```
// Implementing Playfair Cipher

import java.util.*;

class prac2
{
    static char[][] matrix = new char[5][5];
    static char[] alphabets = new char[25];
    static List<Character> plaintext = new ArrayList<Character>();
    static List<Character> encrypted = new ArrayList<Character>();
    static List<Character> decrypted = new ArrayList<Character>();

    static char a,b;

    static void createAlphabetsArray()
    {
        //storing all the alphabets in an array
        int incr1 = 0;
        for(char c = 'a'; c <= 'z'; c++)
        {
            if (c!='j')
            {
                alphabets[incr1] = c;
                incr1++;
            }
        }
    }

    static void initMatrix()
    {
        //initializing the matrix
        for(int i=0; i<5; i++)
        {
            for (int j=0; j<5; j++)
            {
                matrix[i][j] = '0';
            }
        }

    }

    static boolean inMatrix(char c)
    {
        int flag = 0;
        for (int j=0; j<5; j++)
        {
            for(int k=0; k<5; k++)
            {
                if (c == matrix[j][k])
                {
                    flag = flag + 1;
                }
            }
        }

        if ( flag == 0)
        {
            return true;
        }
    else
    {
        return false;
    }
}

static void getKeyMatrix (char[] keywordCharArray, char[] alphabets, int keywordLength)
{
    int h = 0, i = 0;

    for (int j=0; j<5; j++)
    {
        for(int k=0; k<5; k++)
        {
            if (i<keywordLength)
            {
                if (inMatrix(keywordCharArray[i]))
                {
                    matrix[j][k] = keywordCharArray[i];
                    i++;
                }
                else
                {
                    i++;
                    k--;
                }
            }
            else
            {
                if (inMatrix(alphabets[h]))
                {
                    matrix[j][k] = alphabets[h];
                    h++;
                }
                else
                {
                    h++;
                    k--;
                }

            }
        }
    }

}

static void displayMatrix()
{
    //displaying matrix
    System.out.println("\nKey Matrix::\n");
    for(int i=0; i<5; i++)
    {
        for (int j=0; j<5; j++)
        {
            System.out.print(" " + matrix[i][j] + " ");
        }
        System.out.print("\n");
    }

}
```

```
static void inputPlaintext()
{
    Scanner scan = new Scanner(System.in);

    System.out.print("\nEnter word to encrypt (Can't contain 'j'):: ");
    String word = scan.nextLine();
    int wordLen = word.length();

    //substituting character into an array
    for(int i =0; i < wordLen; i++)
    {
        plaintext.add(i, word.charAt(i));
    }

    //making blocks of two
    for(int i =0; i < (plaintext.size() - 1); i+=2)
    {
        if(plaintext.get(i) == plaintext.get(i+1))
        {
            plaintext.add(i+1,'x');

        }
    }
    if(plaintext.size()%2 == 1)
    {
        plaintext.add('x'); //add x in the last if size of the text is odd
    }

    //displaying blocks
    System.out.print("Blocks to be encrypted:: ");
    for (int i =0; i < (plaintext.size() - 1); i+=2)
    {
        System.out.print(plaintext.get(i));
        System.out.print(plaintext.get(i+1) + " ");
    }
    System.out.print("\n");
}

static void encrypt()
{
    int row1 = 0, row2 = 0, col1 = 0, col2 = 0;
    System.out.print("Encrypted text:: ");
    for(int i = 0; i < plaintext.size(); i+=2)
    {
        a = plaintext.get(i);
        b = plaintext.get(i+1);

        for (int j = 0; j <5; j++)
        {
            for (int k =0; k<5; k++)
            {
                if (matrix[j][k] == a)
                {
                    row1 = j;
                    col1 = k;
                }
                if (matrix[j][k] == b)
                {
```

```
            row2 = j;
            col2 = k;
        }
    }
}

if (row1 == row2)
{
    if (col1 != 4)
    {
        a = matrix[row1][col1 + 1];
        if (col2 != 4)
        {
            b = matrix[row2][col2 + 1];
        }
        else
        {
            b = matrix[row2][col2 - 4];
        }
    }
    else
    {
        a = matrix[row1][col1 - 4];
        b = matrix[row2][col2 + 1];
    }
}

if (col1 == col2)
{
    if (row1 != 4)
    {
        a = matrix[row1 + 1][col1];
        if (row2 != 4)
        {
            b = matrix[row2 + 1][col2];
        }
        else
        {
            b = matrix[row2 - 4][col2];
        }
    }
    else
    {
        a = matrix[row1 - 4][col1];
        if (row1 == row2)
        {
            b = matrix[row2 -4][col2];

        }
        else
        {
            b = matrix[row2 - 1][col2];
        }
    }
}

if ( (row1 != row2) & (col1 != col2) )
{
    a = matrix[row1][col2];
    b = matrix[row2][col1];
```

```
        }

        encrypted.add(a);
        encrypted.add(b);
        System.out.print((char)a);
        System.out.print((char)b);
    }
    System.out.print("\n");
}

static void decrypt()
{
    char a,b;
    int row1 = 0, row2 = 0, col1 = 0, col2 = 0;
    System.out.print("Decrypted text:: ");
    for(int i = 0; i < encrypted.size(); i+=2)
    {
        a = encrypted.get(i);
        b = encrypted.get(i+1);

        for (int j = 0; j <5; j++)
        {
            for (int k =0; k<5; k++)
            {
                if (matrix[j][k] == a)
                {
                    row1 = j;
                    col1 = k;
                }
                if (matrix[j][k] == b)
                {
                    row2 = j;
                    col2 = k;
                }
            }
        }

        if (row1 == row2)
        {
            if (col1 != 0)
            {
                a = matrix[row1][col1 - 1];
                if (col2 != 0)
                {
                    b = matrix[row2][col2 - 1];
                }
                else
                {
                    b = matrix[row2][col2 + 4];
                }
            }
            else
            {
                a = matrix[row1][col1 + 4];
                b = matrix[row2][col2 - 1];
            }
        }

        if (col1 == col2)
        {
            if (row1 != 0)
            {
                a = matrix[row1 - 1][col1];
                if (row2 != 0)
                {
                    b = matrix[row2 - 1][col2];
                }
                else
                {
                    b = matrix[row2 + 4][col2];
                }
            }
            else
            {
                a = matrix[row1 + 4][col1];
                if(row1==row2)
                {
                    b = matrix[row2 + 4][col2];
                }
                else
                {
                    b = matrix[row2 + 1][col2];
                }
            }
        }

        if ( (row1 != row2) & (col1 != col2) )
        {
            a = matrix[row1][col2];
            b = matrix[row2][col1];
        }

        decrypted.add(a);
        decrypted.add(b);
        // System.out.print((char)a);
        // System.out.print((char)b);
    }

    //removing x from the plaintext
    for (int i = 0; i < (decrypted.size()); i++)
    {
        if (i%2==1)
        {
            if (decrypted.get(i) !='x')
            {
                System.out.print(decrypted.get(i));
            }

        }
        else
        {
            System.out.print(decrypted.get(i));
        }
    }
    System.out.print("\n");
}

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

```
        System.out.print("\nEnter a keyword (Can't contain 'j'):: ");
        String keyword = scan.nextLine();
        int kLength = keyword.length();
        char[] keyword_char_array = keyword.toCharArray();


        createAlphabetsArray();

        initMatrix();

        getKeyMatrix(keyword_char_array, alphabets, kLength);

        displayMatrix();

        inputPlaintext();

        encrypt();

        decrypt();

    }
}
```