**Lab 10**

|  |
| --- |
| 그림입니다. 원본 그림의 이름: YU_UI_RGB-10.png 원본 그림의 크기: 가로 2256pixel, 세로 3047pixel 프로그램 이름 : Adobe ImageReady |

|  |  |
| --- | --- |
| 과목명 | 프로그래밍 언어 |
| 교수님 | 김영탁 교수님 |
| 이 름 | 김주환 |
| 학 번 | 21812158 |
| 일 자 | 2021.05.14.금 |

// main.cpp(2)

void test\_SelectionSortWords() {

FILE\* fin, \*fout;

char\*\* word\_array;

int num\_words = 0;

fin = fopen("words.txt", "r"); // 데이터 파일 읽기모드 열기

fout = fopen("sorted\_words.txt", "w"); // 정렬 데이터를 입력할 파일 쓰기모드 열기

word\_array = (char\*\*)calloc(MAX\_NUM\_WORDS, sizeof(char\*));

for (int i = 0; i < MAX\_NUM\_WORDS; i++)

word\_array[i] = (char\*)calloc(21, sizeof(char));

for (int i = 0; i < MAX\_NUM\_WORDS; i++) { // 실제 단어 개수 파악

if (fscanf(fin, "%s", word\_array[i]) == EOF)

break;

num\_words++;

}

selectionSortWords(word\_array, num\_words); // selection sorting

for (int i = 1; i <= num\_words; i++) { // 출력

fprintf(fout, "%22s", word\_array[i - 1]);

if ((i % 10) == 0)

fprintf(fout, "\n");

}

free(word\_array); // 동적 할당 해제

fclose(fin); // 파일 닫기

}

void test\_RandomFileAccess\_Student() {

FILE\* fin, \* fout;

int ns = 0;

int record\_len;

extern Student students[];

char student\_record[MAX\_STRING\_LEN] = { 0 };

for (int i = 0; i < MAX\_NUM\_STUDENTS; i++) { // 학생 수 파악

if (students[i].st\_id == -1)

break;

else ns++;

}

fout = fopen("Sorted\_Students.txt", "w"); // 정렬된 데이터를 입력할 파일 쓰기모드 열기

if (fout == NULL) {

printf("Error in opening Sorted\_Students.txt (write mode)!!\n");

exit;

}

/\*

\* 파일명 : main.cpp

\* 목적 및 기본 기능

\* SelectionSortWord, RandomFileAccess, BinaryFileDump, Cipher

\* 작성자 : 김주환(21812158)

\* 작성일 : 2021년 5월 14일

\*/

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#include "MyString.h"

#include "Student.h"

#include "HandlingBinaryFile.h"

#include "CipherMessage.h"

#define MAX\_NUM\_WORDS 1000

#define NUM\_STUDENTS 10

#define MAX\_STRING\_LEN 512

#define MAX\_NUM\_STUDENTS 100

extern Student students[NUM\_STUDENTS];

#define TEST\_SORTING

#define TEST\_BST\_Student

#define MAX\_STR\_LEN 100

#define CIPHER\_CODE 0X00 // cipher\_code

void test\_SelectionSortWords();

void test\_RandomFileAccess\_Student();

void test\_BinaryFileDump();

void test\_simple\_cipher\_text();

void main() {

int menu;

while (1) {

printf(" 1. Test selection sort for words\n");

printf(" 2. Test random access file for students records\n");

printf(" 3. Test binary file dump\n");

printf(" 4. Test cipher text\n");

printf(" 0. Quit\n");

printf("Input menu : ");

scanf("%d", &menu);

if (menu == 0)

break;

switch (menu) {

case 1:

test\_SelectionSortWords();

break;

case 2:

test\_RandomFileAccess\_Student();

break;

case 3:

test\_BinaryFileDump();

break;

case 4:

test\_simple\_cipher\_text();

break;

default:

break;

}

}

}

// main.cpp(4)

void test\_simple\_cipher\_text() {

FILE\* fp\_msg, \* fp\_tx, \* fp\_rx, \* fp\_dump\_msg, \* fout;

fp\_msg = fopen("Message.txt", "r"); // 암호화할 파일 읽기모드 열기

if (fp\_msg == NULL) {

printf("Error in file open - Message.txt !!\n");

exit(-1);

}

fp\_tx = fopen("Ciphered.txt", "w"); // 암호화된 데이터를 입력할 파일 쓰기모드 열기

if (fp\_tx == NULL) {

printf("Error in file open - Transmit.txt !!\n");

exit(-1);

}

fout = fopen("Deciphered.txt", "w"); // 복호화된 데이터를 입력할 파일 쓰기모드 열기

if (fout == NULL) {

printf("Error in file open - Output.txt !!\n");

exit(-1);

}

printf("Generating cipher text with cipher-code (%#04x) ..\n", CIPHER\_CODE);

cipherText(fp\_msg, fp\_tx, CIPHER\_CODE); // 암호화

fclose(fp\_tx); // 암호화된 데이터 파일 닫기

rewind(fp\_msg); // 커서 위치 초기화

fprintf(fout, "Binary dump of message.txt file: \n");

dumpBinaryFile(fp\_msg, fout);

fp\_tx = fopen("Ciphered.txt", "r"); // 암호화된 데이터 파일 읽기모드 열기

if (fp\_tx == NULL) {

printf("Error in file open - Transmit.txt !!\n");

exit(-1);

}

fprintf(fout, "\n=========================================\n");

fprintf(fout, "Binary dump of ciphered document: \n");

dumpBinaryFile(fp\_tx, fout);

rewind(fp\_tx); // 커서 위치 초기화

printf("Generating de-ciphered text with cipher-code (%#04x) ..\n", CIPHER\_CODE);

fprintf(fout, "\n=========================================\n");

fprintf(fout, "Generating de-ciphered text with cipher-code (%#04x) ..\n", CIPHER\_CODE);

deCipherText(fp\_tx, fout, CIPHER\_CODE); // 복호화

fclose(fp\_msg); // 초기 데이터 파일 닫기

fclose(fp\_tx); // 암호화된 데이터 파일 닫기

fclose(fout); // 복호화된 데이터 파일 닫기

}

// main.cpp(3)

fprintf(fout, "Array of students at initialization : \n");

fprintStudents(fout, students, ns);

fprintf(fout, "\n");

sortStudents\_by\_ST\_ID(students, ns); // 학번에 따른 정렬

fprintf(fout, "Storing sorted students by increasing order of student ID into SortedStudent.txt ....\n");

fprintStudents(fout, students, ns);

fprintf(fout, "\n");

fclose(fout); // 파일 닫기

fin = fopen("Sorted\_Students.txt", "r"); // 정렬된 데이터 파일 읽기모드 열기

if (fin == NULL) {

printf("Error in opening Sorted\_Students.txt (read mode)!!\n");

exit;

}

fout = fopen("students.txt", "w"); // 내림차순으로 정렬하고 커서 위치를 출력할 파일 쓰기모드 열기

if (fin == NULL) {

printf("Error in opening Sorted\_Students.txt (read mode)!!\n");

exit;

}

int cur\_pos;

rewind(fin); // 커서 위치 초기화

fgets(student\_record, MAX\_STRING\_LEN, fin); // 원하는 크기만큼의 문자열 읽어오기

record\_len = strlen(student\_record); // 해당 문자열 길이 반환

fprintf(fout, "\nRandom access to Sorted\_students.txt file (Student record length: %d) ...\n", record\_len);

for (int i = ns - 1; i >= 0; i--) { // 내림차순 정렬

fseek(fin, (record\_len + 1) \* i, SEEK\_SET); // 커서 위치 이동

cur\_pos = ftell(fin); // 커서 위치 반환

fprintf(fout, "Current file\_position : %3d\n", cur\_pos);

fgets(student\_record, record\_len + 1, fin);

fprintf(fout, "Student (%2d): ", i);

fprintStudent(fout, students + i);

fprintf(fout, "\n");

}

fclose(fin); // 파일 닫기

fclose(fout); // 파일 닫기

}

void test\_BinaryFileDump() {

FILE\* fin, \* fout;

const char\* fname = "Sorted\_Students.txt"; // 실행 파일 생성

if ((fin = fopen(fname, "rb")) == NULL) { // 실행 파일 읽기/ 바이너리 모드 열기

printf("Error - binary input file (%s) cannot be openned !!\n", fname);

exit;

}

if ((fout = fopen("Output.txt", "w")) == NULL) { // 출력할 파일 쓰기모드 열기

printf("Error - Output.txt cannot be created !!\n");

exit;

}

printf("Dumping binary file (%s) ... \n", fname);

dumpBinaryFile(fin, fout);

fclose(fin); // 파일 닫기

fclose(fout); // 파일 닫기

}

**10.1 문자열의 파일 입력, 정렬 및 파일 출력**

// MyString.h

#ifndef MS\_H

#define MS\_H

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

void selectionSortWords(char\*\* words, int num\_words);

#endif

// MyString.cpp

#ifndef MS\_H

#define MS\_H

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

#include "MyString.h"

void selectionSortWords(char\*\* words, int num\_words) {

char\* temp\_word;

int i, j, min;

temp\_word = (char\*)calloc(num\_words, sizeof(char));

if (temp\_word == NULL) {

printf("Error in dynamic allocation of memory for temp\_word !!\n");

exit;

}

for (i = 0; i < num\_words; i++) {

strcpy(temp\_word, words[i]);

min = i;

for (j = i + 1; j < num\_words; j++) {

if (strcmp(temp\_word, words[j]) > 0) {

min = j;

strcpy(temp\_word, words[j]);

}

}

if (min != i) {

strcpy(words[min], words[i]);

strcpy(words[i], temp\_word);

}

}

}

#endif



**10.2 학생 데이터 레코드의 랜덤 파일 입출력**

// Student.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS

#ifndef STUDENT\_H

#define STUDENT\_H

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

#include "Student.h"

#define U\_SHORT unsigned short

#define MAX\_NAME\_LEN 20

#define NUM\_STUDENTS 10

typedef struct {

int year;

int month;

int day;

} Date;

typedef struct {

U\_SHORT nation\_code;

U\_SHORT region\_no;

U\_SHORT switch\_no;

U\_SHORT line\_no;

} Tel\_Number;

typedef struct {

int st\_id;

char name[MAX\_NAME\_LEN];

Date birth\_date;

Tel\_Number tel\_number;

double GPA; // Grade Point Average

} Student;

void printStudent(Student\* pST) {

printf("Student [ID: %08d, %-10s", pST->st\_id, pST->name);

printf(", GPA: %5.2lf", pST->GPA);

printf(", ");

printf("(%04d, %2d, %2d)", pST->birth\_date.year, pST->birth\_date.month, pST->birth\_date.day);

printf(", ");

printf("(tel: +%d-", pST->tel\_number.nation\_code);

printf("%03d-%04d-%04d)", pST->tel\_number.region\_no, pST->tel\_number.switch\_no, pST->tel\_number.line\_no);

printf("]");

}

// Student.h

#define \_CRT\_SECURE\_NO\_WARNINGS

#ifndef STUDENT\_H

#define STUDENT\_H

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

#define U\_SHORT unsigned short

#define MAX\_NAME\_LEN 20

#define NUM\_STUDENTS 10

typedef struct {

int year;

int month;

int day;

} Date;

typedef struct {

U\_SHORT nation\_code;

U\_SHORT region\_no;

U\_SHORT switch\_no;

U\_SHORT line\_no;

} Tel\_Number;

typedef struct {

int st\_id;

char name[MAX\_NAME\_LEN];

Date birth\_date;

Tel\_Number tel\_number;

double GPA; // Grade Point Average

} Student;

void printStudent(Student\* pST);

void printStudents(Student \* stArr, int num\_students);

void fprintStudent(FILE\* fout, Student\* pST);

void fprintStudents(FILE\* fout, Student\* stArr, int num\_students);

void sortStudents\_by\_ST\_ID(Student students[], int num);

#endif

텍스트이(가) 표시된 사진

자동 생성된 설명

/\* StudentsRecords.cpp \*/

#include "Student.h"

#define NUM\_STUDENTS 10

Student students[NUM\_STUDENTS + 1] = {

{ 21711000, "Kim, G-M", { 1990, 10, 5 }, { 82, 53, 805, 1234 }, 3.57 },

{ 21611075, "Yoon, S-M", { 1990, 4, 5 }, { 82, 53, 811, 1550 }, 4.37 },

{ 21411015, "Hwang, S-S", { 1989, 1, 10 }, { 82, 53, 817, 1005 }, 2.72 },

{ 21611054, "Lee, K-M", { 1991, 5, 15 }, { 82, 10, 9112, 9876 }, 3.35 },

{ 21611340, "Hong, G-M", { 1990, 2, 5 }, { 82, 55, 810, 5678 }, 3.89 },

{ 21712056, "Jang, S-M", { 1990, 3, 15 }, { 82, 10, 9112, 1600 }, 4.42 },

{ 21411017, "Park, S-S", { 1989, 7, 10 }, { 82, 34, 817, 1098 }, 4.12 },

{ 21511053, "Choi, Y-H", { 1992, 9, 25 }, { 82, 53, 845, 5764 }, 3.85 },

{ 21411017, "Shin, D-J", { 1988, 10, 3 }, { 82, 31, 817, 1038 }, 3.21 },

{ 21511053, "Kwak, S-B", { 1994, 11, 15 }, { 82, 2, 897, 8778 }, 4.45 },

{-1}

};

// Student.cpp(2)

void printStudents(Student\* stArr, int num\_students) {

Student\* st = stArr;

for (int i = 0; i < num\_students; i++) {

printStudent(st);

printf("\n");

st++;

}

}

void fprintStudent(FILE\* fout, Student\* pST) {

fprintf(fout, "Student [ID: %08d, %-10s", pST->st\_id, pST->name);

fprintf(fout, ", GPA: %5.2lf", pST->GPA);

fprintf(fout, ", ");

fprintf(fout, "(%04d, %2d, %2d)", pST->birth\_date.year, pST->birth\_date.month, pST->birth\_date.day);

fprintf(fout, ", ");

fprintf(fout, "(tel: +%d-", pST->tel\_number.nation\_code);

fprintf(fout, "%03d-%04d-%04d)", pST->tel\_number.region\_no, pST->tel\_number.switch\_no, pST->tel\_number.line\_no);

fprintf(fout, "]");

}

void fprintStudents(FILE\* fout, Student\* stArr, int num\_students) {

Student\* st = stArr;

for (int i = 0; i < num\_students; i++) {

fprintStudent(fout, st);

fprintf(fout, "\n");

st++;

}

}

void sortStudents\_by\_ST\_ID(Student students[], int num) {

/\* Selection Sorting student array by ST\_ID in increasing order \*/

Student\* pST, \* pST\_min;

Student temp;

int st\_id\_min;

for (int i = 0; i < num; i++) {

pST\_min = pST = &students[i];

st\_id\_min = i;

for (int j = i + 1; j < num; j++) {

pST = &students[j];

if (pST\_min->st\_id > pST->st\_id) {

pST\_min = pST;

st\_id\_min = j;

}

} // end for

if (st\_id\_min != i) {

temp = students[st\_id\_min];

students[st\_id\_min] = students[i];

students[i] = temp;

}

}

}

#endif

**10.3 이진파일의 dump**

// HandlingBinaryFile.h

#ifndef HB\_H

#define HB\_H

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

void dumpBinaryFile(FILE\* fin, FILE\* fout);

#endif

// HandlingBinaryFile.cpp

#ifndef HB\_H

#define HB\_H

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include "HandlingBinaryFile.h"

#define BUFFER\_LEN 16

#define MAX\_FILE\_SIZE 256

void dumpBinaryFile(FILE\* fin, FILE\* fout) {

int nbytes;

unsigned char buffer[BUFFER\_LEN] = { '\0' };

for (int addr = 0; addr < MAX\_FILE\_SIZE; addr += BUFFER\_LEN) {

nbytes = fread(buffer, sizeof(unsigned char), BUFFER\_LEN, fin);

if (nbytes <= 0)

break;

fprintf(fout, "%08X: ", addr);

for (int i = 0; i < nbytes; i++) {

if (i == (BUFFER\_LEN / 2))

fprintf(fout, " ");

fprintf(fout, "%02X ", buffer[i]);

}

fprintf(fout, " ");

for (int i = 0; i < nbytes; i++) {

if (isprint(buffer[i]))

fprintf(fout, "%c", buffer[i]);

else

fprintf(fout, ".");

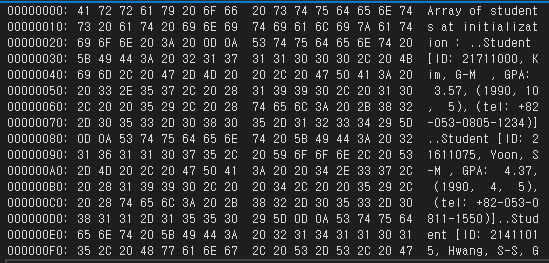
}

fprintf(fout, "\n");

}

}

#endif



**10.4 간단한 문자열의 16진수 문자로의 암호화**

// CipherMessage.cpp(2)

unsigned char deCipherChar(unsigned char rc\_u, unsigned char rc\_l) { // 비트 결합

unsigned char rst;

unsigned char rc\_u\_2, rc\_l\_2;

if (rc\_u >= 'A') // 10 ~ 15

rc\_u\_2 = rc\_u - 'A' + 10;

else // 0 ~ 9

rc\_u\_2 = rc\_u - '0';

if (rc\_l >= 'A')

rc\_l\_2 = rc\_l - 'A' + 10;

else

rc\_l\_2 = rc\_l - '0';

rst = rc\_u\_2 << 4; // 결합

rst |= rc\_l\_2;

return rst;

}

void deCipherText(FILE\* fp\_tx, FILE\* fp\_out, unsigned char cipher\_code) {

int len;

char text[1000] = { '\0' };

unsigned char trsf;

unsigned char rc\_u, rc\_l, rst;

while (fgets(text, 1000, fp\_tx) != NULL) { // 문자열 불러오기

len = strlen(text) - 1; // 불러온 문자열 길이

for (int i = 0; i < len; i += 2) {

rc\_u = text[i]; // 상위

rc\_l = text[i + 1]; // 하위

rst = deCipherChar(rc\_u, rc\_l); // 결합과정

trsf = rst ^ cipher\_code; // 복호화

atox(trsf);

fprintf(fp\_out, "%c", trsf);

}

printf("\n");

}

}

#endif

// CipherMessage.cpp

#ifndef CM\_H

#define CM\_H

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

#include <ctype.h>

#include "CipherMessage.h"

void atox(unsigned char uc) { // 16진수를 문자로 변환

printf("%c", uc);

}

void xtoa(unsigned char uc) { // 문자를 16진수로 변환

unsigned char temp1, temp2;

temp1 = uc / 16; // 상위비트

temp2 = uc % 16; // 하위비트

printf("%X%X", temp1, temp2);

}

void cipherChar(unsigned char ch, unsigned char\* pCC\_u, unsigned char\* pCC\_l) { // 비트 분리

\*pCC\_u = ch / 16;; // 상위비트 반환

\*pCC\_l = ch % 16; // 하위비트 반환

}

void cipherText(FILE\* fp\_msg, FILE\* fp\_tx, unsigned char cipher\_code) {

int len;

unsigned char trsf;

char text[1000] = { '\0' };

unsigned char rc\_u = 0, rc\_l = 0;

unsigned char\* pCC\_u = &rc\_u, \* pCC\_l = &rc\_l;

while (fgets(text, 1000, fp\_msg) != NULL) { // 문자열 불러오기

len = strlen(text); // 불러온 문자열 길이

for (int i = 0; i < len; i++) {

trsf = text[i] ^ cipher\_code; // 암호화

xtoa(trsf);

cipherChar(trsf, pCC\_u, pCC\_l); // 분리과정

fprintf(fp\_tx, "%X%X", rc\_u, rc\_l);

}

fprintf(fp\_tx, "\n");

printf("\n");

}

}

// CipherMessage.h

#ifndef CM\_H

#define CM\_H

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

void atox(unsigned char uc);

void xtoa(unsigned char uc);

void cipherChar(unsigned char ch, unsigned char\* pCC\_u, unsigned char\* pCC\_l);

void cipherText(FILE\* fp\_msg, FILE\* fp\_tx, unsigned char cipher\_code);

unsigned char deCipherChar(unsigned char rc\_u, unsigned char rc\_l);

void deCipherText(FILE\* fp\_tx, FILE\* fp\_out, unsigned char cipher\_code);

#endif

텍스트, 명판이(가) 표시된 사진

자동 생성된 설명

Oral Test

|  |
| --- |
| **(1) C 프로그램에서 파일 입력 및 파일 출력에 사용되는 fscanf(), fprintf(), fgets(), fputs에 대하여 예를 들어 설명하라. 각 함수에서 사용가능한 형식 지정자에 대하여 설명할 것.**   * int fscanf(FILE\* fp, const chat\* format, …); 지정된 포멧으로 파일을 읽어 해당 항목으로 복사 * int fprintf(FILE\* fp, const chat\* format, …); 지정된 포멧으로 해당 항목들을 파일로 출력   위의 두 경우는 기본 입출력 함수에서 사용가능한 형식 지정자를 모두 사용 가능하다.   * char\* fgets(char\* buf, int max\_size, FILE\* fp); 문자열을 최대 max\_size만큼 읽어 버퍼에 저장 * int fputs(const char\*buf, FILE\* fp); 버퍼에 저장되어 있는 문자열을 파일에 쓰기   위의 두 경우는 문자열 단위로 진행되기 때문에 따로 형식 지정자를 지정하지 않는다. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **(2) ASCII 코드표에서 숫자, 대문자, 소문자의 값이 어떤 구간에 있는지에 대하여 설명하라.**   * 숫자(10진수, 16진수)  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | | 0X30 | 0X31 | 0X32 | 0X33 | 0X34 | 0X35 | 0X36 | 0X37 | 0X38 | 0X39 |  * 대문자  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | | 0X41 | 0X42 | 0X43 | 0X44 | 0X45 | 0X46 | 0X47 | 0X48 | 0X49 | 0X4A | 0X4B | 0X4C | 0X4D | 0X4E | 0X4F | 0X50 | 0X51 | 0X52 | 0X53 | 0X54 | 0X55 | 0X56 | 0X57 | 0X58 | 0X59 | 0X5A |  * 소문자  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | | 97 | 98 | 99 | 100 | 101 | 102 | 103 | 104 | 105 | 106 | 107 | 108 | 109 | 110 | 111 | 112 | 113 | 114 | 115 | 116 | 117 | 118 | 119 | 120 | 121 | 122 | | 0X61 | 0X62 | 0X63 | 0X64 | 0X65 | 0X66 | 0X67 | 0X68 | 0X69 | 0X6A | 0X6B | 0X6C | 0X6D | 0X6E | 0X6F | 0X70 | 0X71 | 0X72 | 0X73 | 0X74 | 0X75 | 0X76 | 0X77 | 0X78 | 0X79 | 0X7A | |

|  |
| --- |
| **(3) 함수 void generateCipherText(FILE\* fp\_msg, FILE\* fp\_out, unsigned char cipher\_code)에서 암호화된 문자열이 어떻게 생성되며, 이 문자열을 주어진 암호를 모르는 경우 해석할 수 없는 이유에 대하여 설명하라.**   * 파일 데이터 fgets를 통해 지정된 파일의 문자 데이터를 읽어온다. * 암호화 위에서 읽어온 문자 데이터를 한 문자씩 뽑아서 cipher\_code와 ‘^’연산을 실행한다. * 해석 불가능한 이유 만약, 128비트 이상의 암호코드를 사용하여, 암호화를 하는 경우, 이를 암호코드 없이 해독하여야 하는 경우 수퍼 컴퓨터로도 수백년 이상 걸리게 된다. |

|  |
| --- |
| **(4) 함수 void decipherText(FILE\* fp\_tx, FILE\* fp\_out, unsigned char cipher\_code)를 사용하여 암호화된 문자열을 원래의 문자열로 변환시키는 기능에 대하여 상세하게 설명하라.**   * 파일 데이터 암호화 과정에서 생성한 16진수 데이터를 fgets를 이용해 불러온다. * 두자리 위에서 불러온 16진수 데이터는 현재 경계가 없으므로 2개씩 나눠서 연산한다. * 복호화 생성한 2자리 16진수 집합의 각 자리가 영어 대문자인지 숫자인지 조건문을 통해 판단하고, 영어 대문자라면 ASCII를 이용해 해당 값에 ‘A’를 빼주고 10을 더한다. 왜냐하면 ‘A’는 16진수로 10이기 때문이다. 반대로 해당 값이 숫자라면 위와 같은 이유로 값에 ‘0’을 빼준다. |