|  |  |  |
| --- | --- | --- |
| **2월 5일 1주차 1번 과제** | **제출일 : 2024.2.10** | **이 선 용** |

고객 니즈

1. 자판기의 기능을 갖고 있는 프로그램을 만드시오.

2. 사용자로부터 금액을 입력 받고, 사용자가 구매할 수 있는 상품 목록을 보여줍니다.

3. 사용자는 원하는 상품을 선택하고, 프로그램은 선택된 상품의 가격을 사용자의 금액에서 차감한 후 잔액을 보여줍니다.

4. 프로그램은 여러 상품을 연속적으로 구매할 수 있으며, 잔액이 부족할 경우 구매를 할 수 없다는 메시지를 출력합니다.

상품 목록 : 콜라 사이다 물 비타600

가. 격 : 1800 1300 1000 600

주의 사항 : 해당 프로그램을 만들 때 입력, 출력, if문, for문을 꼭 사용해야 합니다.

pseudo code

1. 최초 1회에만 소비자가 사용 가능한 돈을 먼저 입력 받는다.(이하 사용자돈)

1-1. 그 뒤부터는 계속 품목명을 입력 받는다.

2. 가지고 있는 잔액과 내가 다음에 사려는 물건 값을 비교하여

2-1. 구매 가능 하다면(if true false)

2-2. 반복문을 반복

2-3. 구매할 수 없다면 잔액부족으로 프로그램 종료

변수 : 사용자돈, 품목을 담은 배열, 품목에 대한 가격 배열, 어떤 물건을 살지 입력 받을 배열,

for문에서 사용할 i 변수

#include <stdio.h>

#include <string.h>

#include <stdbool.h>

int main() {

int usrmoney; // 사용자가 최초에 가지고 있는 돈을 입력받을 정수 변수

char stuff[4][10] = {"콜라", "사이다", "물", "비타600"}; // 비타600 길이 생각하고 널널하게 잡아서 배열 10 할당해줌 그러고 품목은 4개니 2차원배열로 배열 초기화.

int price[4] = {1800, 1300, 1000, 600}; // 배열 길이 4로 배열 초기화.

char buy[10]; // 사용자가 구매할 품목을 입력 받을 배열 변수

int i; //for for^^

printf("금액을 입력하세요: ");

scanf("%d", &usrmoney);

while(1) { // 무한 츠쿠요미 "ON"

bool canBuy = false; // 일단 살 능력이 '없다'로 먹고 들어가기. 왜 와이? 매번 사용자가 구매할 형편이 되는지 판단하기 위해!

// 상품 목록 출력용 for문

printf("상품 목록: ");

for(i = 0; i < 4; i++) {

printf("%s / ", stuff[i]);

}

// 가격 출력용 for문

printf("\n가 격: ");

for(i = 0; i < 4; i++) {

printf("%d / ", price[i]);

}

printf("\n");

printf("구매할 물건을 입력하세요: ");

scanf("%s", buy);

// 사실 이 부분부터가 자판기 프로그램의 가장 핵심 로직 부분이라 생각 됨.

for(i = 0; i < 4; i++) { // 물건 종류든 가격이든 4종류니까 4번 반복!

if(strcmp(buy, stuff[i]) == 0) { // strcmp는 파이썬에서 if('???' == '\*\*\*') 같은 느낌? 맞으면 0 리턴 다르면 1 리턴

if(usrmoney >= price[i]) { // 사용자가 가진 돈이 품목들 중 살 수 있는 품목이 있으면

usrmoney -= price[i]; // 사용자 돈에서 그 금액만큼 깐다.

printf("잔액: %d원\n", usrmoney); // 그러고 잔액 출력.

canBuy = true; // 구매가 가능한지 여부 확인

break; // for문 종료

} else {

printf("잔액이 부족합니다.\n");

}

}

}

if(!canBuy) { // 살 능력없으면

break; // while문 탈출

}

}

return 0;

}

}

|  |  |  |
| --- | --- | --- |
| **2월 5일 1주차 2번 과제** | **제출일 : 2024.2.10** | **이 선 용** |

/\*

생년월일 입력 받아서 몇 월인지 영어/별자리로 출력하기[배열]

1) 사용자로부터 생년월일을 입력받고, 해당 월을 영어로 출력하며 해당 월일에 맞게 별자리를 알려주는 프로그램을 작성하시오.

2) 사용자로부터 정확한 생년월일을 입력 받아야하며, 올바르지 않은 입력

[예: 잘못된 날짜 형식, 존재하지않는 날짜 등]이 입력되면 프로그램은 종료되어야 함.

주의 사항

1) 해당 프로그램을 만들땐 배열을 필수적으로 사용할 것.

2) 입력받은 생년월일을 배열을 이용해서 처리하면 됨.

3) 입력받은 월과 일을 기반으로 별자리를 결정함.

별자리는 특정 날짜 범위에 따라 결정되므로, 이 범위를 정의하고 입력받은 날짜가 어느 범위에 속하는지 확인하는 로직을 구현해야 함.

별자리표

양자리 : 3월 21일 ~ 4월 19일

황소자리 : 4월 20일 ~ 5월 20일

쌍둥이자리 : 5월 21일 ~ 6월 21일

게자리 : 6월 22일 ~ 7월 22일

사자자리 : 7월 23일 ~ 8월 22일

처녀자리 : 8월 23일 ~ 9월 23일

천칭자리 : 9월 24일 ~ 10월 22일

전갈자리 : 10월 23일 ~ 11월 22일

사수자리 : 11월 23일 ~ 12월 24일

염소자리 : 12월 25일 ~ 1월 19일

물병자리 : 1월 20일 ~ 2월 18일

물고기자리 : 2월 19일 ~ 3월 20일

=================================================

pseudo code

변수

1) 쪼개진 년 배열, 월 배열, 일 배열

2) for문에 사용할 i

3) 영어로 된 달 배열

4) 입력받은 월에 따라 영어로 된 달을 리턴하는 함수

'-' 기준으로 생년월일을 입력받는다. - 완료

'-'를 기준으로 년 월 일을 나눠서 각 배열에 저장한다. - 완료

1월이면 영어 월 배열에서 1번째 배열에 있는 수를 적용.

1월일때 "january"가 나와야하는 조건 0<mm && mm<2

for

\*/

#include <stdio.h>

// 각 별자리의 시작 날짜 (월과 일)

const int start[12][2] = { //각 별자리별로 시작하는 달과 일을 2차원 배열로 초기화

{3, 21},

{4, 20},

{5, 21},

{6, 21},

{7, 23},

{8, 23},

{9, 23},

{10, 23},

{11, 22},

{12, 22},

{1, 20},

{2, 19}

};

// 각 별자리의 이름

const char \*const star\_signs[12] = {

"양자리",

"황소자리",

"쌍둥이자리",

"게자리",

"사자자리",

"처녀자리",

"천칭자리",

"전갈자리",

"사수자리",

"염소자리",

"물병자리",

"물고기자리" //별자리 배열

};

char month[12][10] = {

"January",

"Fabuary",

"March",

"April",

"May",

"June",

"July",

"August",

"September",

"October",

"November",

"December"}; //달 배열

char\* what\_is\_month(int mm) { // 입력한 달을 판별하여 그에 맞는 영어 달을 판별하는 함수

if (mm >= 1 && mm <= 12)

return month[mm - 1];

else

return "";

}

int main() {

int year;

int month;

int day;

int i;

printf("생년월일을 입력하세요 (YYYY-MM-DD): ");

scanf("%04d-%d-%d", &year, &month, &day);

// 입력한 날짜를 판별해 영어 달과 별자리를 출력하는 로직

for (i = 0; i < 12; i++) {

if ((month == start[i][0] && day >= start[i][1]) ||

(month == start[(i + 1) % 12][0] && day < start[(i + 1) % 12][1])) {

printf("%s %s\n", what\_is\_month(month), star\_signs[i]);

break;

}

}

if (i == 12) {

printf("유효하지 않은 날짜입니다.\n");

}

return 0;

}

/\*

별자리표

양자리 : 3월 21일 ~ 4월 19일

황소자리 : 4월 20일 ~ 5월 20일

쌍둥이자리 : 5월 21일 ~ 6월 21일

게자리 : 6월 22일 ~ 7월 22일

사자자리 : 7월 23일 ~ 8월 22일

처녀자리 : 8월 23일 ~ 9월 23일

천칭자리 : 9월 24일 ~ 10월 22일

전갈자리 : 10월 23일 ~ 11월 22일

사수자리 : 11월 23일 ~ 12월 24일

염소자리 : 12월 25일 ~ 1월 19일

물병자리 : 1월 20일 ~ 2월 18일

물고기자리 : 2월 19일 ~ 3월 20일

// printf("Year: %d\n", yy[0]);

// printf("Month: %02d\n", mm[0]);

// printf("Day: %02d\n", dd[0]);

\*/

|  |  |  |
| --- | --- | --- |
| **2월 5일 1주차 3번 과제** | **제출일 : 2024.2.10** | **이 선 용** |

/\*

구글링을 하다보니 stackoverflow에 인코딩 디코딩에 대한 openssl 코드 중 일부를 발견.

Base64 인코딩, 디코딩에 대해 구글링을 통해 학습 및 이해한 후, 학습한 내용을 토대로 문자열을 입력받아서 해당 문자열을 인코딩, 디코딩 하는 프로그램을 작성하시오.

!!주의사항!!

1. 해당 프로그램을 만들 때 포인터를 필수적으로 사용할 것.
2. 포인터를 사용하여 인코딩/디코딩할 입력 받은 문자열에 접근 가능.

includePath 업데이트 오류로 터미널에서 아래 명령어를 실행하여 작동을 확인하였음. /opt/homebrew/include 와 /opt/homebrew/lib는 저의 Mac 로컬환경에서 openssl 헤더가 있는 위치입니다.

>> gcc -o endecoding endecoding.c -I/opt/homebrew/include -L/opt/homebrew/lib -lssl -lcrypto

>> ./endecoding

\*/

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#include <openssl/bio.h>

#include <openssl/evp.h>

#include <openssl/buffer.h>

#define MAX\_INPUT\_LENGTH 1024

char\* base64\_encode(const unsigned char\* input, int length) {

BIO \*bio, \*b64;

BUF\_MEM \*bufferPtr;

char \*encoded\_buffer;

b64 = BIO\_new(BIO\_f\_base64());

bio = BIO\_new(BIO\_s\_mem());

bio = BIO\_push(b64, bio);

BIO\_set\_flags(bio, BIO\_FLAGS\_BASE64\_NO\_NL);

BIO\_write(bio, input, length);

BIO\_flush(bio);

BIO\_get\_mem\_ptr(bio, &bufferPtr);

BIO\_set\_close(bio, BIO\_NOCLOSE);

BIO\_free\_all(bio);

encoded\_buffer = (char \*)malloc(bufferPtr->length + 1);

memcpy(encoded\_buffer, bufferPtr->data, bufferPtr->length);

encoded\_buffer[bufferPtr->length] = '\0';

BUF\_MEM\_free(bufferPtr);

return encoded\_buffer;

}

char\* base64\_decode(const char\* input, int length) {

BIO \*bio, \*b64;

char \*decoded\_buffer = (char \*)malloc(length);

memset(decoded\_buffer, 0, length);

b64 = BIO\_new(BIO\_f\_base64());

bio = BIO\_new\_mem\_buf(input, length);

bio = BIO\_push(b64, bio);

BIO\_set\_flags(bio, BIO\_FLAGS\_BASE64\_NO\_NL);

BIO\_read(bio, decoded\_buffer, length);

BIO\_free\_all(bio);

return decoded\_buffer;

}

int main() {

char choice[MAX\_INPUT\_LENGTH];

char input\_string[MAX\_INPUT\_LENGTH];

char\* encoded\_string;

char\* decoded\_string;

while (1) {

printf("1. Base64 encoding\n");

printf("2. Base64 decoding\n");

fgets(choice, sizeof(choice), stdin);

choice[strcspn(choice, "\n")] = 0;

if (strcmp(choice, "1") == 0) { // 1을 입력했을 때

fgets(input\_string, sizeof(input\_string), stdin);

input\_string[strcspn(input\_string, "\n")] = 0;

encoded\_string = base64\_encode((const unsigned char \*)input\_string, strlen(input\_string));

printf("%s\n", encoded\_string);

free(encoded\_string);

break;

} else if (strcmp(choice, "2") == 0) { // 2를 입력했을 때

fgets(input\_string, sizeof(input\_string), stdin);

input\_string[strcspn(input\_string, "\n")] = 0;

decoded\_string = base64\_decode(input\_string, strlen(input\_string));

printf("%s\n", decoded\_string);

free(decoded\_string);

break;

} else {

break;

}

}

return 0;

}

|  |  |  |
| --- | --- | --- |
| **2월 5일 1주차 4번 과제** | **제출일 : 2024.2.10** | **이 선 용** |

/\* Input으로 받은 txt 파일의 내용을 base64로 인코딩하기(파일입출력, 동적 메모리 할당)

1. txt 파일을 만들어서 해당.txt[test.txt] 파일에 있는 문자열을 읽어서 인코딩, 디코딩을 수행할 수 있게 프로그램을 작성.
2. !!주의사항!!

배열을 사용하지 않고 malloc을 사용해서 파일 내용 처리해야 할 것.

입력한 문자열이 있는 test.txt 파일에 있는 문자열을 처리해야 하기 때문에 파일 입출력을 사용해야함.

// gcc -o malFileIO mal\_fileIO.c -I/opt/homebrew/include -L/opt/homebrew/lib -lssl -lcrypto

// malloc을 이용한 파일 입출력

// 텍스트파일에 Hello h4c!! 가 적혀있으면 SGVsbG8gaDRjISE= 로 인코딩됨. 디코딩은 그 반대.

\*/

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#include <openssl/bio.h>

#include <openssl/evp.h>

#include <openssl/buffer.h>

#define MAX\_INPUT\_LENGTH 1024

char\* base64\_encode(const unsigned char\* input, int length) {

BIO \*bio, \*b64;

BUF\_MEM \*bufferPtr;

char \*encoded\_buffer;

b64 = BIO\_new(BIO\_f\_base64());

bio = BIO\_new(BIO\_s\_mem());

bio = BIO\_push(b64, bio);

BIO\_set\_flags(bio, BIO\_FLAGS\_BASE64\_NO\_NL);

BIO\_write(bio, input, length);

BIO\_flush(bio);

BIO\_get\_mem\_ptr(bio, &bufferPtr);

BIO\_set\_close(bio, BIO\_NOCLOSE);

BIO\_free\_all(bio);

encoded\_buffer = (char \*)malloc(bufferPtr->length + 1);

memcpy(encoded\_buffer, bufferPtr->data, bufferPtr->length);

encoded\_buffer[bufferPtr->length] = '\0';

BUF\_MEM\_free(bufferPtr);

return encoded\_buffer;

}

char\* base64\_decode(const char\* input, int length) {

BIO \*bio, \*b64;

char \*decoded\_buffer = (char \*)malloc(length);

memset(decoded\_buffer, 0, length);

b64 = BIO\_new(BIO\_f\_base64());

bio = BIO\_new\_mem\_buf(input, length);

bio = BIO\_push(b64, bio);

BIO\_set\_flags(bio, BIO\_FLAGS\_BASE64\_NO\_NL);

BIO\_read(bio, decoded\_buffer, length);

BIO\_free\_all(bio);

return decoded\_buffer;

}

int main() {

char choice[MAX\_INPUT\_LENGTH];

char filename[MAX\_INPUT\_LENGTH];

char \*input\_string = NULL;

char \*output\_string = NULL;

size\_t len = 0;

printf("1. Base64 encoding\n");

printf("2. Base64 decoding\n");

fgets(choice, sizeof(choice), stdin);

choice[strcspn(choice, "\n")] = 0;

fgets(filename, sizeof(filename), stdin);

filename[strcspn(filename, "\n")] = 0;

FILE \*file = fopen(filename, "r");

if (file == NULL) {

printf("Error opening file.\n");

return 1;

}

getline(&input\_string, &len, file);

fclose(file);

if (strcmp(choice, "1") == 0) {

// 읽어온 내용을 인코딩

output\_string = base64\_encode((const unsigned char \*)input\_string, strlen(input\_string));

} else if (strcmp(choice, "2") == 0) {

// 읽어온 내용을 디코딩

output\_string = base64\_decode(input\_string, strlen(input\_string));

} else {

printf("Invalid choice.\n");

return 1;

}

file = fopen(filename, "w");

if (file == NULL) {

printf("Error opening file for writing.\n");

return 1;

}

fprintf(file, "%s", output\_string);

fclose(file);

printf("변환 완료.\n");

if (input\_string != NULL)

free(input\_string);

if (output\_string != NULL)

free(output\_string);

return 0;

}