소프트웨어프로젝트
5주차 주간보고서

컴퓨터SW 18017103 황제현

5주차 진행내용

- 복호화 프로그램 함수 적용 및 테스트 완료
- 파일 단위로 암호화 기능 확장과 관련 변경사항

- 복호화 프로그램의 함수 도입과 테스트를 완료하였다.

CipherToPlain.py - C:₩Users₩one₩Desktop₩황 띠 시엔₩2022₩소프트웨어프로젝트₩기말 프로젝트₩소스...

File Edit Format Run Options Window Help

```
# ========================= Line ============================

def CipherToUniCode(cipher_t, datalength) :
    c_List = []
    for i in range(0, datalength) :
        c_List.append(ord(cipher_t[i]))

    return c_List

# ========================= Line ============================

def Todigit(uni_d, convert_n, datalength) :
    tmp_list = []
    for i in range(0, datalength) :
        tmpNum = uni_d[i]
        tmp_list.append(int(str(tmpNum), convert_n))

    return tmp_list

# ========================= Line ============================

def KeyStringToUniCode(key_string, datalength) :
    key_list = []
    for i in range(0, datalength) :
        key_list.append(ord(key_string[i]))

    return key_list

# ========================= Line ============================

def reXorFunc(cipher_list, key_list, datalength) :
    reXor_list = []
    for i in range(0, datalength) :
        reXor_list.append(cipher_list[i] ^ key_list[i])

    return reXor_list

# ========================= Line ============================

def CreatePlainText(reXor_list, datalength) :
```

Ln: 35 Col: 0

그림 1) 복호화 프로그램에 도입된 함수들

```
dataLength = len(cipherText)

# 암호문 -> 유니코드(n진법)
cipherList = CipherToUniCode(cipherText, dataLength)

# 유니코드(n진법) -> 유니코드(10진법)
cipherList = Todigit(cipherList, conversionNum, dataLength)

print("변환 완료.")

print("키 데이터를 변환하는중...")
# key 문자열 -> 유니코드 리스트
keyList = KeyStringToUniCode(keyString, dataLength)

print("변환 완료.")

print("데이터를 복호화 하는중...")
# (암호문 유니코드) XOR (key 유니코드)
reXorList = reXorFunc(cipherList, keyList, dataLength)

plainText = CreatePlainText(reXorList, dataLength)

outFp.writelines(plainText)

inFp.close()
outFp.close()
print("복호화가 완료되었습니다!")
```

그림 2)
함수가 도입된 메인 코드.
도입 전에 비해 훨씬 간결해졌다.

```
암호 파일의 위치를 입력하세요 >> C:₩Dev₩cipFile.txt
복호문을 저장할 파일 위치를 입력하세요 >>C:₩Dev₩reOriginFile.txt
파일이 확인되었습니다!
키 데이터를 입력하세요 >> ljzabzdufcvczkifxvaeru
변환에 사용된 진법을 입력하세요 >>7
변환 완료.
키 데이터를 변환하는중...
변환 완료.
데이터를 복호화 하는중...
복호화가 완료되었습니다!
```

그림 3) 테스트 화면

파일 단위로 암호화 기능 확장

- 기존 프로그램은 파이썬 셸 창에서 문자열 한 줄 단위로만 암호화가 가능하였다.
- 파일 입출력 코드를 추가하여 파일 단위로 데이터를 암호화할 수 있도록 기능을 확장하였다.
- 추가된 기능에 따라 필요한 변수를 추가하였다.
- 기능 구현을 위해 os 라이브러리를 추가하였다.
- 화이트 스페이스를 고려할 필요가 없어졌으므로 checkRC() 함수와 관련 실행코드는 삭제되었다.

```
import sys
import random
import os

plainText = "" # 암호화할 데이터
keyString = "" # 암,복호화에 사용할 키
keyList = [] # 연산을 위해 리스트화시킨 키
conversionNum = 0 # 변환할 진법
XorList = []
ConvertedXorList = []
InputEncodeType = 'UTF-8'
OutputEncodeType = 'UTF-8'
UniConvertedData = [] # 유니코드로 변환된 데이터
cipherText = "" # 암호화된 데이터
RejectChars = ['Wn', 'Wb', 'Wr', 'Wt', 'W"', 'W"', '#']
IsContainsRC = True
infilePath = '' # 암호화할 원본파일 경로
outfilePath = '' # 암호화된 파일을 저장할 경로
inFp = None
outFp = None
inList = ''
```

```
import sys
import os

cipherText = '' # 암호문
cipherList = [] # 유니코드로 변환시킨 암호문
reXorList = []
plainText = ''
conversionNum = 0
keyString = ''
keyList = []
InputEncodeType = 'UTF-8'
OutputEncodeType = 'UTF-8'
infilePath = ''
outfilePath = ''
inFp = None
outFp = None
inList = ''
```

그림 4, 5) 선언된 변수들 (좌 암호화, 우 복호화)

```
infilePath = input("암호화할 파일의 경로를 입력하세요:")
outfilePath = input("암호파일을 저장할 위치를 입력하세요.(공백 입력시 현재위치에 저장):")
inFp = open(infilePath, "r",  encoding = InputEncodeType)
outFp = open(outfilePath, "w", encoding = OutputEncodeType)

if os.path.exists(infilePath) == False :
    print(infilepath + ": 파일이 없습니다.")
    sys.exit(0)

if os.path.exists(outfilePath) == False :
    print(outfilepath + ": 파일이 없습니다.")
    sys.exit(0)


inList = inFp.readlines()
```

그림 6) 원본파일, 암호화파일 구분과 오류 검출 코드

정리

- 복호화 프로그램에 도입할 함수를 작성하고 테스트하였다.
- 파일 단위로 암호화할 수 있도록 기능을 확장했다.
- 완성된 프로그램의 코드를 첨부하였다.