2차 프로젝트 과제/고급C프로그래밍/박철수교수님/엄희상조교님/컴퓨터정보공학부/2019202052/김호성/12.09

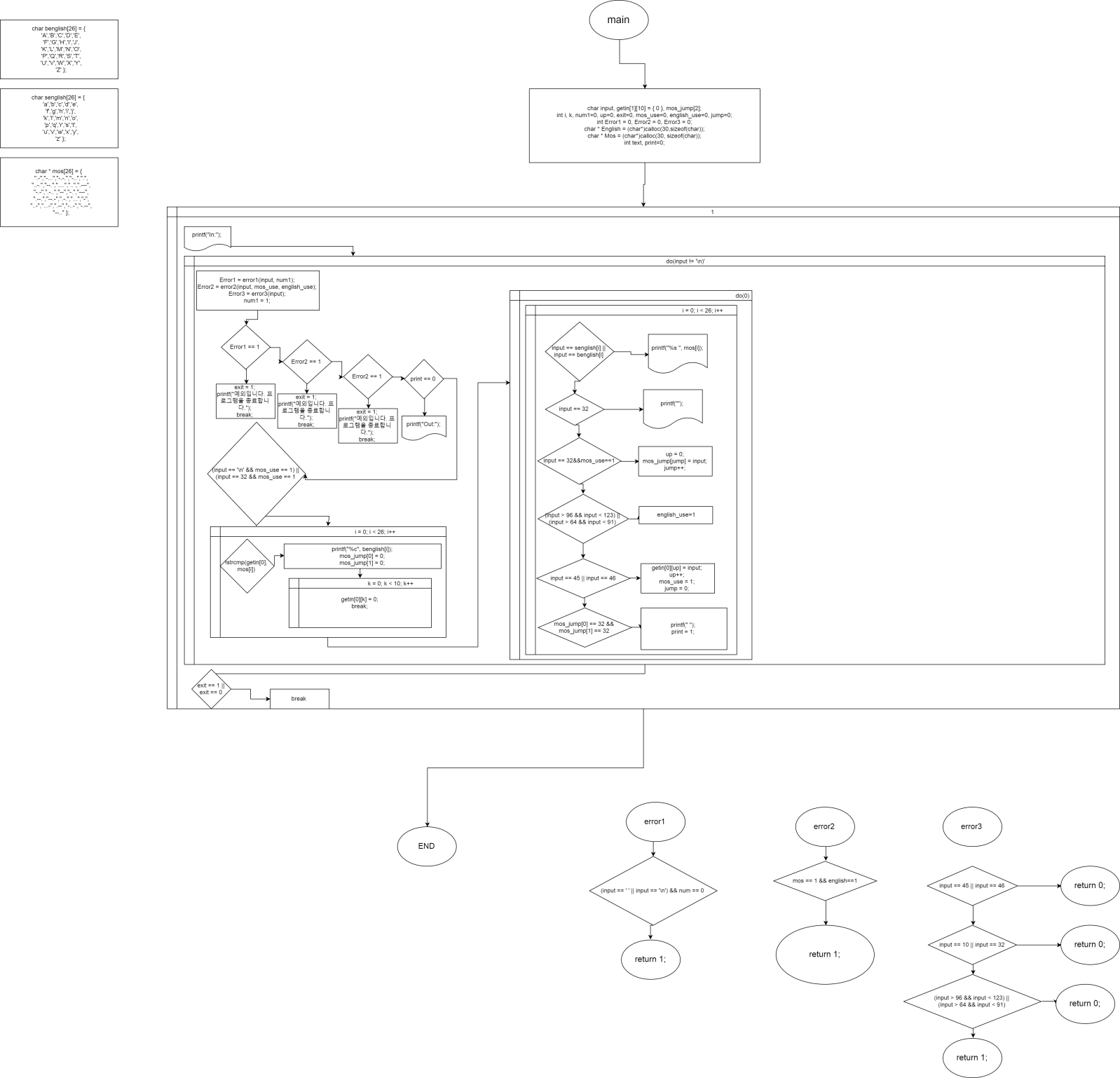
TEST 01

1. 과제소개

영어나 모스코드를 입력해 영어일 경우 모스코드로 모스코드일 경우 영어로 변환한다.

(단, 처음에 영어나 모스코드 값이 아니면 예외처리, 영어와 모스코드가 섞여 있어도 예외처리, 영어나 모스코드 이외의 다른 글씨나 기호가 섞여 있어도 예외처리한다.)

2. 설계내용(flowchart)



3. 코드 및 주석

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

char benglish[26] = {

'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' };//출력을 위한 영어

char senglish[26] = {

'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' };//입력받기 위한 영어

char \* mos[26] = {

".-","-...","-.-.","-..",".",

"..-.","--.","....","..",".---",

"-.-",".-..","--","-.","---",

".--.","--.-",".-.","...","-",

"..-","...-",".--","-..-","-.--",

"--.." };//모스부호

int error1(char input, int num) //첫번쨰 입력 때 스페이스바 또는 엔터를 입력하였다.

{

if ((input == ' ' || input == '\n') && num == 0)

return 1;

}

int error2(char input,int mos,int english)//모스부호가 입력되다 영어로 변하였다.(반대경우 에서도)

{

if ( mos == 1 && english==1)

return 1;

}

int error3(char input) //다른 문자가 들어 갈 경우

{

if (input == 45 || input == 46)

return 0;

else if (input == 10 || input == 32)

return 0;

else if ((input > 96 && input < 123) || (input > 64 && input < 91))

return 0;

else

return 1;

}

int main()

{

char input, getin[1][10] = { 0 }, mos\_jump[2];

//input 입력 받을 값

//getin input에 값을 getin에 저장

//mos\_jump 스페이스바(10)이 두번 입력되면 모스부호도 띄어쓰기를 위한 변수 설정

int i, k, num1=0, up=0, exit=0, mos\_use=0, english\_use=0, jump=0;// i 반복문을 위함

int Error1 = 0, Error2 = 0, Error3 = 0;

char \* English = (char\*)calloc(30,sizeof(char)); //30칸짜리 동적할당

char \* Mos = (char\*)calloc(30, sizeof(char)); //30칸짜리 동적할당

int text, print=0;

while (1)

{

printf("In:");

do { //한글자씩 변환되는 게 반복되는 dowhile문

input = getchar(); //사용자에게 값을 입력받음

Error1 = error1(input, num1); //오류확인1 (num1이 0은 무조건 참인 조건)

Error2 = error2(input, mos\_use, english\_use); //오류확인2

Error3 = error3(input); //오류확인3

num1 = 1;//45= "-",46=".", 32=" ", 10="\n"

if (Error1 == 1)

{

exit = 1;

printf("예외입니다. 프로그램을 종료합니다.");

break; //dowhile문 탈출

}

if (Error2 == 1)

{

exit = 1;

printf("\n코드를 해석하던 중 다른 언어가 들어왔습니다. 프로그램을 종료합니다.");

break; //dowhile문 탈출

}

if (Error3 == 1)

{

exit = 1;

printf("\n영어나 모스부호가 아닌 다른 값을 입력했습니다. 프로그램을 종료합니다.");

break; //dowhile문 탈출

}

if (print == 0) //처음에만 out을 출력하기 위함. 첫번째 바퀴 마지막에서 print = 1로 바꿔줌.

printf("Out:");

if ((input == '\n' && mos\_use == 1) || (input == 32 && mos\_use == 1))

{

for (i = 0; i < 26; i++)//모스부호를 입력 받았을 때

{

if (!strcmp(getin[0], mos[i])) //mos[i]값이 0이 아니면 실행

{

printf("%c", benglish[i]); //대문자 출력

mos\_jump[0] = 0;

mos\_jump[1] = 0;

for (k = 0; k < 10; k++) //모스부호는 여러글자 영어는 한글자씩 이므로

getin[0][k] = 0;

break; //영어를 입력 받았을 때로 넘어가지 않기 위해 필요.

}

}

}

do{

for (i = 0; i < 26; i++)//영어를 입력 받았을 때

{

if (input == senglish[i] || input == benglish[i])//일치 할 때마다 하나씩 값을 받는다.

{

printf("%s ", mos[i]); //포인터로 할당한 구조체 변수에서 실행

}

}

}while(0); //처음에 무조건 값을 안받는 오류를 해결하기 위함(처음 do while문이 돌때는 english\_use값이 무조건 0임.)

if(english\_use==1)

{

if (input == 32) //아스키코드 값으로 스페이스바가 32임.

{

printf(" ");

}

}

if (input == 32&&mos\_use==1)

{

up = 0; //한개가 입력되고 증가 -다음 값을 받을 준비를 한다.

mos\_jump[jump] = input; //모스 부호에 사용을 알린다.

jump++; //점프 값을 초기화해준다.

}

else if ((input > 96 && input < 123) || (input > 64 && input < 91))//a=97, z=122, A=65, Z=90

english\_use=1;

else if (input == 45 || input == 46)

{

getin[0][up] = input; //input에 입력값을 배열에 하나하나 저장한다.

up++; //한개가 입력되고 증가 -다음 값을 받을 준비를 한다.

mos\_use = 1; //모스 부호에 사용을 알린다.

jump = 0; //점프 값을 초기화해준다.

}

if (mos\_jump[0] == 32 && mos\_jump[1] == 32)

printf(" "); //출력

print = 1; //out을 출력을 안하기 위해 증가시킨다.

} while (input != '\n'); //input값이 \n이 올때까지 반복한다.

if (exit == 1) //오류가 걸리면 무한반복 탈출

break;

else if (exit == 0)

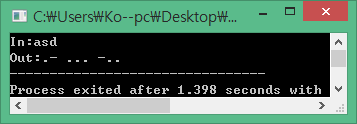
break; //정상적으로 dowhile문이 종료되어도 탈출

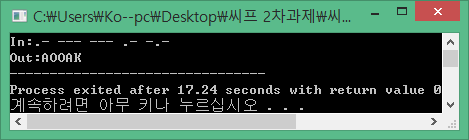
}

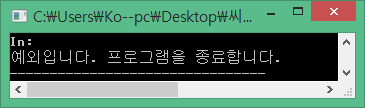
return 0; //main함수 종료

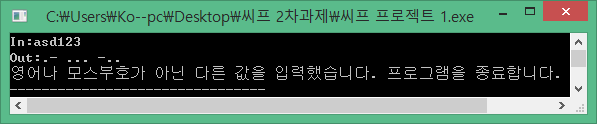
}

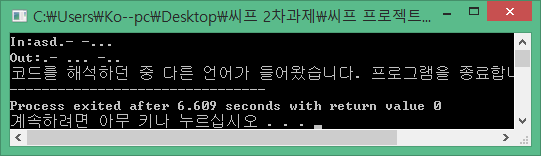
4. 실행결과











5. 고찰

-현재 만든 코드의 경우에는 do while문을 사용해서 변환이 이루어지다가 예외처리를 발생하게 함.

(출력이 되는 도중 바뀜.)

-

TEST 02

1. 과제소개

텍스트파일에서 스트림 rt를 사용해 파일을 읽어온 후 5장씩 플레이어에게 나눠준 후 족보를 비교해 누가 어떤 카드로 이겼는지 알 수 있으며, 게임의 결과를 전체출력하기도, 부분출력하기도 할 수 있어야 하고, 프로그램 내에서만 삭제도 가능해야 하며, 자동 메모리 영역 해제 전에 해제를 시켜줘야 한다.

2. 설계내용(flowchart)

Typedef struct card{}Card;

1. 카드 패에 관한 구조체 후에 족보를 위한 정보들도 구조화시킴

void mapping(char \*c);

1. 16진수로 값 할당 함수

sort(char \*shape, int \*number);

1. 숫자 기준 오름차순 정렬함수

int high\_shape(char shape[5]);

1. 가장 높은 카드의 문양을 지정하는 함수

int straight\_check(int num[]);

1. 스트레이트 유무 함수

int flush\_check(char color[]);

1. 플러쉬 유무 함수

calculation(int col, int li, char card[11])

1. 계산함수

re\_mapping(char \* c);

1. 16진수로 할당한 값 10, J, Q, K, A로 변환하는 함수

final\_print(int result, int num1, int num2);

1. 최종 출력함수

teamA.player == (char\*)calloc(11,sizeof(char));

1. 동적 할당

scanf("%d", &select);

1. select에 값을 받음.

if (select == 2 || select == 3)

scanf("%d", &game\_number);

1. 조건에 따라 게임넘버에 값을 받을 수도 있음.

else if (select == 4){

printf("동적할당한 메모리 영역을 해제 합니다.");

break;

} while문 나가고

free(teamA.player);

free(teamB.player);

fclose(f);

1. select=4일 때 동적 메모리 영역을 해제한다.

else if (select != 1 && select != 2 && select != 3 && select != 4)

break;

1. 1, 2, 3, 4 이외의 값이면 예외처리

fopen\_s(&f, "input2.txt", "rt");

1. 텍스트 파일을 읽기모드로 개방

memcpy(teamA.player, s[k], sizeof(char) \* 10);

1. player에 택스트 10개만 저장한다.

for (i = 0; i < 10; i = i + 2) {

teamA.shape[j] = teamA.player[i + 1];

teamA.number[j] = teamA.player[i];

j++;

}

1. player값에서 shape와 number값을 추출한다.

else if (select == 2)

printf("teamA:%s\nteamB:%s\n", teamA.player, teamB.player);

if (teamA.result > teamB.result)

{

result = 1;

final\_print(result, teamA.result, num2);

}

else if (teamA.result < teamB.result)

{

result = 2;

final\_print(result, teamB.result, num2);

}

else

{

if (teamA.high\_shape > teamB.high\_shape)

{

result = 1;

final\_print(result, teamA.result, num2);

}

else

{

result = 2;

final\_print(result, teamB.result, num2);

}

}

1. final\_print함수에 전달할 인자를 정하는 조건으로 final\_print에 들어가 누가 어떤 패로 이겼는지 출력한다.

if (select == 2)

{

num2 = 0; //초기화한다.

break; //탈출

}

1. 값을 받고 select2면 한 번 출력 후 선택으로 1이면 전체출력으로 넘어간다.

for (i = 0; i < 4; i++) //버블정렬 숫자에 맞게

{

for (j = i + 1; j < 5; j++)

{

if (number[i] > number[j])

{

aa = number[i]; //카드 숫자와

number[i] = number[j];

number[j] = aa;

bb = shape[i]; //카드 문양이 같이 움직인다.

shape[i] = shape[j];

shape[j] = bb;

}

}

}

1. sort함수 안에 있는 정렬 알고리즘으로 카드와 문양의 순서가 섞이지 않게(패가 바뀌지 않게) 같이 움직인다.

void mapping(char \* c) { //A, K, Q, J, 10은 16진수 E, D, C, B, A로 바꿔준다.

int i = 0;

for (i = 0; i < 5; i++)

{

if (c[2 \* i] == 'A')

c[2 \* i] = '0xE';

else if (c[2 \* i] == 'K')

c[2 \* i] = '0xD';

else if (c[2 \* i] == 'Q')

c[2 \* i] = '0xC';

else if (c[2 \* i] == 'J')

c[2 \* i] = '0xB';

else if (c[2 \* i] == '0')

c[2 \* i] = '0xA';

else;

}

}

1. 포커 카드의 경우 13개의 숫자와 4개의 문양으로 나뉘므로 13진수 이상으로 받아야 숫자 비교가 가능하기 때문에 16진수로 값을 변환해주었다.

int straight\_check(int num[]) //정렬되어 있는 상태이므로 +1씩 하면서 스트레이트의 유무를 확인한다.

{

int i;

int c = 1;

for (i = 0; i < 4; i++) //숫자가 연속적인가?

{

if (num[i] + 1 != num[i + 1])

if (num[i + 1] == 14 && num[i] == 5)

c++;

else

break;

c++;

}

}

1. sort함수 이후에 불러오는 함수이므로 정렬이 이미 된 상태에서 왼쪽의 숫자보다 오른쪽의 숫자가 1씩 커지는 게 반복되면 스트레이트 조건을 만족한다.

for (i = 0; i < 4; i++) //한 번 검사했던 경우를 겹치지 않게 검사하고 모든 조건을 검사하는 알고리즘

{

for (j = 0; j < 4 - i; j++)

{

if (card[2 \* i] == card[2 + (2 \* j) + (2 \* i)])

{

count++;

plus = card[2 + (2 \* j) + (2 \* i)];

}

}

}

1. 족보 계산 중 페어 계산을 위해 작성한 알고리즘으로 [1,2] , [1,3], [1,4],[1,5]…[4,5]순으로 비교한다.

void final\_print(int result, int num1, int num2)

1. result는 누가 이겼는지, num2는 select이 1일 때(메인 함수에 적혀있음.) num1은 select이 2일 때 사용되어 출력한다.

3. 코드 및 주석

//스페이드 하트 클로버 다이아 S H C D, A~2

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_COLS 100000 //MAX\_COLS 는 100000이라는 수를 가짐

typedef struct card { //카드 패에 관한 구조체 후에 족보를 위한 정보들도 구조화시킴

char player[11];

int number[5];

char shape[5];

int straight, flush, result, high\_shape;

}CARD;

void mapping(char \*c); //16진수로 값 할당 함수

void sort(char \*shape, int \*number); //숫자 기준 오름차순 정렬함수

int high\_shape(char shape[5]); //가장 높은 카드의 문양

int straight\_check(int num[]); //스트레이트 유무 함수

int flush\_check(char color[]); //플러쉬 유무 함수

int calculation(int col, int li, char card[11]); //계산함수

void re\_mapping(char \* c); //16진수로 할당한 값 10, J, Q, K, A로 변환하는 함수

void final\_print(int result, int num1, int num2); //최종 출력

//함수순서대로 메인함수에서 진행함.

int main()

{

FILE \*f; //파일 선언

CARD teamA, teamB; //구조체 변수 선언

teamA.player == (char\*)calloc(11,sizeof(char)); //동적 할당

teamB.player == (char\*)calloc(11,sizeof(char)); //동적 할당

int select = 0; //번호를 받을 변수 선언

int game\_number = 0; //번호를 받을 변수 선언

int result = 0; //족보 계산 후 A가 이기면 1 B가 이기면 2라는 값을 가짐.

int num1 = 0, num2 = 0, num3 = -1; //select의 조건에 맞게 사용하기 위한 변수

int i, j = 0,k=0;

int asd = 0;

char s[4][MAX\_COLS];

while (1)

{

select = 0;

game\_number = 0;

printf("--------------------------------------------------------------\n");

printf("1. Print all Game result\n");

printf("2. Print Game(number)\n");

printf("3. Delete Game(number)\n");

printf("4. Exit\n");

printf("--------------------------------------------------------------\n");

printf("Select option::");

scanf("%d", &select);

if (select == 2 || select == 3) //select이 2 또는 3일때만

scanf("%d", &game\_number);

else if (select == 4){ //4일때 무한반복 탈출

printf("동적할당한 메모리 영역을 해제 합니다.");

fclose(teamA.player);

fclose(teamB.player);

fclose(f);

break;

}

else if (select != 1 && select != 2 && select != 3 && select != 4) //다른 값 입력시 프로그램 종료

break;

fopen\_s(&f, "input2.txt", "rt"); //텍스트 파일을 읽기모드로 개방

for (k = 0; k < 4; k++) //s배열을 만들어 한개씩 저장한다

fgets(s[k], MAX\_COLS, f);

for (k = 0; k < 4; k++)

{

if (select == 3) //3일 경우 실행

{

num3 = game\_number - 1; //num3값을 game\_number입력 받은거에서 하나 작은 값을 저장한다.

num2 = 2;

asd = 1; //3을 선택했다는 것을 표시해준다.

}

else if (select == 2) //2를 선택 했다는 것을 표시한다.

{

num2 = 1;

k = game\_number - 1; //game\_number입력 받은 것에서 1 작은 값을 저장한다.

}

if (select == 2 && (k >= num3) && asd == 1) //3에서 num3입력 받은 값과 k값이 같거나 클 때 실행

k++; //하나 증가시켜 뒤에있는 값을 받아온다

if (select == 1)

num2 = 0; //1이 선택되면 초기화한다.

if (k != num3) //num3과 k값이 같다면 실행하지 않는다

{

teamA.player[10] = '\0';

teamB.player[10] = '\0';

memcpy(teamA.player, s[k], sizeof(char) \* 10); //player에 택스트 10개만 저장한다.

memcpy(teamB.player, s[k] + 10, sizeof(char) \* 10); //player에 택스트 10개를 저장하는데 위에 것보다 10앞에 있는 것을 저장한다.

for (i = 0; i < 10; i = i + 2) //숫자와 모양을 나누어 받을 반복문

{

teamA.shape[j] = teamA.player[i + 1]; //모양을 저장한다.

teamA.number[j] = teamA.player[i]; //숫자를 저장한다.

j++; //player에는 1,3,5...홀 수에는 모양이 짝수에는 숫자가 저장되어 있고 그것을 5개 나누어 받기 위함

}

j = 0; //다른 값도 받기 위해 초기화

for (i = 0; i < 10; i = i + 2) //숫자와 모양을 나누어 받을 반복문

{

teamB.shape[j] = teamB.player[i + 1]; //모양을 저장한다

teamB.number[j] = teamB.player[i]; //숫자를 저장한다.

j++; //player에는 1,3,5...홀 수에는 모양이 짝수에는 숫자가 저장되어 있고 그것을 5개 나누어 받기 위함

}

mapping(teamA.player); //mapping함수 실행

mapping(teamB.player); //mapping함수 실행

sort(&teamA.shape, &teamA.number); //정렬 함수 실행

sort(&teamB.shape, &teamB.number); //정렬 함수 실행

teamA.straight = straight\_check(&teamA.number); //스트레이트 유무확인

teamB.straight = straight\_check(&teamB.number); //스트레이트 유무확인

teamA.flush = flush\_check(&teamA.shape); //플러쉬 유무확인

teamB.flush = flush\_check(&teamB.shape); //플러쉬 유무확인

teamA.high\_shape = high\_shape(teamA.shape); //패의 메인카드의 모양확인

teamB.high\_shape = high\_shape(teamB.shape); //패의 메인카드의 모양확인

teamA.result = calculation(teamA.flush, teamA.straight, teamA.player); //족보 최종 결정

teamB.result = calculation(teamB.flush, teamB.straight, teamB.player); //족보 최종 결정

re\_mapping(&teamA.player); //16진수로 할당한 값 10, J, Q, K, A로 변환하는 함수

re\_mapping(&teamB.player); //16진수로 할당한 값 10, J, Q, K, A로 변환하는 함수

if (select == 3) //선택된 select이 3이면 탈출

break;

else if (select == 2) //# 다시 #이 주석에 나오기 전까지 누가 어떤 패로 이겼는지 정하는 조건

printf("teamA:%s\nteamB:%s\n", teamA.player, teamB.player);

if (teamA.result > teamB.result)

{

result = 1;

final\_print(result, teamA.result, num2);

}

else if (teamA.result < teamB.result)

{

result = 2;

final\_print(result, teamB.result, num2);

}

else

{

if (teamA.high\_shape > teamB.high\_shape)

{

result = 1;

final\_print(result, teamA.result, num2);

}

else

{

result = 2;

final\_print(result, teamB.result, num2);

}

} //#

if (select == 2)

{

num2 = 0; //초기화한다.

break; //탈출

}

}

}

fclose(f); //반복문 나올 시 fclose로 f 메모리 영역 해제

}

return 0;

}

void mapping(char \* c) { //A, K, Q, J, 10은 16진수 E, D, C, B, A로 바꿔준다.

int i = 0;

for (i = 0; i < 5; i++)

{

if (c[2 \* i] == 'A')

c[2 \* i] = '0xE';

else if (c[2 \* i] == 'K')

c[2 \* i] = '0xD';

else if (c[2 \* i] == 'Q')

c[2 \* i] = '0xC';

else if (c[2 \* i] == 'J')

c[2 \* i] = '0xB';

else if (c[2 \* i] == '0')

c[2 \* i] = '0xA';

else;

}

}

void re\_mapping(char \*c ) { //mapping함수에서 16진수로 변환한 걸 다시 바꾼다.

int i = 0;

for (i = 0; i < 5; i++)

{

if (c[2 \* i] == 'E')

c[2 \* i] = 'A';

else if (c[2 \* i] == 'D')

c[2 \* i] = 'K';

else if (c[2 \* i] == 'C')

c[2 \* i] = 'A';

else if (c[2 \* i] == 'B')

c[2 \* i] = 'J';

else if (c[2 \* i] == 'A')

c[2 \* i] = 'T';

else;

}

}

void sort(char \*shape, int \*number) //숫자를 기준으로 오름차순 정렬한다.(후에 페어 계산을 위해)

{

int aa, i, j;

char bb;

for (i = 0; i < 4; i++) //버블정렬 숫자에 맞게

{

for (j = i + 1; j < 5; j++)

{

if (number[i] > number[j])

{

aa = number[i]; //카드 숫자와

number[i] = number[j];

number[j] = aa;

bb = shape[i]; //카드 문양이 같이 움직인다.

shape[i] = shape[j];

shape[j] = bb;

}

}

}

}

int high\_shape(char shape[5]) //만약 같은 숫자 페어로 비겼을시 실행되어 모양을 비교한다.

{

if (shape[5] = 'S')

{

return 4;

}

else if (shape = 'H')

{

return 3;

}

else if (shape = 'C')

{

return 2;

}

else {

return 1;

}

}

int straight\_check(int num[]) //정렬되어 있는 상태이므로 +1씩 하면서 스트레이트의 유무를 확인한다.

{

int i;

int c = 1;

for (i = 0; i < 4; i++) //숫자가 연속적인가?

{

if (num[i] + 1 != num[i + 1])

if (num[i + 1] == 14 && num[i] == 5)

c++;

else

break;

c++;

}

}

int flush\_check(char color[]) //color가 다 똑같지 않으면 플러쉬는 존재하지 않는다.

{

int i, c = 1;

for (i = 1; i < 5; i++)

{

if (color[0] != color[i])

break;

c++;

}

return c;

}

int calculation(int col, int li, char card[11])

//스트레이트, 플러쉬계산한 것도 매개변수로 들어와 최종 족보를 숫자로 정한다.

{

if (col == 5 && li == 5) //1. 카드 5장이 모두 같은 색이면서 숫자가 연속적일 때

if (card[8] == 14) { //로티플

return 1000 + card[8];

}

else { //스티플

return 900 + card[8];

}

else if (col == 5) //4. 카드 5장이 모두 같은 색일 때 (플러쉬)

return 600 + card[8];

else if (li == 5) //5. 카드 5장의 숫자가 연속적일 때 (스트레이트)

return 500 + card[8];

else

{

int num = 1;

int num2 = 1;

int plus = 0;

int count = 0;

int i, j;

for (i = 0; i < 4; i++) //한 번 검사했던 경우를 겹치지 않게 검사하고 모든 조건을 검사하는 알고리즘

{

for (j = 0; j < 4 - i; j++)

{

if (card[2 \* i] == card[2 + (2 \* j) + (2 \* i)])

{

count++;

plus = card[2 + (2 \* j) + (2 \* i)];

}

}

}

if (count == 4) //포카드와 풀하우스는 정렬상태기준 4번으로 카운트가 같음

{

if (card[2] == card[6])

{ //포카드

num = 4;

plus = card[4];

}

else

{

num = 3; //풀 하우스

num2 = 2;

plus = card[4];

}

}

else if (count == 2) //트리플은 정렬기준 3만 나오지만 투페어는 예외가 있어 카운트3과 2에 나눠주었다.

{

if (card[0] == card[2] && card[0] == card[4])

{//트리플

num = 3;

num2 = 1;

plus = card[4];

}

else if (card[2] == card[4] && card[2] == card[6])

{//트리플

num = 3;

num2 = 1;

plus = card[4];

}

else if (card[4] == card[6] && card[4] == card[8])

{//트리플

num = 3;

num2 = 1;

plus = card[4];

}

else

{//투페어

num = 2;

num2 = 2;

plus = card[6];

}

}

else if (count == 1) //원페어

{

num = 2;

num2 = 1;

}

else //탑

{

num = 1;

plus = card[8];

}

if (num == 4) //2. 카드 4장의 숫자가 같을 때

return 800 + plus;

else if (num == 3)

{

if (num2 == 2) //3. 카드 3장의 숫자가 같고 나머지 2장도 숫자가 같을 때

return 700 + plus;

else //6. 카드 3장의 숫자가 같을 때

return 400 + plus;

}

else if (num == 2)

{

if (num2 == 2) //7. 카드 2장의 숫자가 같고 또 다른 2장의 숫자가 같을 때

return 300 + plus;

else //8. 카드 2장의 숫자가 같을 때

return 200 + plus;

}

else //어떤 경우도 해당하지 않을 때

return 100 + plus;

}

}

void final\_print(int result, int num1, int num2)

//result에 따라 누가 이겼는지 어떤 카드로 이겼는지.

{

if (result == 1)

{

if (num2 == 0) //select =1일때 사용

printf("Team A wins \n");

else

{// num1은 calculation함수에서 반환받은 값.

if (num1 / 100 == 10)

printf("로얄 스트레이트 플러쉬로 A가 이겼습니다.\n");

else if (num1 / 100 == 9)

printf("스트레이트 플러쉬로 A가 이겼습니다.\n");

else if (num1 / 100 == 8)

printf("포카드로 A가 이겼습니다.\n");

else if (num1 / 100 == 7)

printf("풀하우스로 A가 이겼습니다.\n");

else if (num1 / 100 == 6)

printf("플러쉬로 A가 이겼습니다.\n");

else if (num1 / 100 == 5)

printf("스트레이트로 A가 이겼습니다.\n");

else if (num1 / 100 == 4)

printf("트리플로 A가 이겼습니다.\n");

else if (num1 / 100 == 3)

printf("투페어로 A가 이겼습니다.\n");

else if (num1 / 100 == 2)

printf("원페어로 A가 이겼습니다.\n");

else

printf("탑으로 A가 이겼습니다.\n");

}

}

else

{

if (num2 == 0) //select =1일때 사용

printf("Team B wins \n");

else

{ // num1은 calculation함수에서 반환받은 값.

if (num1 / 100 == 10)

printf("로얄 스트레이트 플러쉬로 B가 이겼습니다.\n");

else if (num1 / 100 == 9)

printf("스트레이트 플러쉬로 B가 이겼습니다.\n");

else if (num1 / 100 == 8)

printf("포카드로 B가 이겼습니다.\n");

else if (num1 / 100 == 7)

printf("풀하우스로 B가 이겼습니다.\n");

else if (num1 / 100 == 6)

printf("플러쉬로 B가 이겼습니다.\n");

else if (num1 / 100 == 5)

printf("스트레이트로 B가 이겼습니다.\n");

else if (num1 / 100 == 4)

printf("트리플로 B가 이겼습니다.\n");

else if (num1 / 100 == 3)

printf("투페어로 B가 이겼습니다.\n");

else if (num1 / 100 == 2)

printf("원페어로 B가 이겼습니다.\n");

else

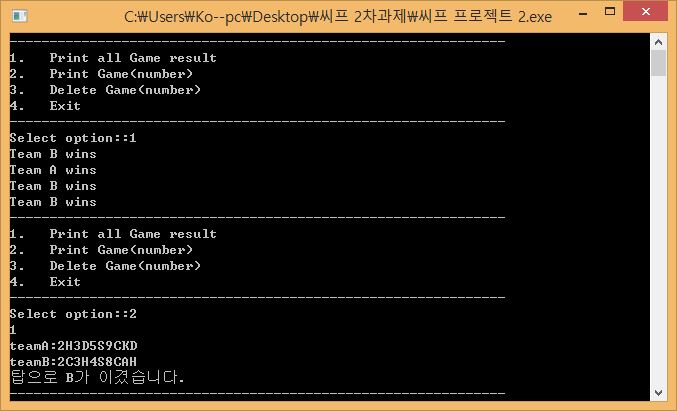
printf("탑으로 B가 이겼습니다.\n");

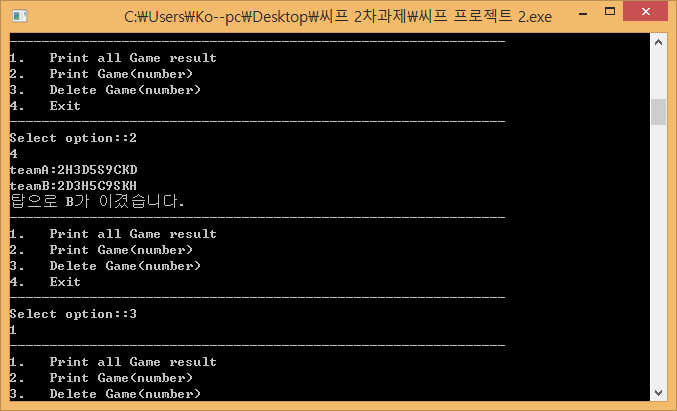
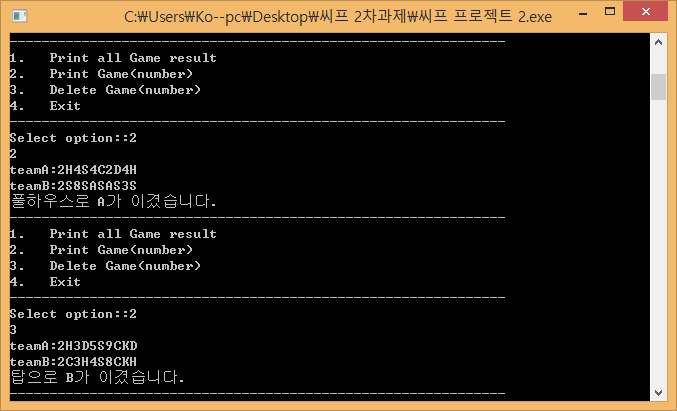
}

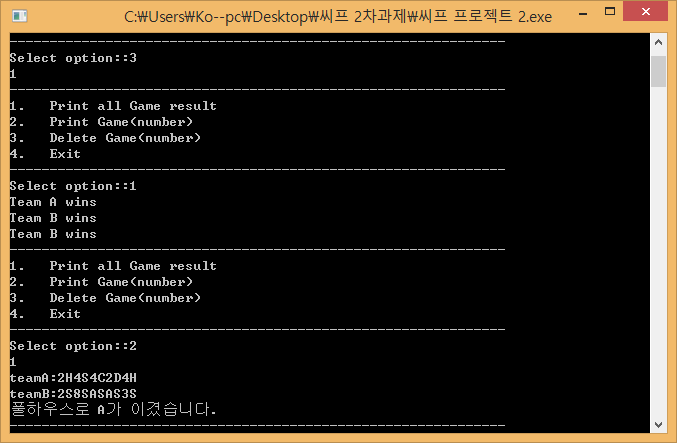
}

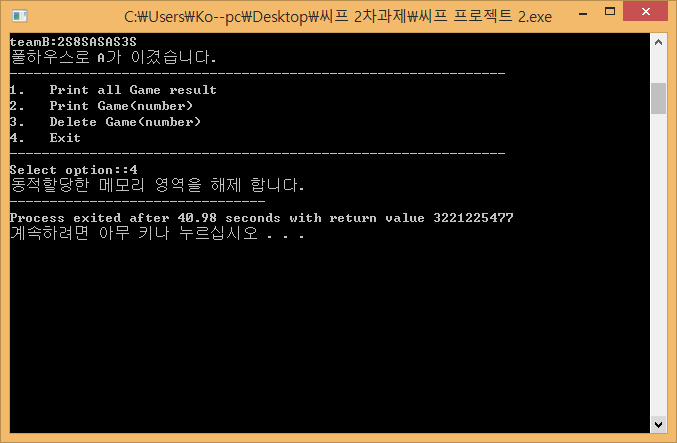
}

4. 실행결과









5. 고찰

-포커의 경우 숫자의 종류 13가지 문양의 종류 4가지 총 52장의 카드로 진행되는 게임이므로 숫자를 나타내기 위해서는 13진수이상을 표현할 수 형태로 저장이 되어야 한다. C언어에서는 16진수형태가 대표적인 예로, (문제상에서 10 = 0으로 표현한다고 되어 있음.) 이렇게 해야 하는 이유는 문제상에서 카드를 받는 변수를 1차원 배열 [10]으로 할당하라고 적혀있기 때문이다.

결론: 0 = 0xA, J = 0xB, Q = 0xC, K = 0xD, A = 0xE로 변환해서 한 글자씩 들어갈 수 있게 해주었다.