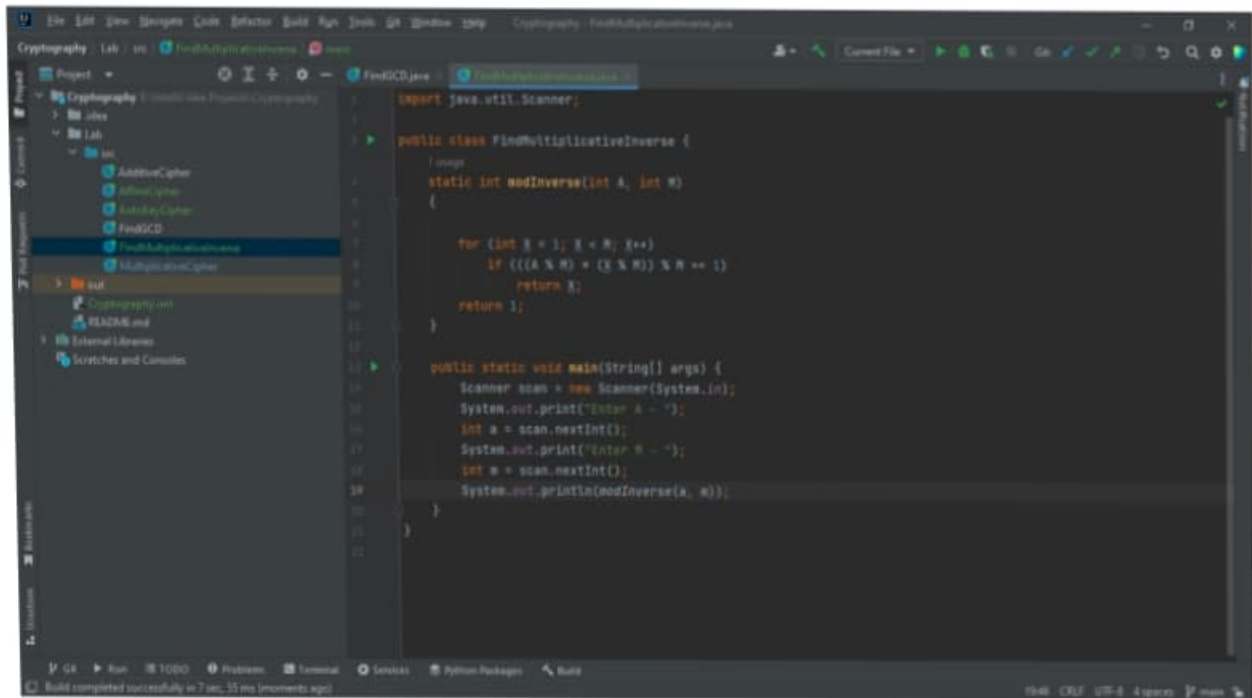
        System.out.print("Enter A - ");

        int a = scan.nextInt();

        System.out.print("Enter M - ");

        int m = scan.nextInt();

        System.out.println(modInverse(a, m) )}
```



Name-Nikhil Srivastava

Sec-c

U roll num—191500505

```
import java.security.Key;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.SecureRandom;
import java.security.Security;

import javax.crypto.Cipher;

public class MainClass {
    public static void main(String[] args) throws Exception {
        Security.addProvider(new org.bouncycastle.jce.provider.BouncyCastleProvider());

        byte[] input = "ab".getBytes();
        Cipher cipher = Cipher.getInstance("ElGamal/None/NoPadding", "BC");
        KeyPairGenerator generator = KeyPairGenerator.getInstance("ElGamal", "BC");
        SecureRandom random = new SecureRandom();

        generator.initialize(128, random);

        KeyPair pair = generator.generateKeyPair();
        Key pubKey = pair.getPublic();
        Key privKey = pair.getPrivate();
        cipher.init(Cipher.ENCRYPT_MODE, pubKey, random);
        byte[] cipherText = cipher.doFinal(input);
        System.out.println("cipher: " + new String(cipherText));

        cipher.init(Cipher.DECRYPT_MODE, privKey);
```

```
byte[] plainText = cipher.doFinal(cipherText);  
System.out.println("plain : " + new String(plainText));  
}  
}
```

Name-Nikhil Srivastava

Sec-c

U roll num—191500505

The screenshot displays the OnlineGDB IDE interface. The left sidebar contains navigation links: OnlineGDB beta, code.compile.run.debug.share., IDE, My Projects, Classroom, Learn Programming, Programming Questions, Sign Up, and Login. The main editor area shows a Java file named 'Main.java' with the following code:

```
17- import java.math.*;
18- import java.util.*;
19-
20- public class Main {
21-     public static void main(String args[])
22-     {
23-         int p, q, n, z, d = 0, e, i;
24-
25-         // The number to be encrypted and decrypted
26-         int msg = 12;
27-         double c;
28-         BigInteger msgback;
29-
30-         // 1st prime number p
31-         p = 3;
32-
33-         // 2nd prime number q
34-         q = 11;
35-         n = p * q;
36-         z = (p - 1) * (q - 1);
37-         System.out.println("the value of z = " + z);
38-
39-         for (e = 2; e < z; e++) {
40-
41-             // e is for public key exponent
42-             if (gcd(e, z) == 1) {
43-                 break;
44-             }
45-         }
46-         System.out.println("the value of e = " + e);
```

The console output shows the results of the program execution:

```
the value of z = 20
the value of e = 3
the value of d = 7
Encrypted message is : 12.0
Decrypted message is : 12
...Program finished with exit code 0
Press ENTER to exit console.
```

The bottom screenshot shows the continuation of the Java code in 'Main.java':

```
46- System.out.println("the value of e = " + e);
47- for (i = 0; i <= 9; i++) {
48-     int x = 1 + (i * z);
49-
50-     // d is for private key exponent
51-     if (x % e == 0) {
52-         d = x / e;
53-         break;
54-     }
55- }
56- System.out.println("the value of d = " + d);
57- c = (Math.pow(msg, e)) % n;
58- System.out.println("Encrypted message is : " + c);
59-
60- // converting int value of n to BigInteger
61- BigInteger N = BigInteger.valueOf(n);
62-
63- // converting float value of c to BigInteger
64- BigInteger C = BigDecimal.valueOf(c).toBigInteger();
65- msgback = (C.pow(d)).mod(N);
66- System.out.println("Decrypted message is : "
67-     + msgback);
68- }
69-
70- static int gcd(int e, int z)
71- {
72-     if (e == 0)
73-         return z;
74-     else
75-         return gcd(z % e, e);
76- }
```

The console output for the second screenshot is identical to the first one, showing the same results for z, e, d, and the encrypted/decrypted messages.

Student Corner Miller Rabin Primality Test in Java - Sa Primality Test | Set 3 (Miller-Rabin) - G Firewall Authentication Keepalive Win Online Java Compiler - online edi x

onlinegdb.com/online_java_compiler

OnlineGDB beta
online compiler and debugger for c/c++
code compile run debug share.

IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login

Main.java

```
19- import java.util.Scanner;
20- import java.util.Random;
21- import java.math.BigInteger;
22-
23- /** Class MillerRabin */
24- public class Main
25- {
26-     /** Function to check if prime or not */
27-     public boolean isPrime(long n, int iteration)
28-     {
29-         /** base case */
30-         if (n == 0 || n == 1)
31-             return false;
32-         /** base case - 2 is prime */
33-         if (n == 2)
34-             return true;
35-         /** an even number other than 2 is composite */
36-         if (n % 2 == 0)
37-             return false;
38-
39-         long s = n - 1;
40-         while (s % 2 == 0)
41-             s /= 2;
42-
43-         Random rand = new Random();
44-         for (int i = 0; i < iteration; i++)
45-         {
46-             long r = Math.abs(rand.nextLong());
47-             long a = r % (n - 1) + 1, temp = s;
48-             long mod = modPow(a, temp, n);
```

Input

Enter number of iterations
2
5510389 is prime

...Program finished with exit code 0
Press ENTER to exit console.

About • FAQ • Blog • Terms of Use • Contact Us • GDB
Tutorial • Credits • Privacy
© 2016 - 2022 GDB Online

Student Corner Miller Rabin Primality Test in Java - Sa Primality Test | Set 3 (Miller-Rabin) - G Firewall Authentication Keepalive Win Online Java Compiler - online edi x

onlinegdb.com/online_java_compiler

OnlineGDB beta
online compiler and debugger for c/c++
code compile run debug share.

IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login

Main.java

```
48-         long mod = modPow(a, temp, n);
49-         while (temp != n - 1 && mod != 1 && mod != n - 1)
50-         {
51-             mod = mulMod(mod, mod, n);
52-             temp *= 2;
53-         }
54-         if (mod != n - 1 && temp % 2 == 0)
55-             return false;
56-         return true;
57-     }
58-     /** Function to calculate (a ^ b) % c */
59-     public long modPow(long a, long b, long c)
60-     {
61-         long res = 1;
62-         for (int i = 0; i < b; i++)
63-         {
64-             res *= a;
65-             res %= c;
66-         }
67-         return res % c;
68-     }
69-     /** Function to calculate (a * b) % c */
70-     public long mulMod(long a, long b, long mod)
71-     {
72-         return BigInteger.valueOf(a).multiply(BigInteger.valueOf(b)).mod(BigInteger.valueOf(mod)).longValue();
73-     }
74-     /** Main function */
75-     public static void main (String[] args)
76-     {
77-     }
```

Input

Enter number of iterations
2
5510389 is prime

...Program finished with exit code 0
Press ENTER to exit console.

About • FAQ • Blog • Terms of Use • Contact Us • GDB
Tutorial • Credits • Privacy
© 2016 - 2022 GDB Online

Student Corner

Miller Rabin Primality Test in Java - Sa

Primality Test | Set 3 (Miller-Rabin) - G

Firewall Authentication Keepsalive Win

Online Java Compiler - online edi

onlinegdb.com/online_java_compiler

Language Java

OnlineGDB beta
online compiler and debugger for c/c++
code compile run debug share
IDE
My Projects
Classroom **new**
Learn Programming
Programming Questions
Sign Up
Login

68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97

```
        return res % c;
    }
    /** Function to calculate (a * b) % c */
    public long mulMod(long a, long b, long mod)
    {
        return BigInteger.valueOf(a).multiply(BigInteger.valueOf(b)).mod(BigInteger.valueOf(mod)).longValue();
    }
    /** Main function */
    public static void main (String[] args)
    {
        Scanner scan = new Scanner(System.in);
        System.out.println("Miller Rabin Primality Algorithm Test\n");
        /** Make an object of MillerRabin class */
        Main mr = new Main();
        /** Accept number */
        System.out.println("Enter number\n");
        long num = scan.nextLong();
        /** Accept number of iterations */
        System.out.println("\nEnter number of iterations");
        int k = scan.nextInt();
        /** check if prime */
        boolean prime = mr.isPrime(num, k);
        if (prime)
            System.out.println("\n"+ num + " is prime");
        else
            System.out.println("\n"+ num + " is composite");
    }
}
```

Input

Miller Rabin Primality Algorithm Test
Enter number
5510389
Enter number of iterations
2

About • FAQ • Blog • Terms of Use • Contact Us • GDB
Tutorial • Credits • Privacy
© 2016 - 2022 GDB Online

Name-Nikhil Srivastava

Sec-c

U roll num—191500505

The screenshot displays the OnlineGDB Java compiler interface. The left sidebar contains navigation links: OnlineGDB beta, code compile run debug share, IDE, My Projects, Classroom, Learn Programming, Programming Questions, Sign Up, and Login. The main editor shows a Java file named 'Main.java' with the following code:

```
19 import java.util.*;
20 // create class DiffieHellmanAlgorithmExample to calculate the key for two persons
21 class Main {
22     // main() method start
23     public static void main(String[] args)
24     {
25         long P, G, X, A, Y, B, KA, KB;
26         // create Scanner class object to take input from user
27         Scanner sc = new Scanner(System.in);
28         System.out.println("Both the users should be agreed upon the public keys G and P");
29         // take inputs for public keys from the user
30         System.out.println("Enter value for public key G:");
31         G = sc.nextLong();
32         System.out.println("Enter value for public key P:");
33         P = sc.nextLong();
34         // get input from user for private keys a and b selected by User1 and User2
35         System.out.println("Enter value for private key a selected by user1:");
36         A = sc.nextLong();
37         System.out.println("Enter value for private key b selected by user2:");
38         B = sc.nextLong();
39
40         // call calculatePower() method to generate x and y keys
41         X = calculatePower(G, A, P);
42         Y = calculatePower(G, B, P);
43         // call calculatePower() method to generate ka and kb secret keys after the exchange of x and y keys
44         // calculate secret key for User1
45         KA = calculatePower(Y, A, P);
46         // calculate secret key for User2
47         KB = calculatePower(X, B, P);
48
49         // print secret keys of user1 and user2
50         System.out.println("Secret key for User1 is: " + KA);
51         System.out.println("Secret key for User2 is: " + KB);
52     }
53     // create calculatePower() method to find the value of x ^ y mod P
54     private static long calculatePower(long x, long y, long P)
55     {
56         long result = 0;
57         if (y == 1){
58             return x;
59         }
60         else{
61             result = ((long)Math.pow(x, y)) % P;
62             return result;
63         }
64     }
65 }
```

The console output shows the program execution:

```
Enter value for private key b selected by user2:
2
Secret key for User1 is:25
Secret key for User2 is:25
...Program finished with exit code 0
Press ENTER to exit console.
```

The second screenshot shows the same code with the console output:

```
Both the users should be agreed upon the public keys G and P
Enter value for public key G:
8
Enter value for public key P:
33
Enter value for private key a selected by user1:
33
Enter value for private key b selected by user2:
2
```

Name-Nikhil Srivastava

Sec-C

U.roll no. 191500505

The image shows two screenshots of the OnlineGDB Java compiler interface. The top screenshot displays the initial code for an RSA encryption program. The bottom screenshot shows the same code with additional decryption logic and the final output.

Top Screenshot Code:

```
17 import java.math.*;
18 import java.util.*;
19
20 public class Main {
21     public static void main(String args[])
22     {
23         int p, q, n, z, d = 0, e, i;
24
25         // The number to be encrypted and decrypted
26         int msg = 12;
27         double c;
28         BigInteger msgback;
29
30         // 1st prime number p
31         p = 3;
32
33         // 2nd prime number q
34         q = 11;
35         n = p * q;
36         z = (p - 1) * (q - 1);
37         System.out.println("the value of z = " + z);
38
39         for (e = 2; e < z; e++) {
40
41             // e is for public key exponent
42             if (gcd(e, z) == 1) {
43                 break;
44             }
45         }
46         System.out.println("the value of e = " + e);
```

Top Screenshot Output:

```
the value of z = 20
the value of e = 3
the value of d = 7
Encrypted message is : 12.0
Decrypted message is : 12
...Program finished with exit code 0
Press ENTER to exit console.
```

Bottom Screenshot Code:

```
46 System.out.println("the value of e = " + e);
47 for (i = 0; i <= 9; i++) {
48     int x = 1 + (i * z);
49
50     // d is for private key exponent
51     if (x % e == 0) {
52         d = x / e;
53         break;
54     }
55 }
56 System.out.println("the value of d = " + d);
57 c = (Math.pow(msg, e)) % n;
58 System.out.println("Encrypted message is : " + c);
59
60 // converting int value of n to BigInteger
61 BigInteger N = BigInteger.valueOf(n);
62
63 // converting float value of c to BigInteger
64 BigInteger C = BigDecimal.valueOf(c).toBigInteger();
65 msgback = (C.pow(d)).mod(N);
66 System.out.println("Decrypted message is : "
67     + msgback);
68 }
69
70 static int gcd(int e, int z)
71 {
72     if (e == 0)
73         return z;
74     else
75         return gcd(z % e, e);
76 }
```

Bottom Screenshot Output:

```
the value of z = 20
the value of e = 3
the value of d = 7
Encrypted message is : 12.0
Decrypted message is : 12
...Program finished with exit code 0
Press ENTER to exit console.
```

Name-Nikhil Srivastava

Sec-c

U roll num—191500505

```
import java.security.Key;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.SecureRandom;
import java.security.Security;

import javax.crypto.Cipher;

public class MainClass {
    public static void main(String[] args) throws Exception {
        Security.addProvider(new org.bouncycastle.jce.provider.BouncyCastleProvider());

        byte[] input = "ab".getBytes();
        Cipher cipher = Cipher.getInstance("ElGamal/None/NoPadding", "BC");
        KeyPairGenerator generator = KeyPairGenerator.getInstance("ElGamal", "BC");
        SecureRandom random = new SecureRandom();

        generator.initialize(128, random);

        KeyPair pair = generator.generateKeyPair();
        Key pubKey = pair.getPublic();
        Key privKey = pair.getPrivate();
        cipher.init(Cipher.ENCRYPT_MODE, pubKey, random);
        byte[] cipherText = cipher.doFinal(input);
        System.out.println("cipher: " + new String(cipherText));

        cipher.init(Cipher.DECRYPT_MODE, privKey);
```



```
byte[] plainText = cipher.doFinal(cipherText);  
System.out.println("plain : " + new String(plainText));  
}  
}
```