컴퓨터 프로그래밍II 프로젝트 계획서

2018320184 강민규

주제: 암호 생성 및 해독기

1) 프로젝트 개요

a) 문제 상황 및 동기

고등학생 때 '암호의 해독'이라는 책을 읽고 암호라는 분야에 관심을 가지게 되었다. 그에 따라 암호학 스터디에 참여하기도 하였고 이번 학기에 핵심 교양 중 '암호학의 이해'라는 과목을 수강하게 되었다. 과목 자체는 무척이나 흥미로웠으나 수업을 듣는 것과 실제 실습 사이에는 큰 간극이 존재함을 머지않아 깨달을 수 있었다. 암호 관련 과제를 푸는데 문자 하나하나 분석하고 경우의 수를 여러 가지로 나누어 같은 작업을 수없이 반복해야 하는 것이 굉장히 불편하였다. 특히, 짧은 문장에서 그러한 현상이 두드러졌다. 기존의 통계 특성과 차이가 나타나기 때문이다. 그러다 보니 암호화나 복호화를 자동으로 해 주는 프로그램을 짠다면 훨씬 편하게 과제를 할 수 있고, 암호의 이해는 물론 프로그래밍 공부에도 큰 도움이 되지 않을까 하는 생각이 들게 되었다. 그리하여 나에게 주어진 이러한 문제 상황을 어떻게 하면 효과적으로 해결할 수 있을지 이번 기회를 통해 고민해 보는 시간을 가지고 싶었다.

b) 가능한 해결 방안 (방향성)

① 직접 손으로 암호문을 계산하되, 그 과정을 효율적으로 만들기 위한 보조 계산 기법을 구상하도록 한다. 그리고 그 효율성을 입증하기 위한 증명 과정 또한 완료하도록 한다.

② 암호화 과정과 복호화 과정을 모두 지원하는 하나의 JAVA 프로그램을 작성한다. 사용자는 암호화 기능을 사용할 지, 복호화 기능을 사용할 지 선택할 수 있으며 경우에 따라 평문 또는 암호문을 입력 받고 추가적으로 key 값과 같은 부수적인 요소들을 받는 방식을 취하는 함수를 만들도록 한다. 이에 대해서는 공백 처리나 입력 요소의 예외 케이스들을 구분하는 것이 중요하다.

③ 암호화 과정은 자동으로 완료해 주고, 복호화 과정은 부분적으로, 즉 문제 해결에 도움을 주는 보조 도구 형식의 JAVA 프로그램을 작성한다. 암호화 과정의 해결 방법은 방법 ②와 동일하게 하고, 복호화 과정의 경우는 해독을 위한 도움말을 주는 방식을 취한다. 예를 들어, substitution cipher의 복호화에 있어서 중요한 것은 평문의 통계 특성을 파악하여 그 정보를 바탕으로 문자를 대응시키는 것이 중요하다. 이 경우에서 암호문에 나타나는 문자들의 출현 빈도를 자동으로 분석해 주는 등의 기능까지만 지원하는 방식으로 하는 것이다.

c) 내가 선택한 방법

방법 ②와 방법 ③을 절충하여 문제 해결을 하도록 한다. 최대한 decryption을 자동적으로 수행해 주는 함수를 만들도록 하되, 경우의 수가 굉장히 많아 효율적인 방법을 찾기 어렵거나 코드 작성 시 예외 케이스들이 과도하게 많이 등장하는 경우 복호화를 위한 힌트, 방향성만을 제시해

주는 함수를 만들도록 한다.

d) 예상되는 모습, 기대 효과

여러 가지 암호들의 암호화 및 복호화 과정을 더욱 깊게 공부하면서 암호에 대한 이해도를 높일 수 있다. 이전에 손으로 계산하면서 암호를 공부할 때는 계산 과정을 따라가는 것에조차 다소 벅찬 느낌을 받았고 이해하는 데 시간이 오래 걸렸는데 이번 기회를 통해 암호라는 분야 자체에 대한 식견을 높일 수 있을 것이라 생각된다. 그리고 이는 추후 다른 암호들의 구조를 이해하거나 직접 하나의 암호 체계를 만들어 볼 때 탄탄한 밑바탕이 되어줄 것이다.

2) 프로젝트 계획

주차	작업 계획
1주차 (2018.10.28 ~ 2018.11.3)	main 함수 구현 프로그램이 실행될 때 어떤 기능을 사용할 것인가? 어떤 암호 방식을 선택할 것인가? (선택지 제공) try catch 구문을 이용해 적정 input을 제외한 다른 모든 값들을 예외 처리하도록 하기
2주차 (2018.11.4 ~ 2018.11.10)	•Shift cipher(=Caesar cipher)의 encryption 및 decryption 함수 구현 1) Encryption 함수: 암호화하고자 하는 문장과 shift시킬 key 값을 입력값으로 받는다. 이때, 문장 혹은 단락이 끝날 때까지 입력이 강제 종료되면 안 되므로 공백 등의 처리에 유의하며 특별한 입력을 통해 문장의 끝을 알린다. 이후에 key 값을 입력받는데 이는 무조건 integer 값을 받도록 한다. 이외의 경우에는 예외 케이스를 잡아 주게 하여 경고 메시지를 출력하도록 만든다. 2) Decryption 함수: 복호화 하고자 하는 문장을 입력받는다. 이때, 암호문은 공백 없이 받도록 한다. (암호문에 띄어 쓰기가 되어 있으면 해독이 무척이나 쉽기 때문) 복호화 방법은 전수 조사를 기준으로 한다. Key 값의 경우의 수가 26가지(mod 26의 경우는 의미가 거의 없으므로 사실상 25가지)에 불과하므로 각 경우에 대한 shift 결과를 모두 출력하여 보여주도록 한다. •Substitution cipher의 encryption 함수 구현

	➔ Encryption 함수: 평문의 각 알파벳을 암호문으로 바꿀 때 어떤 알파벳으로 받을지 그 대응 관계를 설정하도록 만들고 그 이후에 암호화 시킬 문장을 입력 받을 수 있도록 만든다.
3주차 (2018.11.11 ~ 2018.11.17)	•Substitution cipher의 decryption 함수 구현 ➔ Decryption 함수 ⇨ 평문의 통계 특성을 반영하여 그대로 암호문을 해독할 수 있게 만들거나, ⇨ 암호를 해독할 수 있는 힌트까지만 제공하도록 코드를 짠다. (추후 결정) • Vigenère cipher의 encryption 및 decryption 함수 구현 1) Encryption 함수: 평문을 입력 받고 key length를 몇으로 적용할 지 사용자에게 확인을 받고 이후에 그 length에 해당하는 key word를 받아 그에 따라 평문을 나누어 암호화하도록 한다. 이때 key 값이 여러 개라는 점을 제외하면 shift cipher와 매우 유사한 원리임을 인지한다. 2) Decryption 함수: ⇨ kasiski test와 index of coincidence 등의 연산 방법을 활용하여 암호문을 해독할 수 있게 만들거나, ⇨ 암호를 해독할 수 있는 힌트까지만 제공하도록 코드를 짠다. (추후 결정)
4주차 (2018.11.18 ~ 2018.11.24)	•Transposition cipher의 encryption 및 decryption 함수 구현 1) Encryption 함수: 입력 받은 평문을 5글자씩 나누고 5글자로 이루어진 문자열쌍들을 어떠한 순서로 섞을 것인지 숫자 입력을 통해 그 배열을 정한다. 그리고 평문의 문자들의 개수가 완전히 5로 나누어지지 않을 경우에는 임의의 더미 데이터가 들어가게 만든다. 2) Decryption 함수: ⇨ 배열의 순서쌍을 분석하여 암호문을 해독할 수 있게 만들거나, ⇨ 암호를 해독할 수 있는 힌트까지만 제공하도록 코드를 짠다. (추후 결정)
5주차 (2018.11.25 ~ 2018.12.1)	•Block cipher (SPN structure)의 encryption 함수 구현 ➔ Encryption 함수: 입력 받은 평문을 SPN structure를 기준으로 XOR 연산과 Permutation 연산을 통해 암

	호화 과정을 생성한다. # Block cipher에 한하여 decryption 과정은 만들지 않을 것이다. Block cipher의 구조가 매우 다양하고 XOR 연산이나 순서쌍 생성에 대한 경우의 수가 굉장히 많이 나오기 때문이다.
6주차 (2018.12.2 ~ 2018.12.8)	프로그램 수정 및 보완
7주차 (2018.12.9 ~ 2018.12.12)	프로그램 테스트 및 마무리 & 제출

3) 참고한 자료

핵심 교양 (GEST124) [암호학의 이해] 수업 자료 (ppt)

A handbook of applied cryptography

Serious Cryptography – A Practical Introduction to Modern Encryption

추후 프로젝트 진행에 따라 stack overflow 등의 사이트를 참고할 것임.