CSED101. Programming & Problem solving

Fall, 2018

Programming Assignment #2

(60 points)

신동민(xiaofo09@postech.ac.kr)

- ***Due***: **2018.10.19 23:59**
- ***Development Environment:*** **Windows Visual Studio 2017**

- ***제출물***
  - **C Code file (.c)**
    - 확장자를 포함한 소스파일의 이름은 **assn2.c**로 할 것.
      - 파일명 양식을 미 준수 하거나 프로젝트 폴더를 통째로 제출 할 시 감점
    - 프로그램의 소스 코드를 이해하기 쉽도록 반드시 **주석**을 붙일 것.
  - **보고서 파일** (.doc(x) or .hwp) 예) assn2.doc(x) 또는 assn2.hwp
    - AssnReadMe.pdf 를 참조하여 작성할 것.
    - **프로그램 실행화면을 캡쳐하여 보고서에 포함시키고, 간단히 설명할 것**
    - **명예서약(Honor code)**: 표지에 다음의 내용을 포함한다. "나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다." 보고서 표지에 명예서약이 없는 경우는 과제를 제출하지 않은 것으로 처리한다.
    - 소스코드와 보고서 파일을 LMS를 이용하여 제출한다.
  - **LMS에 제출할 때는 소스파일과 보고서를 압축하지 말고 각각 업로드 할 것.**

- ***주의사항***
  - 문제의 요구사항을 반드시 지킬 것.
  - 컴파일 & 실행이 안되면 무조건 0점 처리된다.
  - 하루 late시 20%가 감점되며, 3일 이상 지나면 받지 않는다. (0점 처리)
  - 부정행위에 관한 규정은 POSTECH 전자컴퓨터공학부 학부위원회의 'POSTECH 전자컴퓨터공학부 부정행위 정의'를 따른다. (LMS의 과목 공지사항의 제목 [document about cheating]의 첨부파일인 disciplinary.pdf를 참조할 것.)
  - 이번 과제에서는 추가 기능 구현에 대한 추가 점수는 없습니다.

■ Problem: E-Card

(목적)

이번 과제를 통하여 조건문, 반복문, 사용자 정의 함수 및 라이브러리 함수 사용법을 익힌다.

(주의사항)

- 이번 과제는 함수를 정의하고 사용하는 방법을 익히는 문제이므로 사용자 정의 함수를 사용하지 않고, main함수에 모든 기능을 구현한 경우 감점 처리 함. (반드시 정의해서 사용해야 할 사용자 정의 함수가 설명되어 있으니 확인한 후, 구현하도록 한다. 그 외의 필요한 함수를 정의해서 사용할 수 있다.)
- 프로그램 구현 시, main() 함수를 호출하여 사용하지 않는다. 즉, 소스 코드 내에 main(); 이라고 호출하지 않는다.
- 전역 변수, 배열 및 goto 문은 사용할 수 없으며, 포인터의 경우 수업시간에 다룬 내용에 한에서 사용이 가능하다.
- 사용자로부터 메뉴 선택, 카드 패 선택, 카드 선택을 위해 숫자를 입력 받을 때 정수 외의 입력에 대해서는 고려하지 않아도 된다.
- **문제의 출력 형식은 채점을 위해 아래의 실행 예시와 최대한 비슷하게 작성해 주세요.**

(설명)

본 게임은 "도박묵시록 카이지" 애니메이션에 등장하는 E-Card 게임을 기반으로 한다. 플레이어는 황제패와 노예패 중 하나를 선택할 수 있으며, 아래에 기술되어 있는 정해진 규칙에 따라 게임을 진행하게 된다.

- 각 카드패는 다음과 같이 구성된다.
  - 황제패 = 황제 카드 1장 + 시민 카드 4장
  - 노예패 = 노예 카드 1장 + 시민 카드 4장

- 게임 규칙은 다음과 같다.
  1. 플레이어는 황제패와 노예패 중 하나를 선택한다.
  2. 상대방(컴퓨터)은 남은 패를 가져간다.
  3. 플레이어와 상대방은 각자 자신의 패에서 카드를 한 장 선택한다.
  4. 선택한 카드를 서로 비교하여 승패를 결정한다.
  5. 승패가 결정되었다면 종료, 비겼을 경우에는 3번으로 돌아가 반복한다.

- 각 카드의 관계는 다음과 같다.
  - 황제 > 시민 > 노예 > 황제

- 플레이어가 만약 황제패를 선택한다면, 다음과 같은 경우가 있다.
  - ➢ 플레이어가 황제 카드를 고르고, 상대방은 시민 카드를 고르는 경우
    - 황제가 시민을 이기므로, 플레이어의 승리와 함께 게임 종료
  - ➢ 플레이어가 황제 카드를 고르고, 상대방은 노예 카드를 고르는 경우
    - 노예가 황제를 이기므로, 플레이어의 패배와 함께 게임 종료
  - ➢ 플레이어가 시민 카드를 고르고, 상대방도 시민 카드를 고르는 경우
    - 무승부로 끝이 나며, 고른 카드는 소모되고 패에서 새로운 카드를 각자 다시 선택하여 비교
  - ➢ 플레이어가 시민 카드를 고르고, 상대방은 노예 카드를 고르는 경우
    - 시민이 노예를 이기므로, 플레이어의 승리와 함께 게임 종료

I. 초기 선택 메뉴

프로그램이 시작되면, 그림 1처럼 3가지 선택사항이 있는 메뉴를 화면에 출력하고 사용자로부터 3가지 선택사항 중 하나를 입력 받도록 한다.

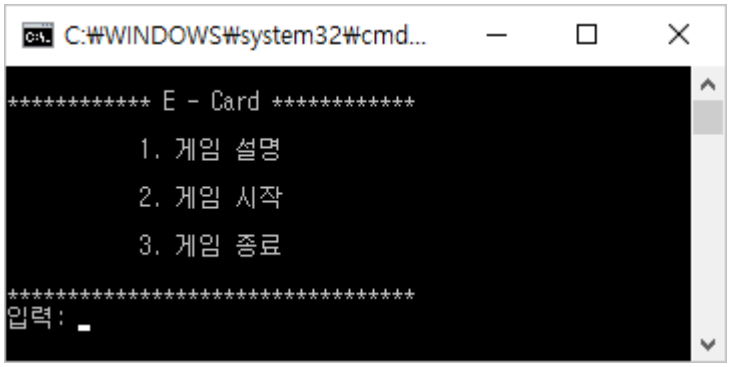

그림 1. 초기 선택 메뉴

- ➢ 초기 선택 메뉴는 main() 함수 내에서 아래의 사용자 정의 함수를 호출하여 출력하도록 하며, 반드시 switch-case 문을 사용하여 메뉴 선택을 하도록 한다.
  - **show_menu():** 초기 메뉴 화면을 출력하고, 사용자로부터 메뉴 번호를 입력 받아 그 번호를 반환

메뉴 입력 시, 숫자만 입력된다고 가정한다. 단, 1, 2, 3 외의 값을 입력할 경우, 그림 2와 같이 올바르게 입력해달라는 메시지를 출력하고 1초 뒤에 초기 선택 메뉴(그림 1)를 다시 화면에 출력하도록 한다. 이때, 반드시 화면을 먼저 지운 후에 그림 1의 초기 선택 메뉴를 출력하도록 한다.
1초 딜레이 효과는 *Tips-지연 효과 주기(마지막 쪽)*, 화면을 지우는 방법은 *Tips-화면 지우기(마지막 쪽)*을 참고하여 구현한다.

그림 1에서 "3. 게임 종료" 메뉴를 선택하면, 그림 3과 같이 프로그램 종료메시지를 출력 후 프로그램을 종료하면 된다.

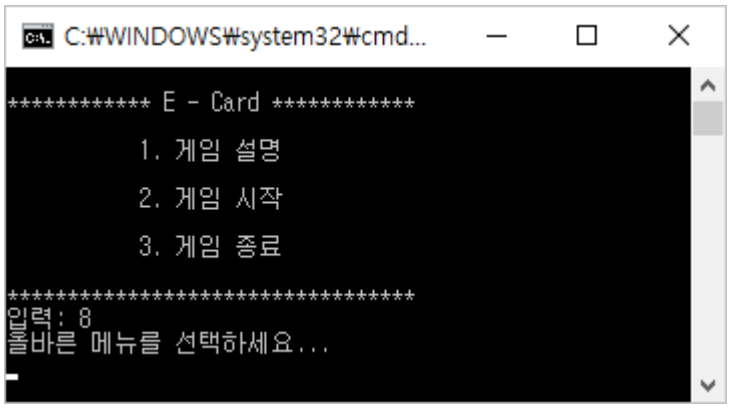

그림 2. 비정상적인 메뉴 선택 시

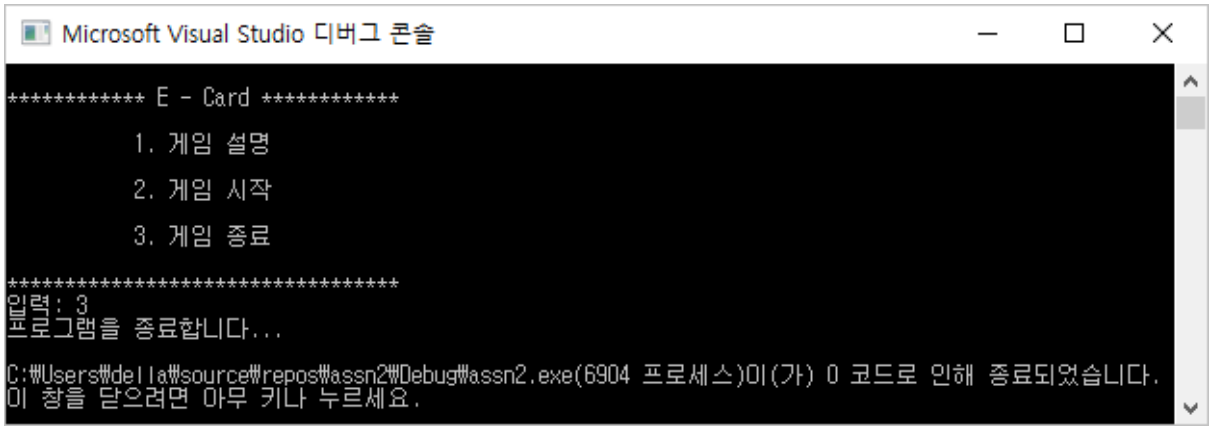

그림 3. 메뉴 "3. 게임 종료" 선택 시의 화면

그림 1에서 "1. 게임 설명" 메뉴를 선택하면 화면이 지워진 후, 아래처럼 게임 설명이 출력된다. 그 아래에 "메뉴로 돌아가려면 Enter키를 입력하세요..."라는 메시지가 출력되며 사용자 입력을 기다린다. 사용자가 Enter키를 입력하면 화면 지우기 후 그림 1의 메뉴가 출력된다.

- **print_tutorial():** 아래의 tutorial을 출력하는 함수

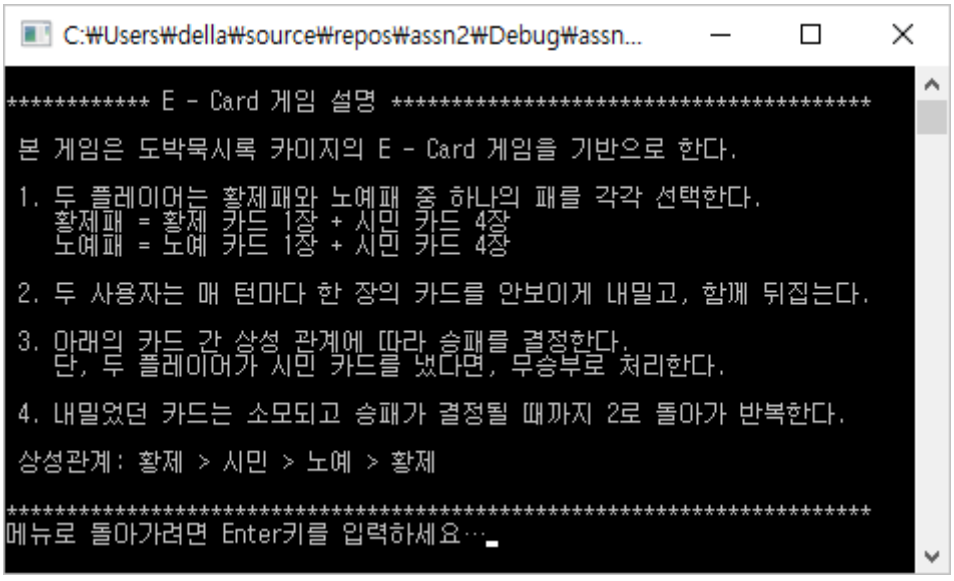

그림 4. 메뉴 "1. 게임 설명" 선택 시의 화면

II. 게임 시작

그림 1에서 "2. 게임 시작" 메뉴를 선택하면 **start_game()** 함수가 호출되며 게임이 시작된다. 게임의 모든 진행 과정은 start_game() 함수 내에서 수행되어야 한다.

아래 설명에서 반드시 정의해서 사용해야 할 사용자 정의 함수가 설명되어 있으니 확인한 후, 구현하도록 한다. 설명에서 구현하라고 한 함수 외에 필요한 함수를 정의하고 사용할 수 있다.

1. 카드 패 선택

게임이 시작되면 제일 먼저 화면을 지운 후, 그림 5와 같이 사용자의 패 선택 메뉴를 화면에 출력하고 사용자로부터 어떤 패를 선택할 것인지 입력 받게 된다.

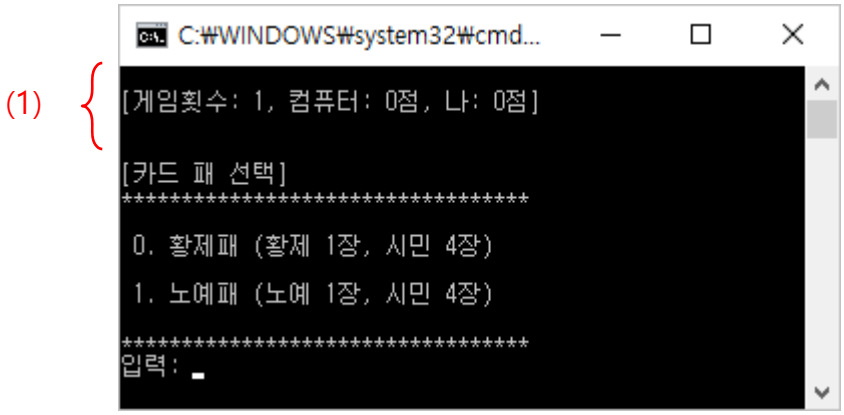

그림 5. 사용자의 패 선택 메뉴

그림 5의 화면에서 사용자가 정해진 메뉴(0과 1)외의 다른 메뉴를 입력할 경우, 그림 2의 상황과 비슷하게 올바른 메뉴를 선택해달라는 메시지를 출력하고, 1초 뒤에 그림 5를 다시 화면에 출력하도록 한다. 이때, 화면을 먼저 지우고 메뉴를 출력하도록 한다. 사용자가 0을 입력하면 사용자의 패는 황제패가 되며, 1을 입력하면 노예패가 된다.

화면 상단(그림 5의 (1))에는 게임 횟수, 컴퓨터 점수, 나(사용자)의 점수가 출력된다. 첫번째 게임이며, 모든 플레이어가 0점에서 시작함을 볼 수 있다.

2. 카드 선택

사용자가 패를 선택하고 나면 화면을 먼저 지우고 그림 6 또는 그림 7과 같이 카드가 놓인 테이블을 출력한다. 그리고 자신의 패에 있는 카드 중 어떤 것을 내밀지 선택해야 한다. 그림 6을 보면 알파벳 E와 C를 볼 수 있는데, 'E'는 황제(**E**mperor)를 의미, 'C'는 시민(**C**itizen)을 의미한다. 만약 사용자가 황제패가 아닌 노예패를 선택했다면(그림 6) 'E'가 있는 자리에는 'S'가 오게 된다. 'S'는 노예(**S**lave)를 의미한다. 그림 6에 대한 더욱 자세한 설명은 아래와 같다.

- 그림 6의 (1)
  - 화면의 상단에는 게임 횟수, 현재까지의 획득한 컴퓨터의 점수와 사용자의 점수가 출력된다.

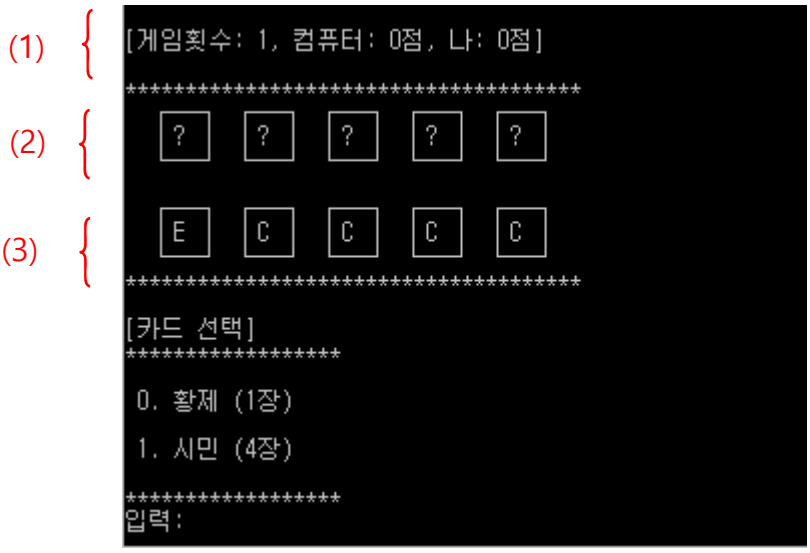

그림 6. 사용자의 카드 선택 메뉴 (황제패 선택 시)

- 그림 6의 (2)
  - 상대방(컴퓨터)의 카드패
  - 상대방의 카드패는 모두 '?'로 표시하도록 한다.
- 그림 6의 (3)
  - 사용자(본인)의 카드패
  - 게임의 규칙과는 상관없지만 사용자의 카드를 화면에 표시할 때는 순서를 지키도록 한다. 패에 따라 황제(E) 또는 노예(S) 카드를 가장 왼쪽에 위치시키고, 그림에서 볼 수 있듯이 시민(C) 카드 4장을 오른쪽으로 위치시키도록 한다. 현재는 사용자가 황제패를 고른 상황이므로 황제(E) 카드가 가장 왼쪽에 있으며, 시민(C) 카드 4장이 오른쪽으로 위치해있다. 만약 사용자가 노예패를 골랐다면(그림 7) 노예(S) 카드가 가장 왼쪽에 위치하며, 시민(C) 카드 4장이 오른쪽으로 위치한다.

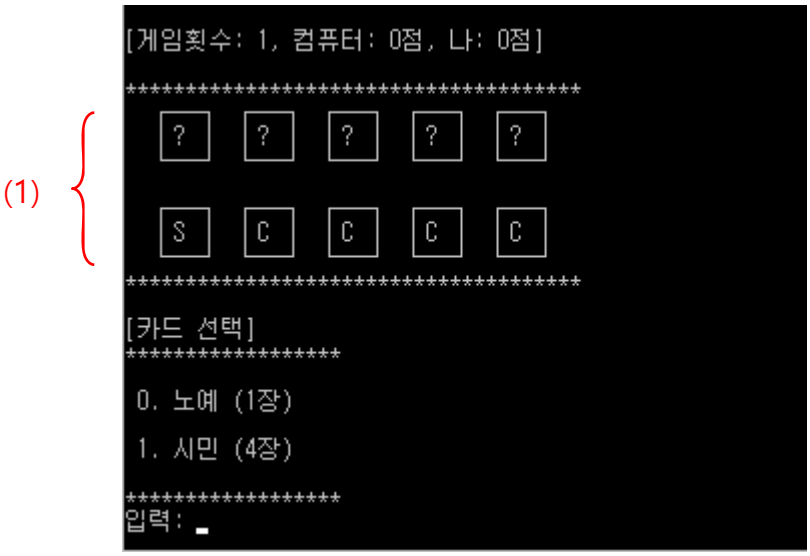

그림 7. 사용자의 카드 선택 메뉴 (노예패 선택 시)

- 카드가 놓인 테이블(그림 7의 (1))을 출력하기 위해서는 다음과 같은 사용자 정의 함수를 정의하고 호출하도록 한다.
  - **show_table()**: 사용자가 선택한 카드패와 남은 시민 카드 개수를 매개변수로 전달받아 상대방과 사용자의 카드들을 화면에 출력한다. (그림 6, 7, 9 참조)
    카드 출력을 위한 ㄱ ㅡ ㄱ | ㄴ ㄱ 문자('ㅂ'키 입력 후, 한자키를 눌러서 해당 문자 선택)를 사용하여 적절히 출력한다.

① 사용자의 카드 선택

그림 6 또는 7의 화면에서 사용자는 어떤 카드를 내밀지 선택한다. 황제패를 선택했다면 그림 6의 화면이 나타나야 하며, 노예패를 선택했다면 그림 7의 화면이 나타나야 한다. 사용자가 정해진 메뉴 외의 다른 메뉴를 입력할 경우, 앞의 입력 화면들처럼 올바른 메뉴를 선택해달라는 메시지를 출력하고 1초 뒤에 화면을 먼저 지운 후 그림 6 또는 7을 다시 화면에 출력하도록 한다.

- **get_user_card()**: 사용자의 카드패와 남은 시민 카드의 개수 등을 매개변수로 전달받아 사용자의 카드패에 따라 그림 6 또는 그림 7과 같이 사용자와 컴퓨터의 카드들과 카드 선택 메뉴를 화면에 출력한다. 그리고 사용자로부터 어떤 카드를 선택할지 입력 받아서 반환한다. 카드들을 출력할 때는 위에서 언급한 show_table() 함수를 이용하도록 한다 (show_table() 함수를 get_user_card() 함수 내에서 호출).

② 컴퓨터의 카드 선택

게임을 위하여 컴퓨터가 자신이 가진 카드 중에 하나를 선택한다. 카드 선택을 위해 rand() 함수를 사용하도록 한다. 즉, 컴퓨터는 카드를 선택할 때마다 임의의 카드를 선택하도록 해야 한다. (힌트. 컴퓨터가 가지고 있을 수 있는 카드는 2종류 이므로 난수 0이나 1을 발생시키면 된다.)

- **get_computer_card()**: 컴퓨터가 가진 카드 중에 1장을 선택하여 반환하는 함수

3. 승패 판정

사용자와 컴퓨터가 게임의 승패를 결정하기 위하여 각각 보유한 카드 중에 한 장씩을 선택하였다. 선택한 카드를 출력하고 승패를 판정한다.

① 선택한 카드 출력 및 승패 결과 출력 - 그림 8의 (1)

- 사용자 및 컴퓨터가 어떤 카드를 선택했는지 표시하도록 한다. 그림 8의 경우, 사용자는 시민 카드(1 입력)를 선택했기 때문에, 선택 결과에 시민 카드가 출력되는 것을 볼 수 있으며 컴퓨터도 시민 카드를 선택하였음을 볼 수 있다.
- 두 카드를 비교하여 누가 승자인지 결정한다. 이 때, 결과는 반드시 사용자를 기준으로 출력한다. 즉, "나의 승리" 또는 "나의 패배"로 표시한다.
  무승부인 경우는 "무승부"로 표시한다.

➢ **compare_card()**: 두 카드를 비교하여 승부 결과(승/패/무승부)를 반환. (적절한 값을 정하여 반환하면 됨)

- 그림 8의 경우는 플레이어 모두 시민 카드를 선택한 경우로 무승부이다. 무승부의 결과를 출력한 후, 다음 턴을 진행하기 위해 사용자로부터 Enter키를 입력 받도록 한다.

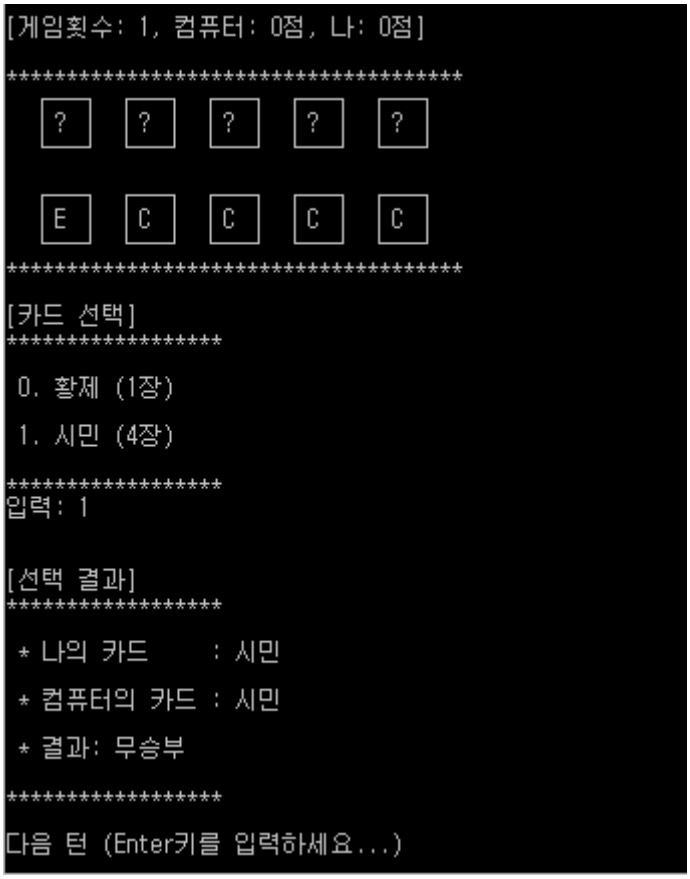

그림 8. 사용자가 카드를 선택한 후의 메뉴 (무승부)

- 그림 8에서 Enter키를 입력 하면, 화면 지우기 후 그림 9의 화면이 출력되도록 한다. 사용자와 컴퓨터의 카드가 1장씩 소모된 것을 볼 수 있다.

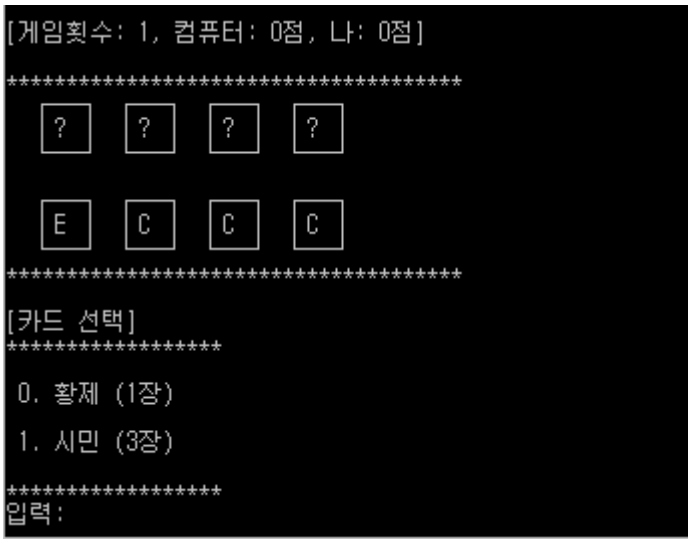

그림 9. 무승부 이후의 턴 화면

- 그림 9에서 볼 수 있듯이 카드는 반드시 오른쪽 카드부터 1장씩 없애도록 한다. 카드가 1장씩 없어진다는 점을 제외하고는 사용자의 카드 선택부터 다시 시작되도록 한다
- 그림 10의 (1) 은 사용자가 황제 카드(0 입력)를 선택했기 때문에, 선택 결과에 황제 카드가 출력되는 것을 볼 수 있다. 컴퓨터는 시민 카드를 선택했고, 그 승부의 결과로 "나의 승리"가 출력되었음을 볼 수 있다.

그림 10. 사용자가 카드를 선택한 후의 메뉴 (사용자의 승리)

② 현재 점수 계산 및 출력 - 그림 10의 (2)

- 승리나 패배 시 점수를 얻거나 잃게 된다. 단, 어느 패를 선택하느냐에 따라 승리/패배 시 득점/실점의 수치가 다르다.
- 플레이어가 황제패를 선택하였을 경우 노예패를 선택한 상대방보다 이길 수 있는 확률이 높기 때문에 승리 시 200점 득점, 패배 시 500점을 실점시킨다.

- 플레이어가 노예패를 선택하였을 경우 황제패를 선택한 상대방보다 이길 수 있는 확률이 낮기 때문에 승리 시 700점 득점, 패배 시 300점을 실점시킨다.
- 현재 게임의 결과 점수까지 반영한 총 점수를 출력시키도록 한다.
  - **calc_score()**: 플레이어의 카드패와 플레이어의 승부 결과를 전달 받아, 점수가 얼만큼 득점이 되어야 하는지 또는 실점이 되어야 하는지 해당 값을 반환. (만약, 황제패를 선택해서 이겼다면 200점이 반환된다.) 반환 받은 이 값을 이용하여 사용자와 컴퓨터의 최종 점수를 갱신한다.
- 그림 10의 (2)

  황제패를 가진 사용자가 승리한 경우로 사용자의 게임 점수는 200점을 획득했고, 컴퓨터는 300점을 잃었다.

4. 게임 종료

게임의 승패가 결정 된 후, 게임이 끝나는 경우는 아래의 2가지 경우이다.

① 사용자 선택

- 게임의 승패가 결정된 후에, 그림 10과 같이 "게임을 계속하겠습니까? (y/n): "라는 메시지가 출력되고 사용자 입력을 기다린다.
- 이 때, y를 입력하면 화면 지우기 후 그림 11처럼 현재 점수를 가지고, 카드 패 선택부터 게임을 이어나간다. 화면의 상단의 정보를 보면, 이 게임이 2번째 게임이며, 컴퓨터와 사용자의 현재 게임 점수를 확인 할 수 있다.

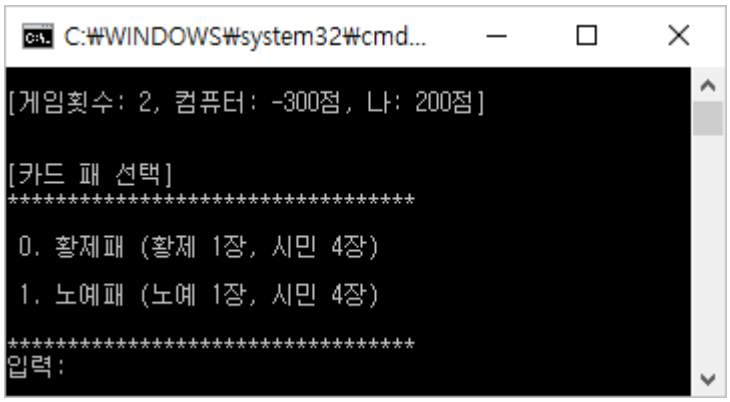

그림 11. 그림 10에서 y를 입력하여 카드 게임을 계속 진행하는 화면

- n을 입력하면, 화면 지우기 후 그림 1의 초기 선택 메뉴로 돌아간다.
- y 또는 n을 입력할 때 대소문자와 관계 없이 입력 가능하도록 한다. 즉, 입력 가능한 값으로 y/Y/n/N이 있다. 이외의 입력은 고려하지 않는다.

② 사용자 또는 컴퓨터 중에서 2000점에 먼저 도달한 경우

- 사용자 또는 컴퓨터 중에서 획득 점수가 2000점에 먼저 도달하는 경우, 그림 12의 화면

과 같이 출력한다. 사용자가 최종 목표 점수에 먼저 도달했다면 "당신의 승리 !"를 출력하도록 하고, 컴퓨터가 최종 목표 점수에 먼저 도달했다면 "당신의 패배 !"를 출력하도록 한다. Enter키를 입력 하면 화면 지우기 후 그림 1의 초기 선택 메뉴로 돌아간다.

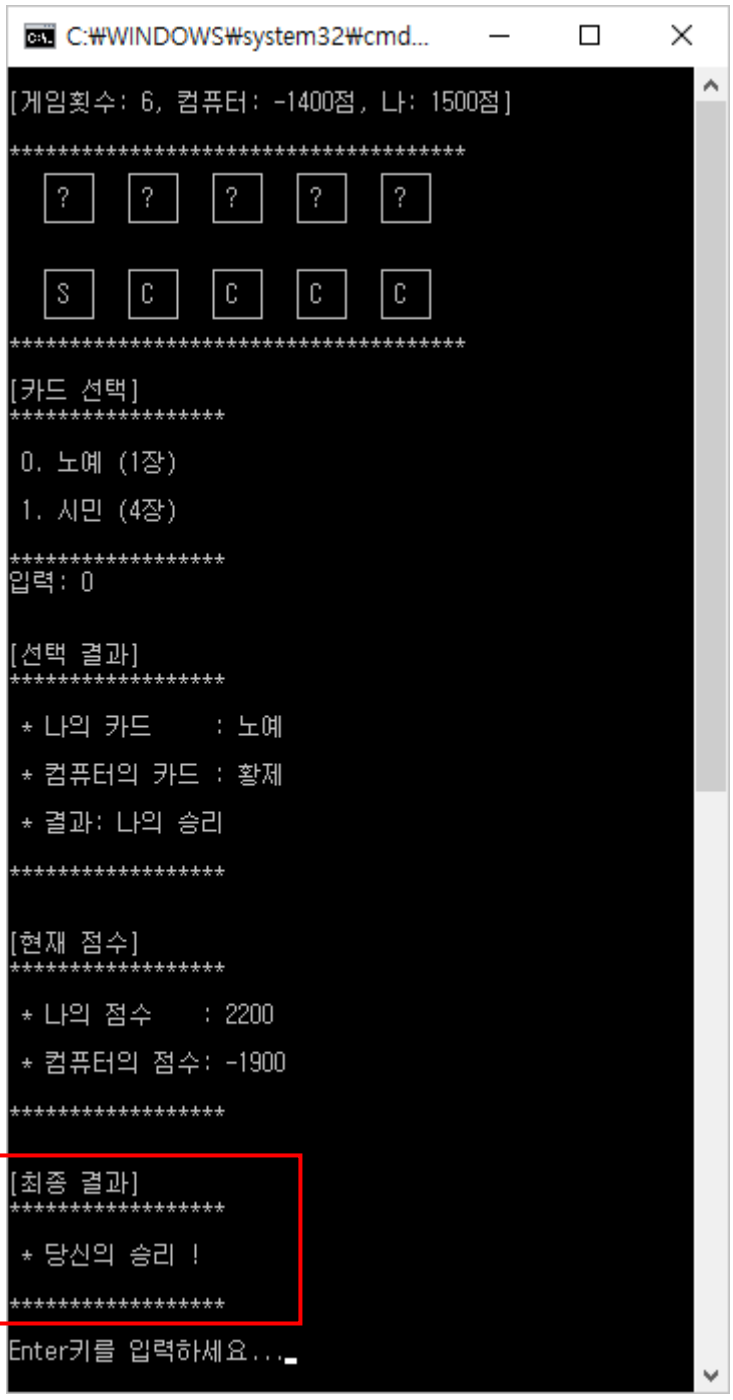

그림 12. 사용자가 최종 점수 2000점 이상에 먼저 도달한 화면

그림 1의 초기 선택 메뉴에서 "2. 게임 시작"을 선택하면, 게임 횟수는 1, 컴퓨터 및 사용자의 점수는 0에서부터 다시 시작된다. (초기 선택 메뉴에서 게임 시작을 하면 그림 5와 같이 항상 게임 횟수는 1, 플레이어들의 점수는 0으로 시작된다.)

Tips.

- ***화면 지우기***

화면을 지우고 싶을 경우, stdlib 헤더 파일을 포함 시킨 후 system() 함수를 사용하도록 한다. system() 함수의 매개변수로 "cls"를 넘겨주면 된다. 아래의 소스 코드를 컴파일하여 실행하면 콘솔 화면에 "Erase me !" 문자열이 출력된다. Enter키를 입력하면 화면이 지워진 후에 "Erased" 문자열이 출력되는 것을 볼 수 있다.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
	char c;

	printf("Erase me !₩n");
	scanf("%c", &c);
	system("cls");
	printf("Erased₩n");

	return 0;
}
```

- ***지연 효과 주기***

지연 효과는 Sleep()함수를 통해 구현 가능하다. 해당 함수는 인자로 지연시간을 millisecond 단위로 받으며, 사용하기 위해서는<Windows.h> 헤더파일을 포함해야 한다. 사용 예는 다음과 같다.

```
#include <stdio.h>
#include <Windows.h>
int main()
{
	printf("첫 번째 메시지 입니다₩n");
	Sleep(3000);
	printf("이 메시지는 3초뒤에 출력됩니다₩n");
	return 0;
}
```