

정보보호이론 [과제-1]

목차

[과제-1]A - (1)	-----	1
[과제-1]A - (2)	-----	2
[과제-1]B - (1)	-----	6
[과제-1]B - (2)	-----	8
[과제-1]B - (3)	-----	8

[과제-1]A - (1)

평문과 암호문들이 3비트로 구성된 Block 암호에서 이론적으로 사용 가능한 키의 개수
그리고 이에 해당하는 키의 최소길이는? **도출과정 설명 포함**

Permutation(순열) 개수 = $2^n!$ = Key 개수

개별 Table 식별자 길이 = $\log_2(2^n!) = \text{Key 길이}$

Key 개수 (3비트) = $2^3! = 8! = 40,320$

Key 길이 (3비트) = $\log_2(2^8!) = 15.29920801838728 \approx 15$

[과제-1]A - (2)

비트맵 파일을, ECB, CTR로 암호화 시킨 후에, 수업 시간에 논의된 Header 부분을 조정하여 암호화된 사진을 출력하여 비교 하시오.

(a) 평문 = 비트맵 파일 (자신의 성명과 학번이 기재된 파일)



(b) 각각의 java 소스 프로그램과 실행후의 Netbean 화면 캡처 출력

소스코드
<pre>package Symmetric; import javax.crypto.Cipher; import javax.crypto.KeyGenerator; import javax.crypto.SecretKey; import java.security.SecureRandom; import java.io.FileInputStream; import java.io.FileOutputStream; public class BMPFileEncryptNDecrypt { public static void main(String[] args) throws Exception { KeyGenerator keyGenerator = KeyGenerator.getInstance("DES"); keyGenerator.init(new SecureRandom()); SecretKey secretKey = keyGenerator.generateKey(); FileInputStream inFile = new FileInputStream("lku.bmp"); FileOutputStream outFile = new FileOutputStream("암호화 lku.bmp"); // Cipher cipher1 = Cipher.getInstance("DES/ECB/PKCS5Padding");</pre>

```

Cipher cipher1 = Cipher.getInstance("DES/CTR/PKCS5Padding");
cipher1.init(Cipher.ENCRYPT_MODE, secretKey);

byte[] input = new byte[64];
int bytesRead;
while ((bytesRead = inFile.read(input)) != -1) {
    byte[] output1 = cipher1.update(input, 0, bytesRead);
    if (output1 != null) {
        outFile.write(output1);
    }
}
byte[] output1 = cipher1.doFinal();
if (output1 != null) {
    outFile.write(output1);
}
inFile.close();
outFile.flush();
outFile.close();

Cipher cipher2 = Cipher.getInstance("DES");
cipher2.init(Cipher.DECRYPT_MODE, secretKey);

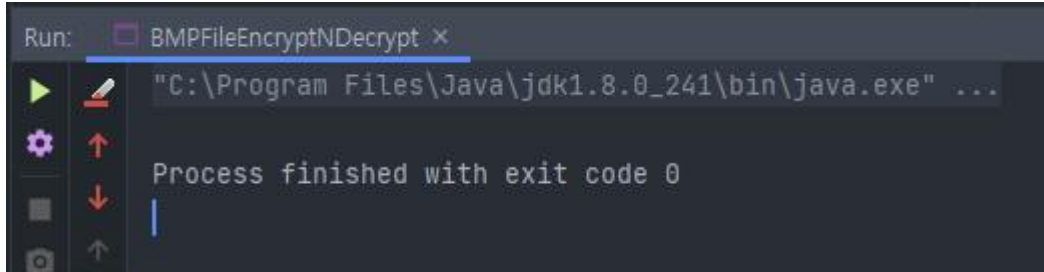
FileInputStream fis = new FileInputStream("암호화 lku.bmp");
FileOutputStream fos = new FileOutputStream("복호화 lku.bmp");

byte[] in = new byte[64];
int read;

while ((read = fis.read(in)) != -1) {
    byte[] output2 = cipher2.update(in, 0, read);
    if (output2 != null) {
        fos.write(output2);
    }
}
byte[] output2 = cipher2.doFinal();
if (output2 != null) {
    fos.write(output2);
}
fis.close();
fos.flush();
fos.close();
}
}

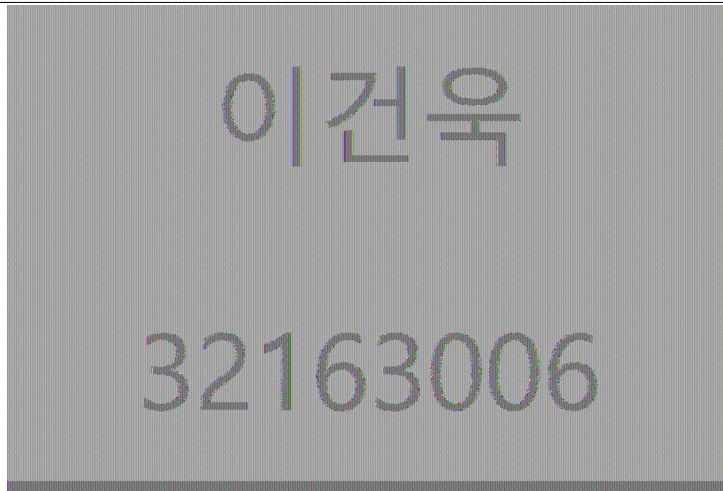
```

실행화면

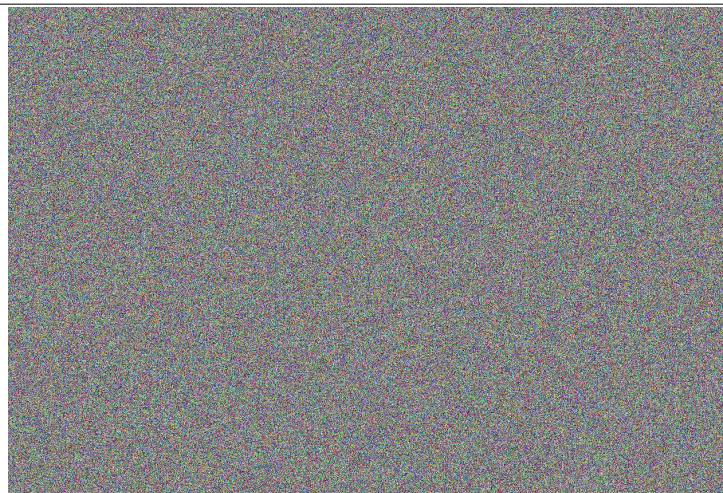


(c) 암호문 = 암호화된 사진 출력화면

수정된 암호화 Iku-ECB.bmp



수정된 암호화 Iku-CTR.bmp



(d) 복호화된 평문



[과제-1]B - (1)

Z_p^* 에 속하는 각각 구성원소에 대한 “곱셈 상의 역원 (Inverse)” 구하기

소스코드

```
package abstractAlgebra;

public class one {
    public static void main(String[] args) {
        int p = 37;
        int[] z = new int[37];

        for (int i = 0; i < p; i++) {
            z[i] = i + 1;
        }

        int[] inverse = new int[37];

        for (int i = 0; i < inverse.length; i++) {
            inverse[i] = -1;
        }

        for (int i = 0; i < z.length; i++) {
            for (int j = 0; j < z.length; j++) {
                if ((z[i] * z[j]) % p == 1) {
                    inverse[i] = z[j];
                    break;
                }
            }
        }

        for (int i = 0; i < inverse.length; i++) {
            if (inverse[i] == -1) break;
            System.out.println((i + 1) + " * " + inverse[i]);
        }
    }
}
```

실행결과

```
one x
"C:\Program
1 * 1
2 * 19
3 * 25
4 * 28
5 * 15
6 * 31
7 * 16
8 * 14
9 * 33
10 * 26
11 * 27
12 * 34
13 * 20
14 * 8
15 * 5
16 * 7
17 * 24
```

```
18 * 35
19 * 2
20 * 13
21 * 30
22 * 32
23 * 29
24 * 17
25 * 3
26 * 10
27 * 11
28 * 4
29 * 23
30 * 21
31 * 6
32 * 22
33 * 9
34 * 12
35 * 18
36 * 36
```


[과제-1]B - (2), (3)

Z_p^* 에 속하는 각각 구성원소에 대한 위수 (Order) 구하기

Z_p^* 에 속하는 구성원소 중에서 원시원소 (Primitive Element) 구하기

소스코드

```
package abstractAlgebra;

public class twoNththree {
    public static void main(String[] args) {
        int p = 7;
        int[] z = new int[7];
        int[] order = new int[6];

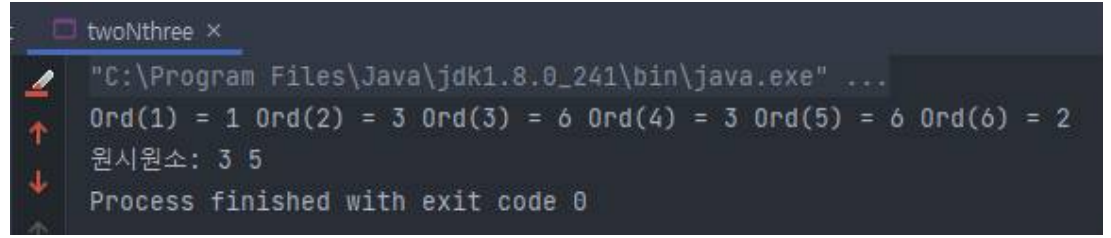
        for (int i = 0; i < z.length; i++) {
            z[i] = i + 1;
        }

        for (int i = 0; i < z.length - 1; i++) {
            int j = 1;
            while (true) {
                double result = Math.pow(z[i], j);
                if (result % p == 1) {
                    order[i] = j;
                    break;
                }
                j++;
            }
        }

        for (int i = 0; i < order.length; i++) {
            System.out.print("Ord(" + (i + 1) + ") = " + order[i] + " ");
        }

        System.out.println();
        System.out.print("원시원소: ");
        for (int i = 0; i < order.length; i++) {
            if (order[i] == order.length) {
                System.out.print((i + 1) + " ");
            }
        }
    }
}
```

실행결과



```
twoNthree ×
"C:\Program Files\Java\jdk1.8.0_241\bin\java.exe" ...
Ord(1) = 1 Ord(2) = 3 Ord(3) = 6 Ord(4) = 3 Ord(5) = 6 Ord(6) = 2
원시원소: 3 5
Process finished with exit code 0
```