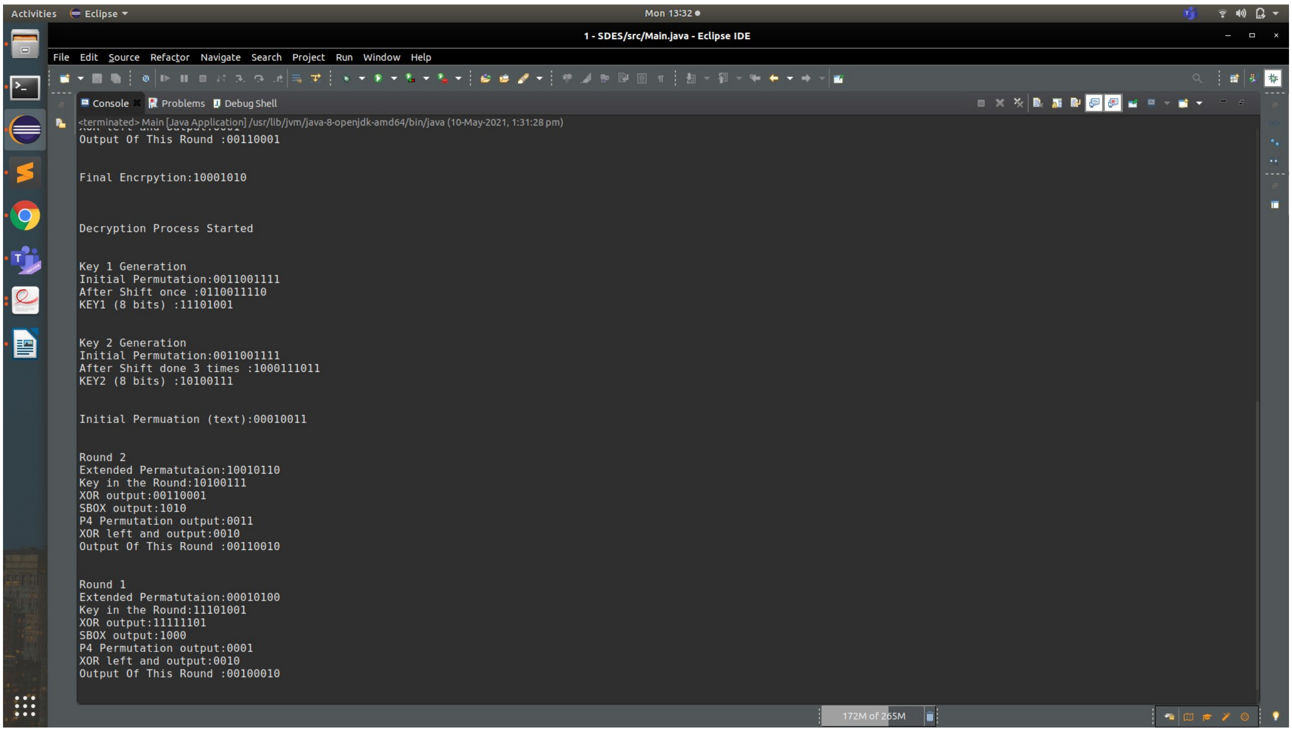
Program

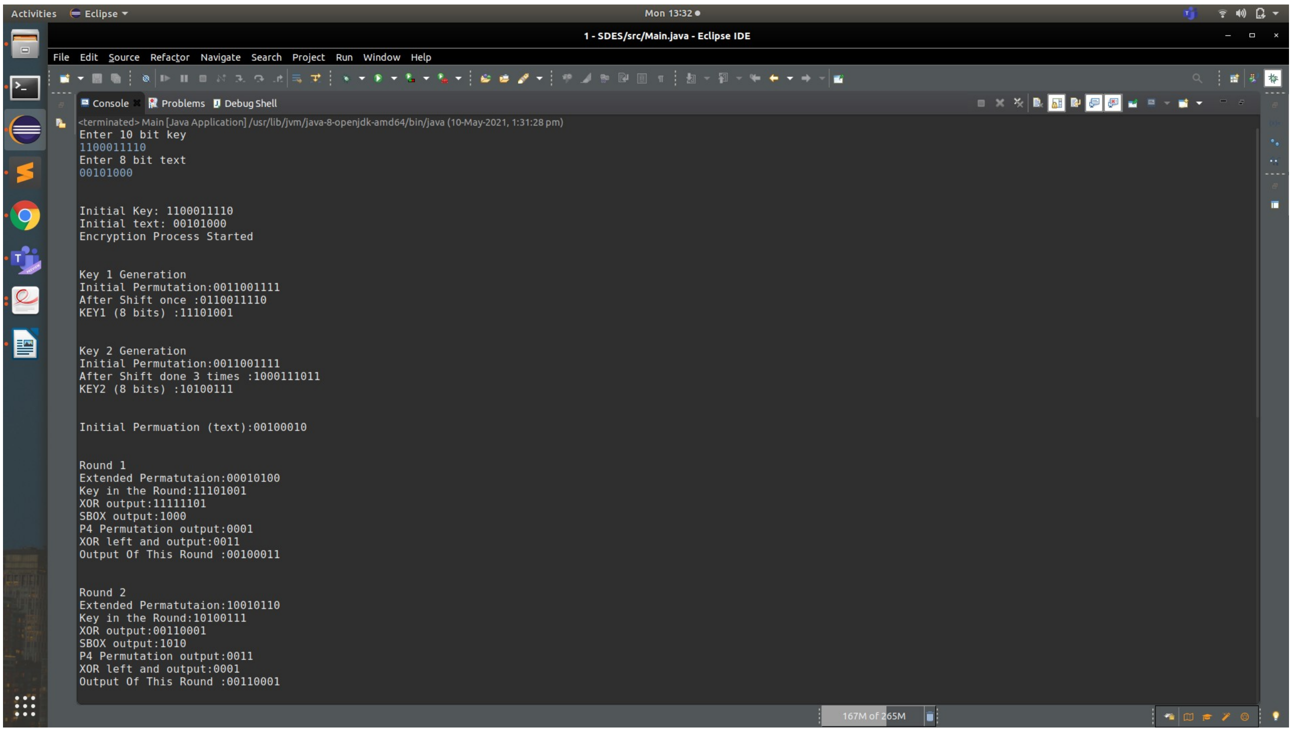
```
import java.util.*;
public class Main {
     static int key[];
     static int text[];
     static int K1[], K2[];
     static int P10[] = { 3, 5, 2, 7, 4, 10, 1, 9, 8, 6};
     static int P8[] = { 6, 3, 7, 4, 8, 5, 10, 9};
     static int P4[] = { 2, 4, 3, 1};
     static int IP[] = { 2, 6, 3, 1, 4, 8, 5, 7};
     static int IPI[] = { 4, 1, 3, 5, 7, 2, 8, 6};
     static int EP[] = { 4, 1, 2, 3, 2, 3, 4, 1};
     static int S0[][] = {{ 1, 0, 3, 2},{ 3, 2, 1, 0},{ 0, 2,
1,3},{ 3, 1, 3, 2}};
     static int S1[][] = {{ 0, 1, 2, 3},{ 2, 0, 1, 3},{ 3, 0,
1,2},{ 2, 1, 0, 3}};


     public static void display(int array[]) {
          for(int i=0;i<array.length;i++)
               System.out.print(array[i]);
          System.out.println();
     }


     static int[] permutation(int[] sequence, int input[]) {
     int output[] = new int[sequence.length];
        for (int i = 0; i < sequence.length; i++)
            output[i] = input[(sequence[i] - 1)];

        return output;
    }


     static int[] leftCircularShift(int input[], int numBits) {
     int len = input.length;
     while(numBits-- > 0) {

          int firstBit = input[0];

          for(int i=0;i<len-1;i++)
               input[i] = input[i+1];

          input[len-1] = firstBit;
     }
     return input;
    }


    static int[] xor(int[] input1, int input2[]) {
     int output[] = new int[input1.length];
        for (int i = 0; i < input1.length; i++)
            output[i] = input1[i] ^ input2[i];

        return output;
    }

    static int[] sBox(int input[]) {

     int output[] = new int[4];

        int left_part[]  = Arrays.copyOfRange(input, 0, 4);
        int right_part[] = Arrays.copyOfRange(input, 4, 8);

        // Doing for the Left PART
        int row = getDecimal(left_part[0], left_part[3]);
        int col = getDecimal(left_part[1], left_part[2]);
        int left_output = S0[row][col];
        int left_output_Part[] = getBinary(left_output);

        //Doing for Right Part
        row = getDecimal(right_part[0], right_part[3]);
        col = getDecimal(right_part[1], right_part[2]);
        int right_output = S1[row][col];
        int right_output_Part[] = getBinary(right_output);

        for(int i=0;i<2;i++)
          output[i] = left_output_Part[i];

        for(int i=0;i<2;i++)
          output[i+2] = right_output_Part[i];

     return output;
    }


    static int getDecimal(int bit1,int bit2) {
     int output = 0;
     output = bit1 * 2 + bit2;
     return output;
    }

    static int[] getBinary(int decimal) {
     int output[] = new int[2];
     String map[] = {"00","01","10","11"};
     for(int i=0;i<2;i++)
          output[i] = 
Integer.parseInt(String.valueOf(map[decimal].charAt(i)));
     return output;
    }

     public static void keyGeneration() {
          K1 = new int[8];
          K2 = new int[8];
          key1Generation();
          key2Generation();
     }


     public static void key1Generation() {

          int P10_1_OUT[] = permutation(P10,key);
          System.out.println("\n\nKey 1 Generation");

          System.out.print("Initial Permutation:");
          display(P10_1_OUT);

          int left_half[]  = Arrays.copyOfRange(P10_1_OUT, 0, 5);
          int right_half[] = Arrays.copyOfRange(P10_1_OUT, 5, 10);

          left_half = leftCircularShift(left_half,1);
          right_half = leftCircularShift(right_half,1);
          int combined[] = new int[10];

          for(int i=0;i<5;i++)
               combined[i] = left_half[i];

          for(int i=0;i<5;i++)
               combined[i+5] = right_half[i];


          System.out.print("After Shift once :" );
          display(combined);

          K1 = permutation(P8,combined);
          System.out.print("KEY1 (8 bits) :");
          display(K1);

     }

     public static void key2Generation() {

          int P10_1_OUT[] = permutation(P10,key);
          System.out.println("\n\nKey 2 Generation");

          System.out.print("Initial Permutation:");
          display(P10_1_OUT);


          int left_half[]  = Arrays.copyOfRange(P10_1_OUT, 0, 5);
          int right_half[] = Arrays.copyOfRange(P10_1_OUT, 5, 10);

          left_half = leftCircularShift(left_half,3);
          right_half = leftCircularShift(right_half,3);
```

```
          int combined[] = new int[10];

          for(int i=0;i<5;i++)
               combined[i] = left_half[i];

          for(int i=0;i<5;i++)
               combined[i+5] = right_half[i];


          System.out.print("After Shift done 3 times :" );
          display(combined);

          K2 = permutation(P8,combined);
          System.out.print("KEY2 (8 bits) :");
          display(K2);
     }

    static int[] round(int input[], int K[]) {

        int left[]  = Arrays.copyOfRange(input, 0, 4);
        int temp[] = Arrays.copyOfRange(input, 4, 8);
        int right[] = temp;

        temp = permutation(EP, temp);
        System.out.print("Extended Permatutaion:");
        display(temp);

        System.out.print("Key in the Round:");
        display(K);

        temp = xor(temp, K);
        System.out.print("XOR output:");
        display(temp);

        temp = sBox(temp);
        System.out.print("SBOX output:");
        display(temp);

        temp = permutation(P4, temp);
        System.out.print("P4 Permutation output:");
        display(temp);


        left = xor(left, temp);
        System.out.print("XOR left and output:");
        display(left);

        int output[] = new int[8];
        for(int i=0;i<4;i++)
          output[i] = right[i];
        for(int i=0;i<4;i++)
          output[i+4] = left[i];
        System.out.print("Output Of This Round :");
        display(output);
        return output;
    }


     static void encrypt() {

          System.out.println("Encryption Process Started");

          keyGeneration();
          int initial_key_permuation[] = permutation(IP, text);

          System.out.print("\n\nInitial Permuation (text):");
          display(initial_key_permuation);

          System.out.println("\n\nRound 1");
          text = round(initial_key_permuation, K1);


          System.out.println("\n\nRound 2");
          text = round(text, K2);

          //Final Permutation
          text = leftCircularShift(text, 4);
          text = permutation(IPI, text);
          System.out.print("\n\nFinal Encrpytion:");
          display(text);

     }


     static void decrypt() {

          System.out.println("\n\n\nDecryption Process Started");

          keyGeneration();
          int initial_key_permuation[] = permutation(IP, text);

          System.out.print("\n\nInitial Permuation (text):");
          display(initial_key_permuation);

          System.out.println("\n\nRound 2");
          text = round(initial_key_permuation, K2);


          System.out.println("\n\nRound 1");
          text = round(text, K1);

          //Final Permutation
          text = leftCircularShift(text, 4);
          text = permutation(IPI, text);
          System.out.print("\n\nFinal Decrpytion:");
          display(text);

     }

     @SuppressWarnings("static-access")
     public static void main(String[] args) {

          Scanner sc = new Scanner(System.in);
          key = new int[10];
          text = new int[8];

          System.out.println("Enter 10 bit key");
          String keyString = sc.next();
          for(int i=0;i<10;i++)
               key[i] =
Integer.parseInt(String.valueOf(keyString.charAt(i)));

          System.out.println("Enter 8 bit text");
          String textString = sc.next();
          for(int i=0;i<8;i++)
               text[i] =
Integer.parseInt(String.valueOf(textString.charAt(i)));

          System.out.print("\n\nInitial Key: ");
          display(key);
          System.out.print("Initial text: ");
          display(text);


          encrypt();
          decrypt();

          sc.close();
     }

}
```

Output:


Activities
Eclipse
Mon 13:32
1 - SDES/src/Main.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console
Problems
Debug Shell
172M of 265M

```
<terminated> Main [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (10-May-2021, 1:31:28 pm)
Output Of This Round :00110001


Final Encrpytion:10001010



Decryption Process Started


Key 1 Generation
Initial Permutation:0011001111
After Shift once :0110011110
KEY1 (8 bits) :11101001


Key 2 Generation
Initial Permutation:0011001111
After Shift done 3 times :1000111011
KEY2 (8 bits) :10100111


Initial Permuation (text):00010011


Round 2
Extended Permatutaion:10010110
Key in the Round:10100111
XOR output:00110001
SBOX output:1010
P4 Permutation output:0011
XOR left and output:0010
Output Of This Round :00110010


Round 1
Extended Permatutaion:00010100
Key in the Round:11101001
XOR output:11111101
SBOX output:1000
P4 Permutation output:0001
XOR left and output:0010
Output Of This Round :00100010
```


Activities
Eclipse
Mon 13:32
1 - SDES/src/Main.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console
Problems
Debug Shell
167M of 265M

```
<terminated> Main [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (10-May-2021, 1:31:28 pm)
Enter 10 bit key
1100011110
Enter 8 bit text
00101000


Initial Key: 1100011110
Initial text: 00101000
Encryption Process Started


Key 1 Generation
Initial Permutation:0011001111
After Shift once :0110011110
KEY1 (8 bits) :11101001


Key 2 Generation
Initial Permutation:0011001111
After Shift done 3 times :1000111011
KEY2 (8 bits) :10100111


Initial Permuation (text):00100010


Round 1
Extended Permatutaion:00010100
Key in the Round:11101001
XOR output:11111101
SBOX output:1000
P4 Permutation output:0001
XOR left and output:0011
Output Of This Round :00100011


Round 2
Extended Permatutaion:10010110
Key in the Round:10100111
XOR output:00110001
SBOX output:1010
P4 Permutation output:0011
XOR left and output:0001
Output Of This Round :00110001
```