**자료구조(박요한 교수님)**

**Caesar's Cipher 구현하기**

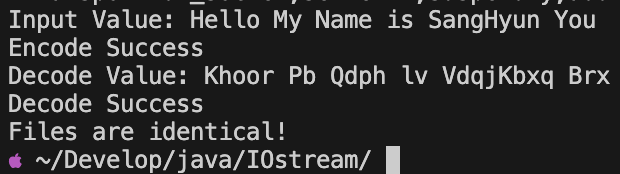
컴퓨터공학과 유상현

**1.** **프로그램 소스코드**

App.java

|  |
| --- |
| package Caeser;  import java.io.\*;  class FileManager{  private String INPUT\_FILE\_PATH = "/Users/sanghyun/Develop/java/IOstream/src/Caeser/input.txt";  private String ENCODED\_FILE\_PATH = "/Users/sanghyun/Develop/java/IOstream/src/Caeser/enc.txt";  private String RECOVERED\_FILE\_PATH = "/Users/sanghyun/Develop/java/IOstream/src/Caeser/recover.txt";  private int SHIFT = 3;  public void encodeAndDecodeFile() throws IOException{  FileReader fin = new FileReader(INPUT\_FILE\_PATH);  FileWriter fout = new FileWriter(ENCODED\_FILE\_PATH);  int c;  System.out.print("Input Value: ");  while ((c = fin.read()) != -1) {  System.out.print(((char) c));  fout.write(encrypt((char) c));  }  System.out.println("");  System.out.println("Encode Success");  fout.close();  fin.close();  FileReader encodedFin = new FileReader(ENCODED\_FILE\_PATH);  FileWriter recoverFout = new FileWriter(RECOVERED\_FILE\_PATH);  System.out.print("Decode Value: ");  while ((c = encodedFin.read()) != -1) {  System.out.print(((char) c));  recoverFout.write(decrypt((char) c));  }  System.out.println("");  System.out.println("Decode Success");  recoverFout.close();  encodedFin.close();  }  public char encrypt(char c) {  if (Character.isLetter(c)) {  char base = Character.isUpperCase(c) ? 'A' : 'a';  return (char) ((c - base + SHIFT + 26) % 26 + base);  }  return c;  }  public char decrypt(char c) {  if (Character.isLetter(c)) {  char base = Character.isUpperCase(c) ? 'A' : 'a';  return (char) ((c - base - SHIFT + 26) % 26 + base);  }  return c;  }  public void compareFiles() throws IOException{  FileReader originalFin = new FileReader(INPUT\_FILE\_PATH);  FileReader recoveredFin = new FileReader(RECOVERED\_FILE\_PATH);  int c1, c2;  while ((c1 = originalFin.read()) != -1 && (c2 = recoveredFin.read()) != -1) {  if (c1 != c2) {  System.out.println("Files are different!");  originalFin.close();  recoveredFin.close();  return;  }  }  System.out.println("Files are identical!");  originalFin.close();  recoveredFin.close();  }  }  public class App {  public static void main(String[] args) throws IOException{  FileManager fm = new FileManager();  fm.encodeAndDecodeFile();  fm.compareFiles();  }  } |

**2. 실행 화면**



텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

**3. 느낀 점(고찰)**

이번 파일 입출력과제를 하면서 터미널에서만 작동하던 코드를 메모장에서도 볼 수 있어서 많이 재밌는 과제가 되었습니다. 그리고 FileReader와 FileWriter의 사용법에 대해 이해할 수 있는 과제였습니다.