College of Computer and Information Sciences

Department of Software Engineering

SWE 314: Software Security Engineering

1st Semester 1441-1442H

# Modern Block Cipher

# Project Report

| Section: 56740 | | |
|---|---|---|
| **#** | **Student Name** | **ID** |
| **1** | **Eman Aldosari** | **438201506** |
| **2** | **Fatimah Alrobaie** | **438204309** |
| **3** | **Deema Alekresh** | **437200093** |
| **4** | **Abeer Alharran** | **437201752** |
| **5** | **Alia Alsheha** | **437200766** |

# 1. Introduction

In this project, we design our block cipher and implement it in Java programming language. First, our block cipher is a keyless cryptographic algorithm that operates on a fixed-size block of data which will be in our block cipher 8 characters.

We restricted the input to be without any spaces and has a fixed size of 8 characters as we mentioned before, also in order to increase complexity, we decided to do two rounds for the input.

We chose 3 components to build the cipher block, starting with swap function (half swap) as the first component which involves about taking the user input and separate it from the middle into two parts (4 characters for each part) then join the two parts oppositely.

Second component is straight permutation box which will be taking the swapped result from the previous component and rearrange the characters by a predefined order. The third and the last component is circular left shift (3 characters), we will take the p-box resulted text and moving the first 3 characters together to the last position.

Keep in mind that when we want to decrypt, we will reverse the circular shift process by changing the shifting direction from left shift to right shift. After getting the resulted text from the right shift process, the text will be rearranging its characters to its original position, look at Figure 1 for clarity. Then the resulted text from the P-box decryption process the text will be swapped again (4 characters) and will represent the plaintext .

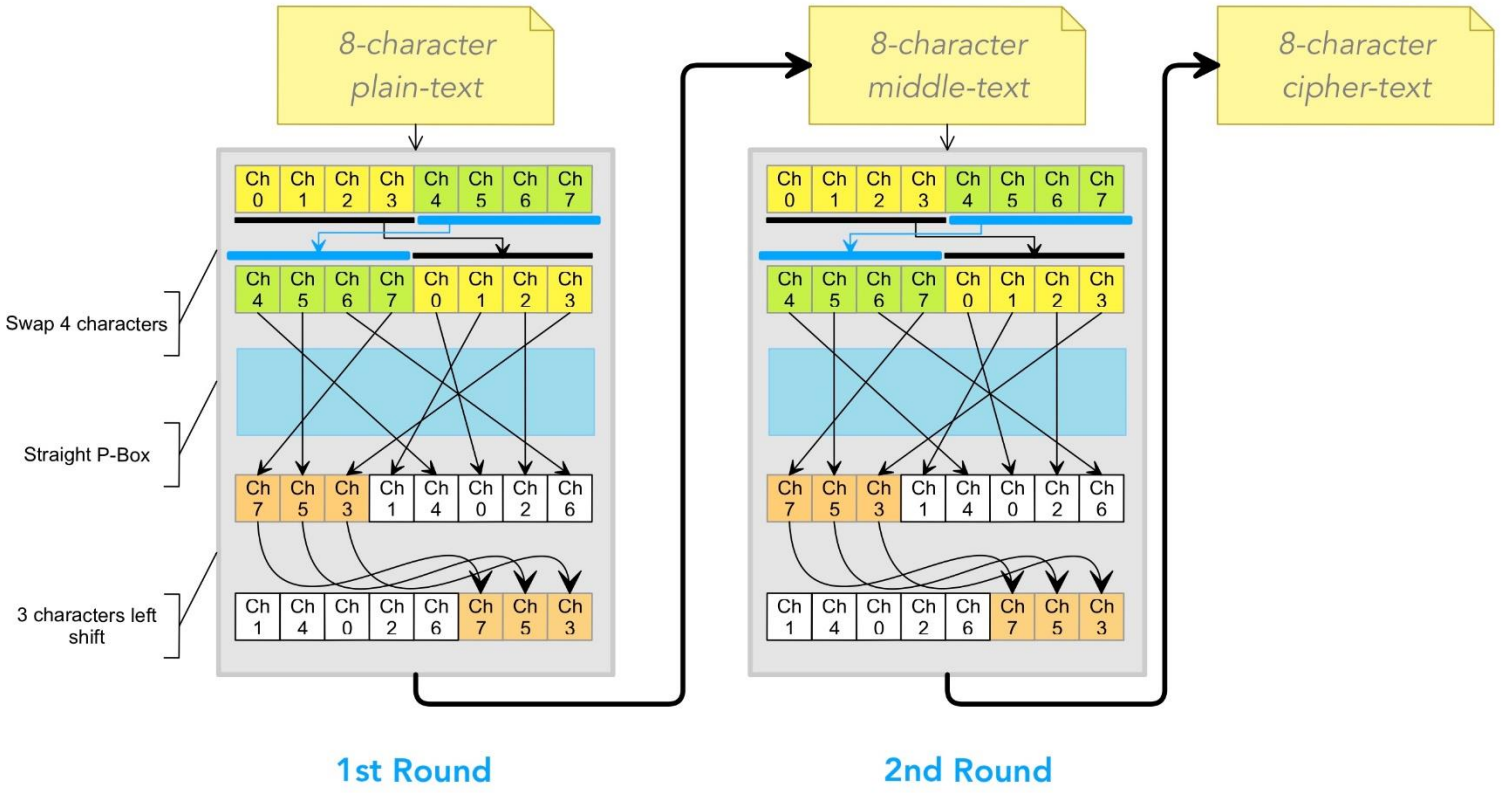



*Figure 1: Modern cipher design, encryption process*

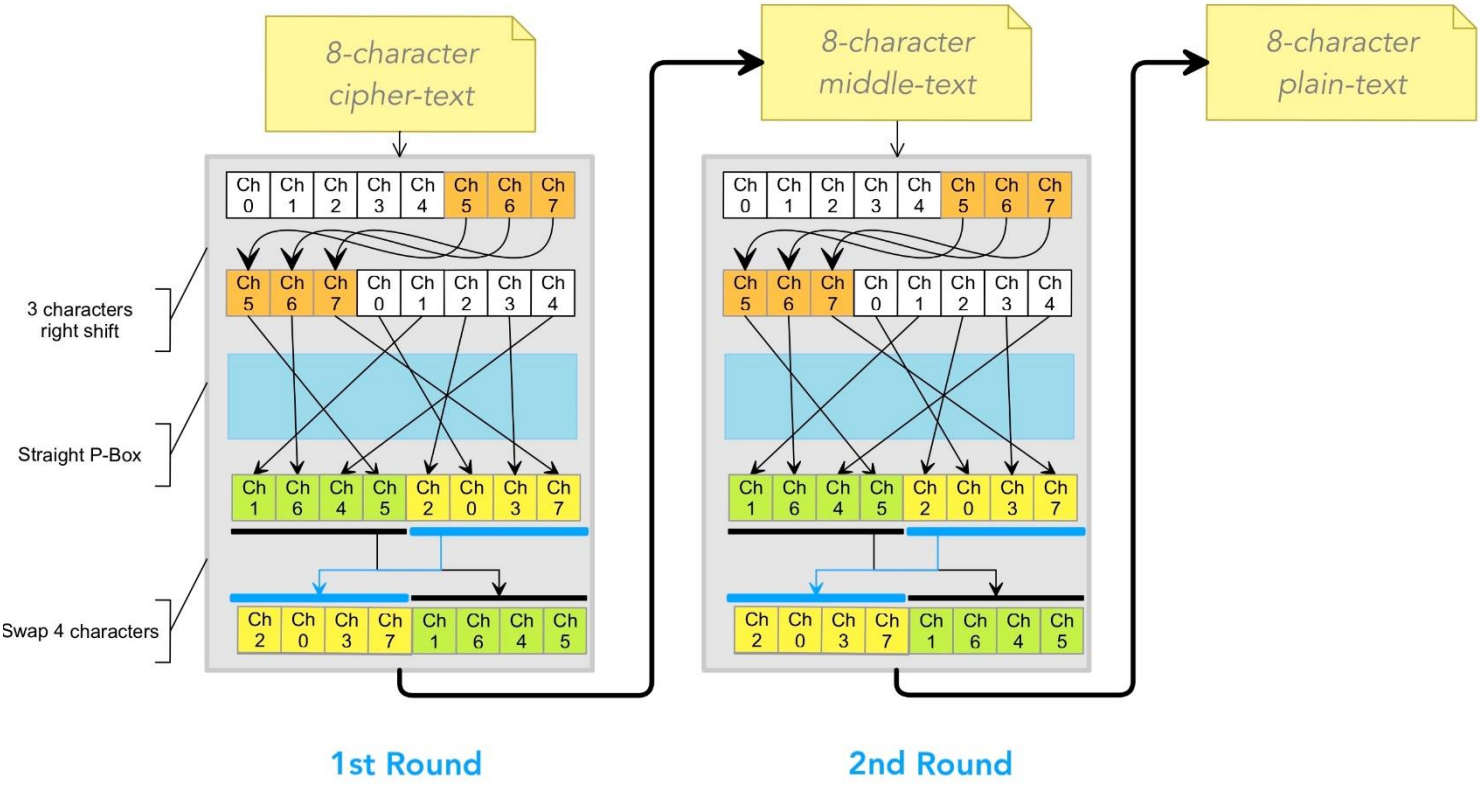



*Figure 2: Modern cipher design, decryption process*

## 2. Encryption source code

The encryption method processes the text by 3 parts :

- Swap function
- Straight permutation box function
- Circular left shift function

```
static String encryption(String plaintext){
        String ciphertext = "";
        for(int i=0;i<2;i++){
         char[] ch1 = plaintext.toCharArray();
         //swapping
         int mid = ( ch1.length + 1 ) / 2;
          for( int left = 0, right = mid; right < ch1.length; left++, right++ ) {
            char tmp = ch1[left];
            ch1[left] = ch1[right];
            ch1[right] = tmp;
          }
          String swappedString = new String(ch1);
          //p-boxing
            char [] result1 = new char [swappedString.length()];
            char [] copy = swappedString.toCharArray();
            result1[0] = copy [3];
            result1[1] = copy [1];
            result1[2] = copy [7];
            result1[3] = copy [5];
            result1[4] = copy [0];
            result1[5] = copy [4];
            result1[6] = copy [6];
            result1[7] = copy [2];
            String finaltext = new String(result1);
          //3 left shifts
            int k=3;
           ciphertext =  finaltext.substring(k) + finaltext.substring(0,k);

        plaintext = ciphertext; // update the plaintext to the middle text
         }
                    return ciphertext;
    }
```

## 3. Decryption source code

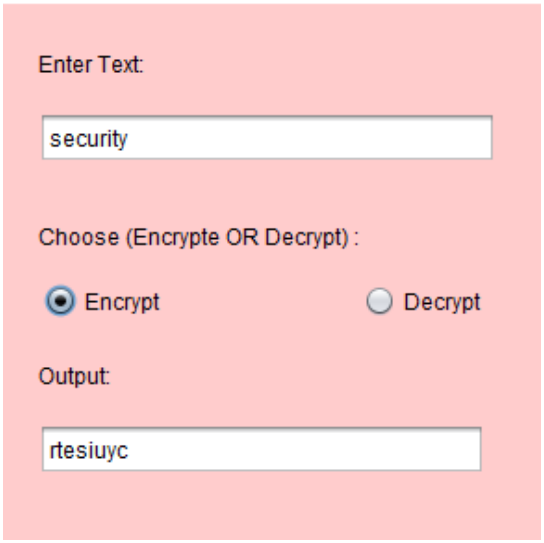
The decryption method processes the text by 3 parts:

- Circular right shift function
- Straight permutation box function
- Swap function

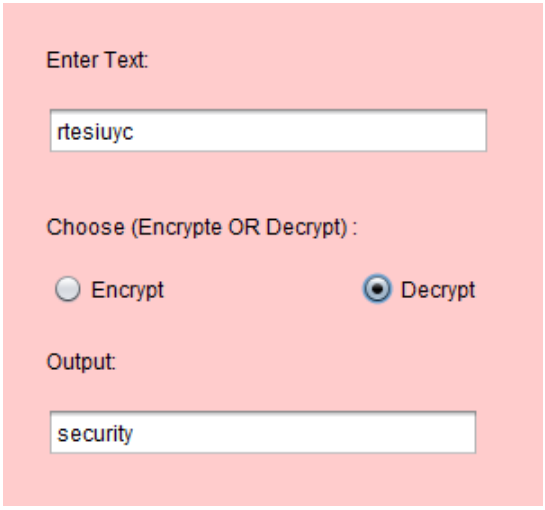
```
static String Decryption(String ciphertext){
    String plaintext ="";

       for(int j=0;j<2;j++){
            // 3 right shifts
             int n = 5;
            String rText= ciphertext.substring(n) + ciphertext.substring(0,n) ;
            //p-boxing
           char [] result1 = new char [rText.length()];
           char [] copy = rText.toCharArray();
           result1[0] = copy [4];
           result1[1] = copy [1];
           result1[2] = copy [7];
           result1[3] = copy [0];
           result1[4] = copy [5];
           result1[5] = copy [3];
           result1[6] = copy [6];
           result1[7] = copy [2];
           String resulttext = new String(result1);
         // swapping
         char[] ch1 = resulttext.toCharArray();
         int mid = ( ch1.length + 1 ) / 2; // or fixed length = 4 ?

          for( int left = 0, right = mid; right < ch1.length; left++, right++ ) {
            char tmp = ch1[left];
            ch1[left] = ch1[right];
            ch1[right] = tmp;
          }
           plaintext = new String(ch1);

        ciphertext = plaintext; //update the cipher text to the middle text
        }
       return plaintext;
}
```

# 4. Test cases

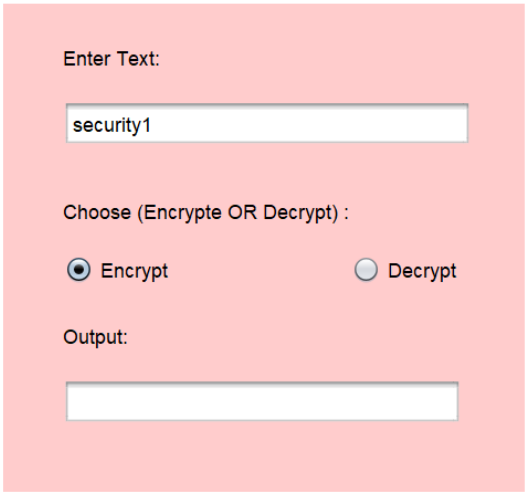
1- Enter 8 characters without spaces and choose Encrypt.

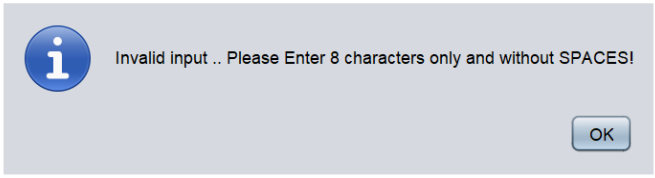



2- Enter 8 characters without spaces and choose Decrypt.

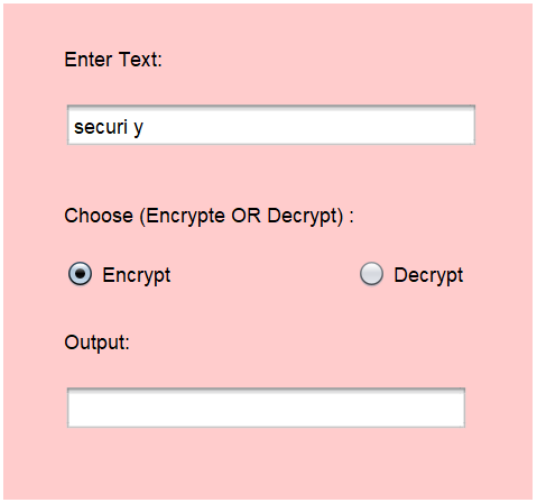


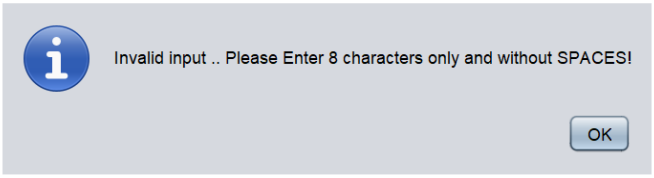
3- Enter more than 8 characters without spaces.



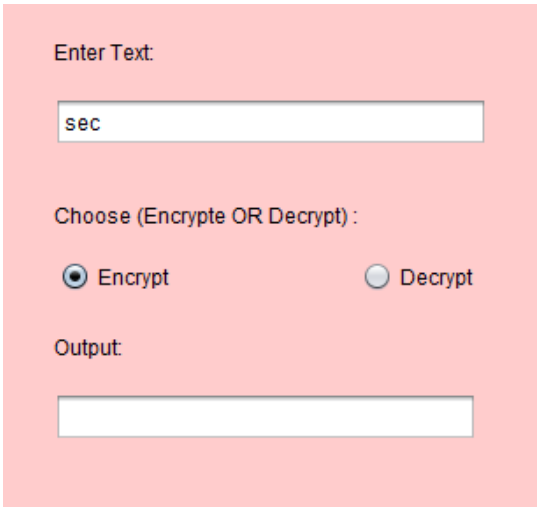



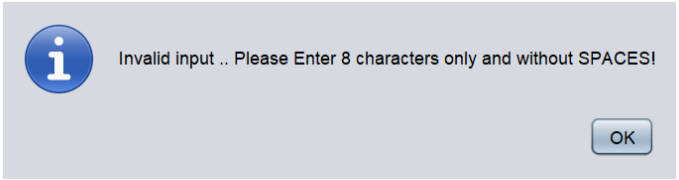

4- Enter 8 characters with spaces.




5- Enter less than 8 characters without spaces.



## 5. Conclusion

Learning and gaining knowledge in different aspects of software security is very important, this project has taught us to build our own block cipher in an excellent way by using java programming language which is fortunately all team members are familiar with it. We worked in this project together even though we are at different levels currently and it was kind of hard to find suitable hours for all of us to meet. This was the only issue, but this didn't affect our work on the project the result turned out as good as expected.

## ***References:***

[1] T. Richardson and C. N. Thies, Secure Software Design. Burlington, Mass: Jones & Bartlett Learning, 2013.