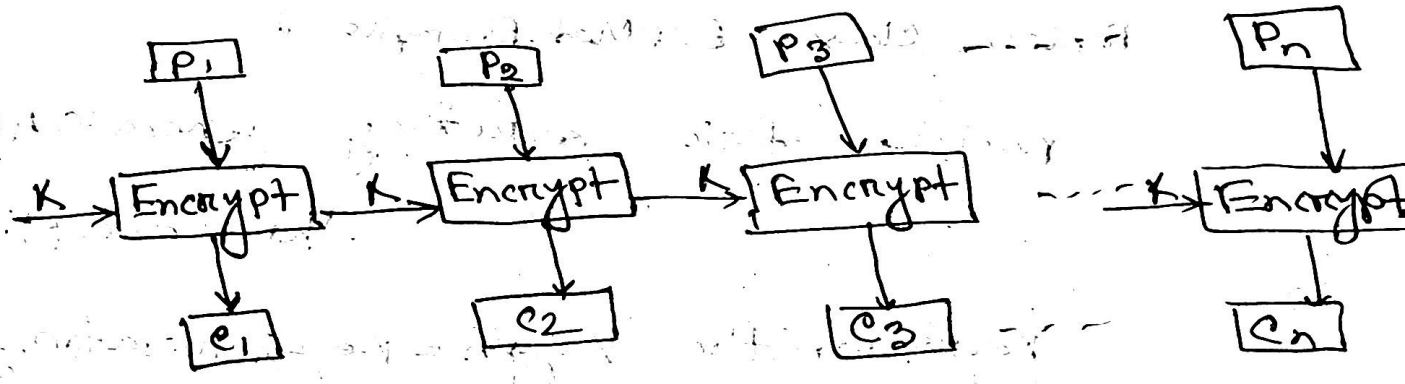
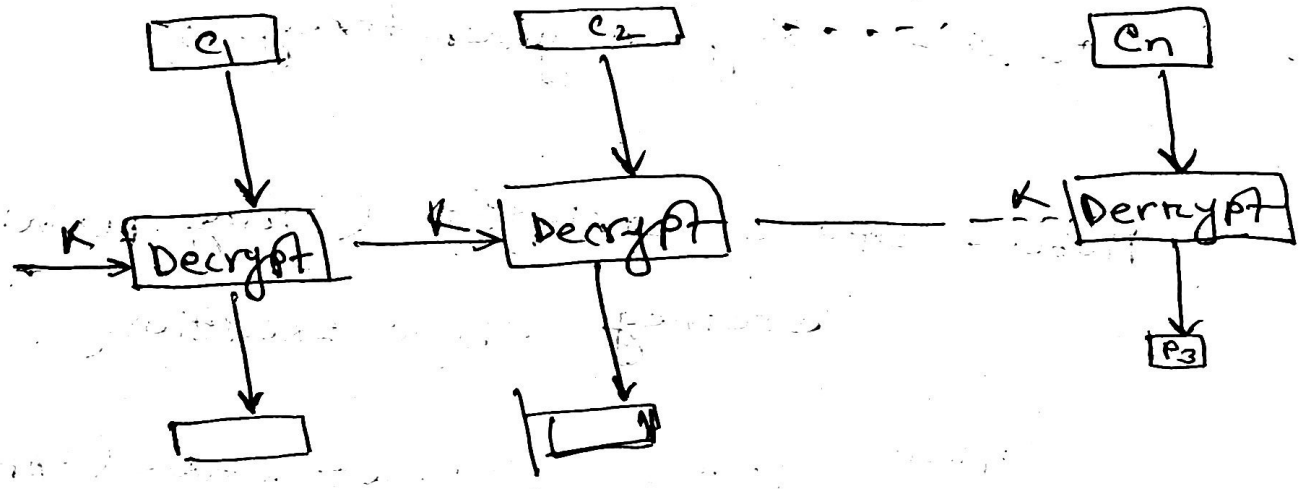
ECB (Electric codebook) : In an electric codebook each block of bits of plain text is encoded independently with the same Key.

Block Diagram :





Java implementation of ECB

```
import Javax.Crypto.Cipher;
import Javax.crypto.KeyGenerator;
Import Javax.crypto.secretKey;
import Javax.crypto.spec.secretKeyspace;
import Java.util.Base64;

Public class ECBModeExample2
    public static secretKey generateAESkey()
                                Throw exception{
    KeyGeneration Keygen = KeyGenerator.get
                        instance("AES");
        KeyGen.init(128);
    return Keygen.generatekey();
}
Public static encrypt (string plaintext,
        secretkey) Throw Exception{
    Cipher cipher = Cipher.getInstance("AES/ECB/
                        PKCS5Padding");
```

```
Cipher.init(cipher.ENCRYPT_Mode secret);
byte[] encryptedBytes = cipher.doFinal(plaintext.getbytes);
return Base64.getEncoder().encodeString(EncryptBytes);
}
// Decrypt cipher text using AES in ECB mode
public static String decrypt(String encryptedText,
                             SecretKey secretKey) Throws Exception{
Cipher cipher = Cipher.getInstance("AES/ECB/PCKS5");
    Cipher init (cipher.DECRYPT_NODE, secret key);
byte // decryptedBytes = cipher.doFinal(Base64.get
                                Decoder, decode(encryptedText));
    return new string (decrypted Bytes);
public static void main (String[] args) {
    try{ secret key = generateAESKey();
```

```
String original = "Hello cyber world";
String encrypted = encrypt(original, key);
String decrypted = decrypt(encrypted, key);
System.out.println("Original Text: " + original);
System.out.println("Encrypted text: " + encrypted);
System.out.println("Decrypted text: " + decrypted)

}
catch (Exception e) {
    e.printStackTrace();
}
}
}
```

output:

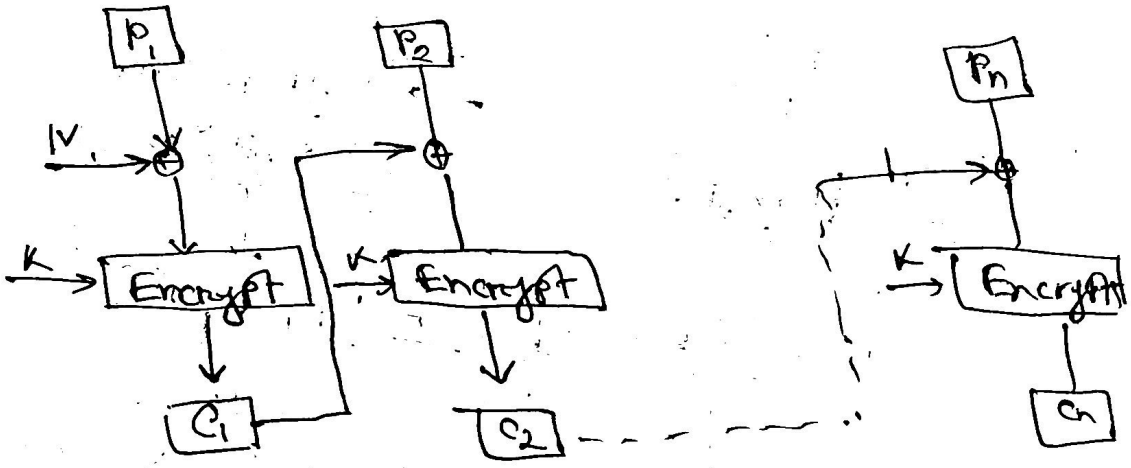
original Text : Hello cyber world !

Encrypted text: fRQ+*yKBp 3AnX IV4IQWPIQRWRgU

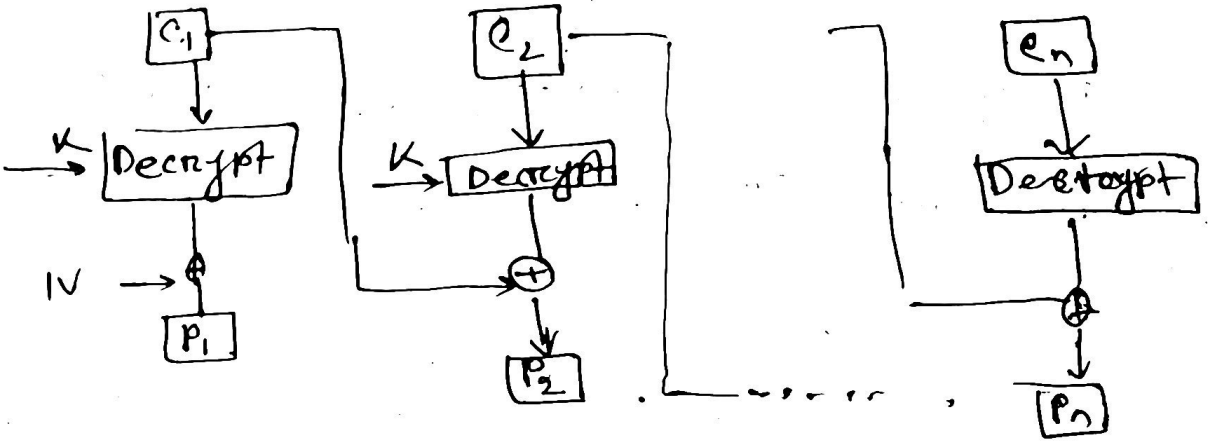
Decrypted text: Hello cyber world !

CBC (Cipher BLOCK chaining)

Encryption :

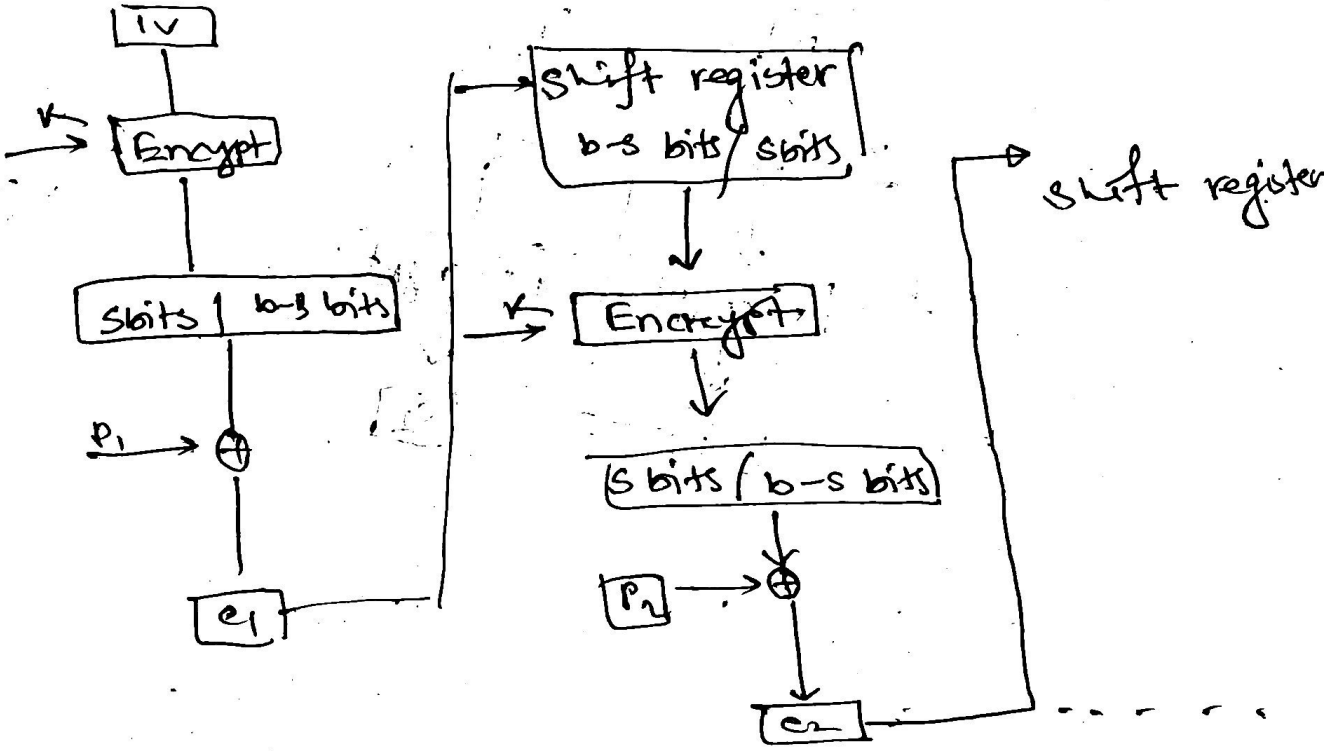


Decryption :

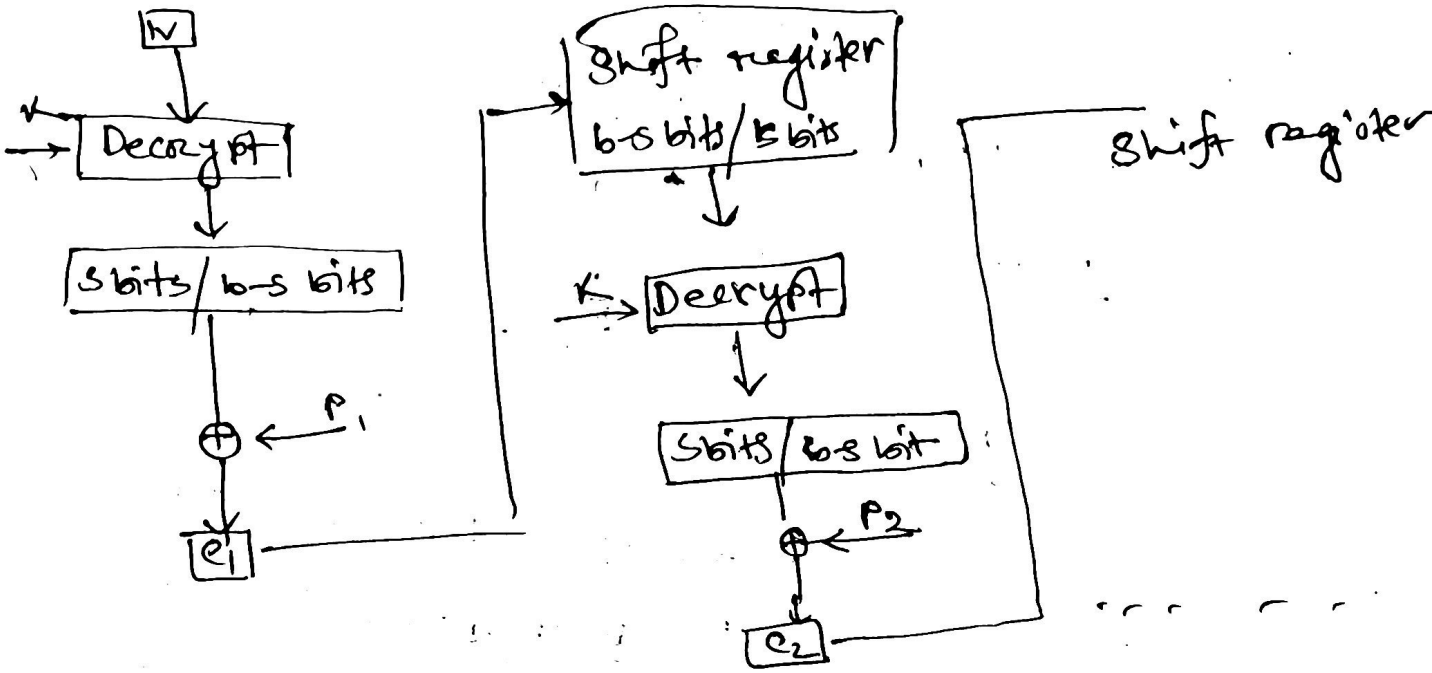


Cipher Feedback Mode (CFB)

Encryption

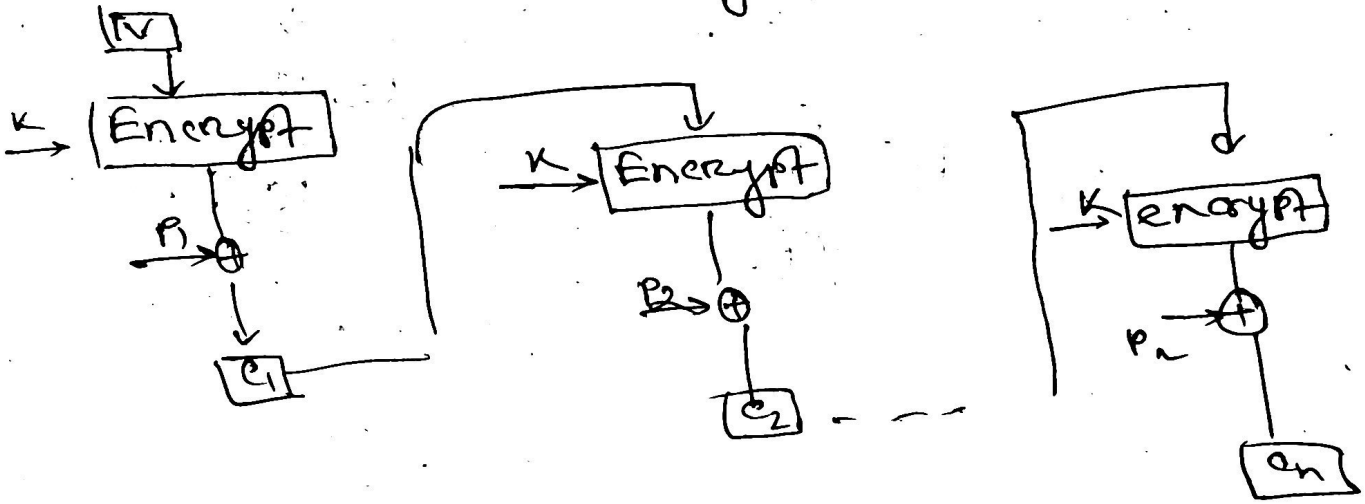


Decryption



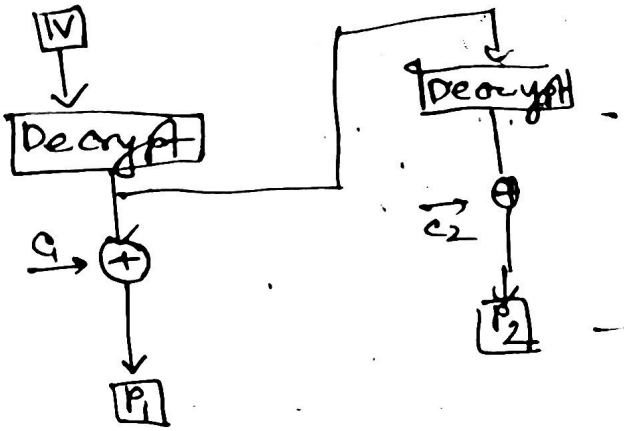
Output feedback (OFB)

Encryption



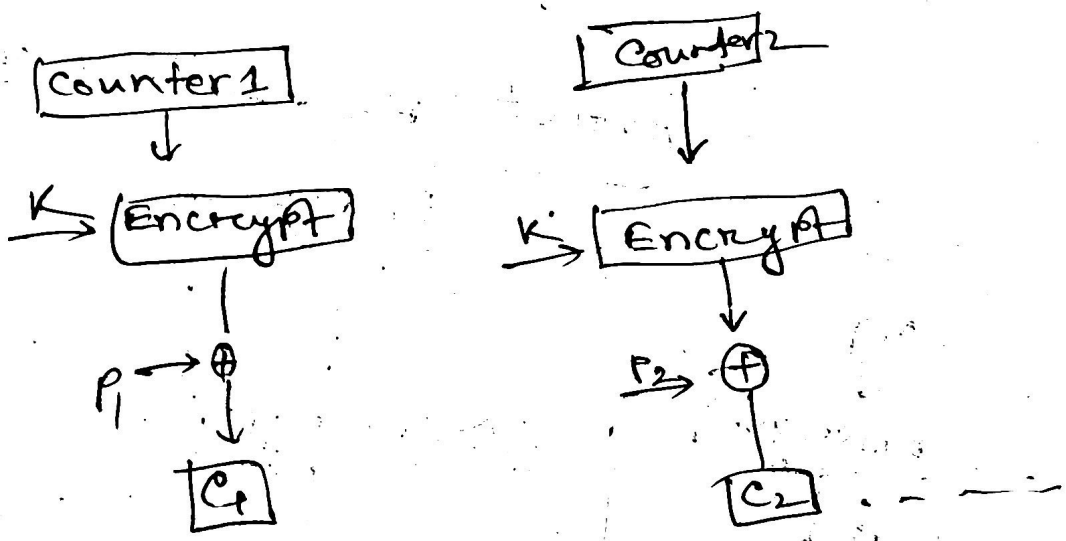
~~ECB~~

Decryption

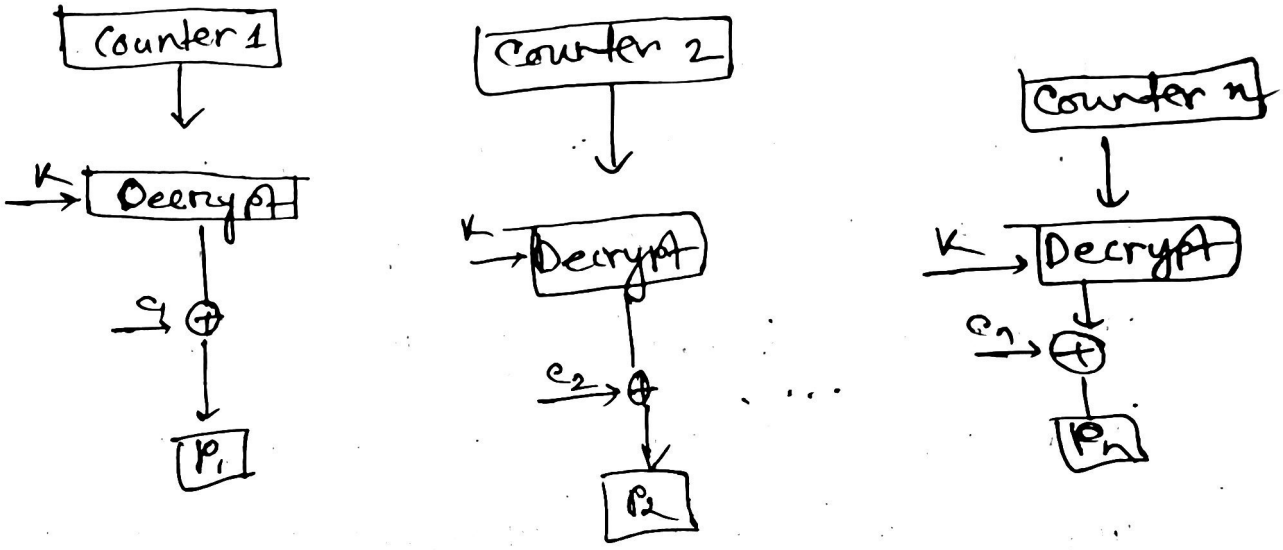


Counter Mode (CTR)

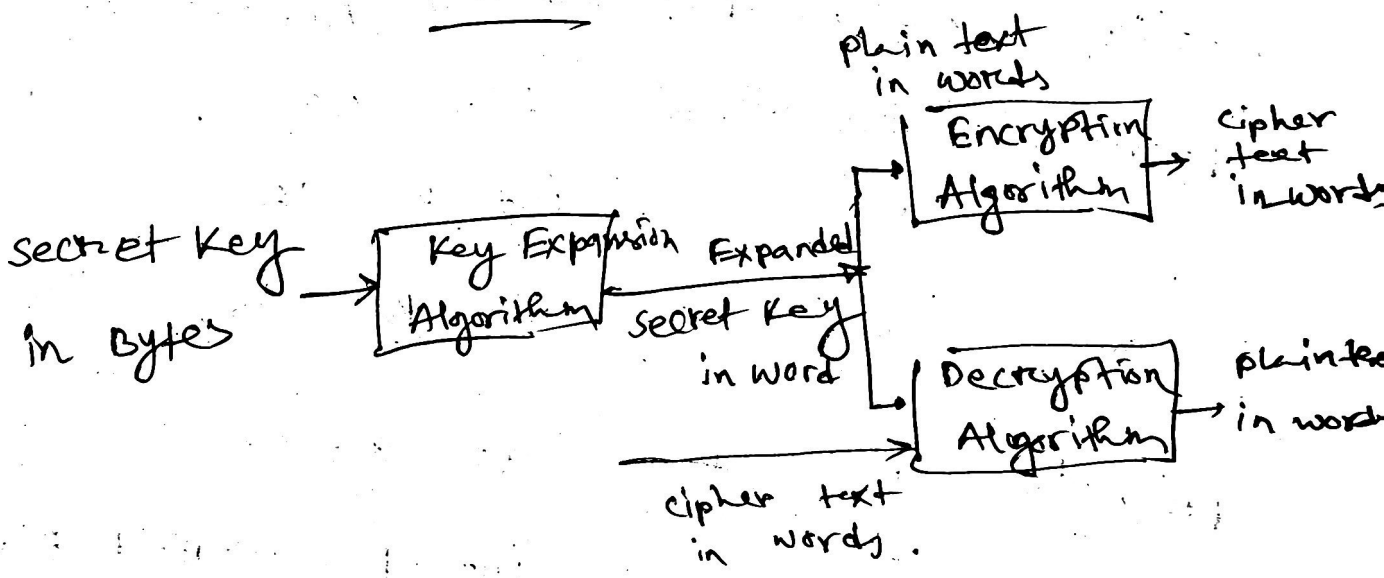
Encryption :



Decryption



RC5



Code:

```
package org.example.nobfx;
import javafx.application.Application
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.VBox;
import java.nio.ByteBuffer;
import java.nio.stage.Stage;
import java.nio.charset.StandardCharsets;
import java.util.Arrays;

public class RC5Fx1 extends Application
private static final int w = 32;
        "    "    int R = 12;
        "    "    int B = 16;
private "    "    int c = 4;
        "    "    int P = 0xB7E15163;
        "    "    int Q = 0x9E3779B9;
        "    "    int rot (int x, int y) {
```

```
    return (x << y) | (x >>> (w - y));
}

Private static void rc5 key setup (byte [] key)
{
  private static void RC5 key setup (byte [] key)
    int [] = new int [c];
    for (int i = B-1; i >= 0; i--) {
    L (i/4] = (L [i/4] << 8) + key [i] & 0xFF);
}
  S[0] = P;
  for (int i = 0; i < 2 * (R+1) ; ++i) {
        S [i] = S [i-1] + Q;
}

int A = 0, B = 0;
int i = 0, j = 0

Private static void rc5 Encrypt (int [] s, int [] data
{
    int A = data [0]
    int B = data [i]
```

```
        A = A + S[0];
        B = B + S[1];
    for (int i = 1; i <= R; ++i) {
        A = not | (A ^ B; B) + S[2*i];
        B = not | (B ^ A, A) + S[2*i+1];
    }
    data[0] = A;
    data[1] = B;
}
Private static String encryptText (String plainte.
        byte[] key = new byte[B];
        int[] S = new int[2*[R*1)];
        rc5 key Setup (key, 5);
byte[] plainBytes = plaintext.getBytes (standa
        charsets UTF = 8);
int .
```

```
while (buffer.hasRemaining()){
    int A = buffer.getInt();
    int B = buffer.getInt();
    int[] data = 2, A, B;
res5 Encrypt(5, data);
    ciphertex.append(String.format("%.08x
            %.08x", data[0], data[1]);
}
return ciphertext.toString().trim();
}
```

```
@ overnide
  public void start (Stage . stage){
 TextfieH input = new Textfield ();
input . sbtpnamptText ("Enten Englinh to Enc
Button enenypt Button = new Button ("Enenypt")
 TextArea output = new . TextArea ();
 output . set Edittable (falne);
 output . set unapText (tnue);
enenypt Button . set on Action (e →{
 String plainttext = input . getText ();
if (plaintext ! = null && ! plainText . is Empty (
 String ciphenText = enenypt . Text (plainTex
}
else
   { output . Settext ("Please enten nome text
   }
}
}
```

```
VBox layout = new VBox (10, input, encryptButton,
                        output);
layout.setStyle ("-fx-padding: 20;");
Scene scene = new Scene (layout, 500, 300);
stage.setScence (scene);
stage.show();
Public static void main (String [] args) {
    launch (args)
}
}
```

output

input : hello world

padding into 64 bit blocks

Block 1 : "hellowo" (8 bytes)

Block 2 : "rld10101010" (Padded)

Encrypted output

07c9f2a2, b1557b56, 24935031

original : hello world