CSED101. Programming & Problem solving
Spring, 2020

Programming Assignment #2
(75 points)

박해성(hea9549@postech.ac.kr)

■ *Due*: 2020.06.04 23:59

■ *Development Environment: Windows Visual Studio 2019*

■ *제출물*

- C Code files (assn2_1.c, assn2_2.c)
  - ➢ 프로그램의 소스 코드를 이해하기 쉽도록 반드시 **주석**을 붙일 것.
- **보고서 파일** (assn2.docx, assn2.hwp 또는 assn2.pdf)
  - ➢ AssnReadMe.pdf 를 참조하여 작성할 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.

■ *주의사항*

- 각 문제에 해당하는 요구사항을 반드시 지킬 것.
- 모든 문제의 출력 형식은 아래의 예시들과 동일해야 하며, 같지 않을 시는 감점이 된다.
- 문제에 제시되어 있는 파일이름으로 제출 할 것. 그 외의 다른 이름으로 제출하면 감점 또는 0점 처리된다.
- 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.

[들어가기 전]

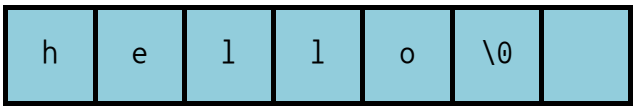
1. 문자열(string)

- 연속된 문자들로 C 언어에서 문자열 앞 뒤에 " "를 이용한다.
- char 형의 1차원 배열을 이용하여 문자열을 저장한다.
- 배열에 문자열을 저장할 때는 끝을 NULL 문자 ('\0')를 넣어서 표시한다.

2. 선언

- **char str[] = "hello";**
  위와 같이 선언과 동시에 초기화를 하게 되면, 자동으로 문자열의 끝에 널문자가 추가된다.

  str



3. 입력과 출력

```
char str[100];
printf("Inut your email: ");
scanf("%s", str);
printf("%s", str);
```

<결과화면> (아래의 파란 글자는 사용자 입력입니다.)

```
Input your email: hea9549@postech.ac.kr
hea9549@postech.ac.kr
```

■ Problem 1: HEA 로그인 시스템 (HEA login system) (30점)

[문제]

조교가 개발한 첨단 HEA 시스템에 로그인 기능을 더하려고 한다. 로그인을 하기 위해선 이메일 주소와 비밀번호를 입력해야 한다. 해당 시스템은 이메일 주소와 비밀번호를 입력 받아 이메일 형식이 맞는지 확인하고 일정한 규칙에 의해 생성된 비밀번호와 일치하는지 확인하는 프로그램이다. 텍스트 파일에 허용되는 이메일 도메인 주소를 나열하고, 입력된 이메일이 이메일 형식을 띠는지, 허용된 도메인 인지, 비밀번호가 맞게 입력됐는지 확인하는 프로그램을 작성해 보자.

[목적]

- 1차원 배열의 선언과 사용을 익힌다.
- 함수에서 1차원 배열을 매개변수로 사용하는 방법을 익힌다.
- 텍스트 파일을 통한 입력을 익힌다.

[주의사항]

(1) 소스코드는 "assn2_1.c"로 저장할 것.

(2) 보고서는 "assn2.docx", "assn2.hwp" 또는 "assn2.pdf"로 저장 한다. (보고서는 통합하여 작성)

(3) 전역 변수, goto 문, string 관련 함수, 구조체 등은 사용할 수 없으며, 포인터의 경우 수업 시간에 다룬 내용에 한해서 사용이 가능하다.

(4) string.h에 정의된 함수를 사용하지 않는다.
해당 기능이 필요한 경우 char 배열을 이용하여 직접 구현하여 사용한다.

(5) 모든 기능을 main 함수에서 구현한 경우 감점 처리한다. 기능적으로 독립됐거나 반복적으로 사용되는 기능은 사용자 함수를 정의해서 구현하여야 한다.

(6) 프로그램 구현 시, main() 함수를 호출하여 사용하지 않는다. 즉, 소스 코드 내에 main(); 이라고 호출하지 않는다.

(7) 문제의 출력 형식은 채점을 위해 아래의 실행 예시와 반드시 같게 작성해 주세요. '-'의 개수와 공백의 개수는 상관 없음. 불일치시 감점 있음.

[가정]

(1) 이메일
- 입력 이메일의 최대길이는 30 이라고 가정한다.
  #define 지시자를 이용하여 이메일의 최대길이를 정의하여 사용한다.
- 영문 소문자, 숫자, '@', '.'로만 구성

(2) 비밀번호
- 입력 비밀번호의 최대길이는 30 이라고 가정한다.
  #define 지시자를 이용하여 비밀번호의 최대길이를 정의하여 사용한다.
- 영문 소문자, 숫자로만 구성

(3) 모든 입력 값은 영문 소문자, 숫자, '@', '.'만 입력 받는다고 가정한다.

[함수 요구사항]

이 문제를 해결하기 위해 반드시 아래 기능에 해당하는 **사용자 정의 함수**를 정의하고 사용해야 한다. 아래 기능 외의 필요한 함수를 정의해서 사용할 수 있다.

- **make_password: 이메일로부터 패스워드를 생성하는 함수**
- **validate_email: 이메일로부터 형식이 일치하는지, 허용되는 도메인 인지 체크하는 함수**

[설명 및 실행 예시]

컴퓨터가 사용자로부터 이메일과 비밀번호를 입력 받아 입력 값을 확인한 후, 입력 값에 따라 적절한 결과 문구가 출력된다. 프로그램은 입력 -> 검사 -> 결과 출력 3단계로 이루어진다. 먼저 유효한 이메일인지 체크하고 유효한 이메일이라면 아래 ‘비밀번호 생성규칙‘에 맞게 만들어진 비밀번호로 입력됐는지 확인한다.

(1) 입력단계에서는 사용자로부터 30자 이내의 이메일과 비밀번호를 입력 받는다.
프로그램을 실행하면, 아래와 같이 출력 후, 사용자로부터 email을 입력 받기 위해 기다린다.

```
---LOGIN TO ACCESS HEA SYSTEM---

email :
```

그림 1. 초기화면 (1)

사용자가 email로 아래와 같이 hea9549@postech.ac.kr를 입력 후 엔터키를 누르면, 다음으로 password를 입력 받기 위해 아래와 같이 출력 후 기다린다.

```
---LOGIN TO ACCESS HEA SYSTEM---

email : hea9549@postech.ac.kr
password :
```

그림 2. 초기화면 (2)

(2) 검사 및 출력 단계
사용자가 이메일과 비밀번호를 입력 후 엔터키를 누르면, 컴퓨터가 사용자로부터 입력 받은 이메일과 비밀번호를 검사한다.
아래는 모든 요건을 만족하는 경우로, 로그인 성공 메시지를 출력 후, 프로그램을 종료한다.

```
---LOGIN TO ACCESS HEA SYSTEM---

email : hea9549@postech.ac.kr
password : hpeoas9t5e

--------------CHECKING--------------
SUCCESS! now you can access HEA system!

계속하려면 아무 키나 누르십시오 . . .
```

그림 3. 로그인 성공 예시

가. 이메일 검사
입력 받은 이메일 문자열을 다음과 같이 3가지 조건에 따라 검사 후, 조건에 맞춰 에러 메시지를 출력한다.

1. 이메일 형식이 올바르지 않음: 이메일 모양을 띄고있지 않을 경우(그림 4)
2. 도메인 형식이 올바르지 않음: 주소 부분의 형식이 올바르지 않을 경우(그림 5)
3. 허용되지 않은 도메인: 주소 부분이 허용되지 않는 도메인일 경우(그림 6)

A. 이메일 형식의 충족 조건은 ‘아이디@주소’이며,
주소의 조건은 naver(이름).com(최상위 도메인)과 같이 최소 하나 이상의 점(.)으로 이루어져야 하고 각 이름, 도메인들은 최소 1글자 이상이 되어야한다. 즉 hea9549@postech.ac.good.kr은 허용되지만, hea9549@.ac.kr, hea9549@postech...ac.kr, hea9549@postech.ac. 은 허용되지 않는다.

B. 주소는 ‘허용되는 도메인’에 속하는 이메일만 검사를 통과한다. 예를 들어, 허용되는 도메인이 ‘postech.ac.kr’ 일 경우 hea9549@postech.ac.kr 을 입력하면 통과, hea9549@daum.net 을 입력하면 검사를 통과하지 못한다.
‘허용되는 도메인’은 파일로 주어지며 5쪽을 참고하여 구현한다. 주어진 파일에 없는 도메인인 경우 검사를 통과하지 못한다.

```
---LOGIN TO ACCESS HEA SYSTEM---

email : hea9549
password : hpeoas9t5e

--------------CHECKING--------------
ERROR! invalid email format!

Do you want try again? (y/n)
```

그림 4. 이메일 - 형식 에러

```
---LOGIN TO ACCESS HEA SYSTEM---

email : hea9549@bbbb
password : hpeoas9t5e

--------------CHECKING--------------
ERROR! invalid domain format!

Do you want try again? (y/n)
```

그림 5. 이메일 - 도메인 형식 에러

```
---LOGIN TO ACCESS HEA SYSTEM---

email : hea9549@daum.net
password : hpeoas9t5e

--------------CHECKING--------------
ERROR! This domain is not allowed!

Do you want try again? (y/n)
```

그림 6. 이메일 - 허용되지 않은 도메인

그림 3과 같이 로그인 성공인 경우를 제외하고, 나머지 경우 즉 그림 4, 5, 6의 예시처럼, 다시 시도하겠냐는 문구를 출력한다. 사용자의 입력에 따라 다시 시도 또는 프로그램이 종료된다(그림 9).

- **<u>허용되는 도메인</u>**

  허용되는 도메인을 검사할 때, 파일 allowed_domain.txt 에 있는 도메인인지 검사를 하게 된다.

```
4
postech.ac.kr
gmail.com
naver.com
test.com
```

그림 7. 허용되는 도메인 파일 예시

A. 파일 allowed_domain.txt 의 구성은 그림 7과 같다.
B. 첫 번째 줄의 숫자는 전체 허용 도메인 수를 나타낸다.
C. 나머지 데이터들은 도메인 주소이며 줄(line)단위로 구분된다.
D. 도메인 최대 개수는 20개를 넘지 않으며, 도메인 주소의 길이는 20을 넘지 않는다. (단, 도메인 최대 개수와 주소 길이 부분은 구현 방법에 따라 사용하지 않을 수도 있다.)
E. 위의 파일 내용은 하나의 예제이며, 위와 같이 구성된 어떤 파일에 대해서도 처리할 수 있도록 구현해야 한다. 단, 도메인 주소 형식이 틀리거나 파일 구성에 대한

에러는 고려할 필요가 없다.

나. 비밀번호 검사

이메일 검사를 통과 한 경우, 입력된 비밀번호가 아래 ‘ 비밀번호 생성 규칙 ’ 에 맞게 생성된 비밀번호와 일치하는지 확인한다. 일치하지 않는 경우에는 그림 7과 같이 에러 메시지를 출력하며, 일치하는 경우에는 그림 3의 예시를 참조하여 구현한다.

- **비밀번호 생성 규칙**

비밀번호는 입력한 사용자의 이메일 주소를 이용해 생성한다.
이메일 주소는 크게 아이디 부분과 도메인 부분으로 나눌 수 있다.
(예: hea9549(아이디)@postech.ac.kr(도메인))
이때, 아이디 부분과 도메인 부분을 번갈아 가면서 한 글자씩 이용해 10자의 비밀번호를 만든다. 즉 hea9549@postech.ac.kr 의 이메일 경우 비밀번호는 hpeoas9t5e 이다.
만약 아이디와 도메인 합이 10글자가 넘지 않을 경우 부족한 공간은 0으로 채운다.
만약 아이디 또는 이메일 한쪽이 짧을 경우 나머지 공간은 아이디 또는 이메일로 나머지를 채운다.
(예: a@postech.ac.kr -> apostech.a)
(예: a@b.c -> ab.c000000)

```
---LOGIN TO ACCESS HEA SYSTEM---

email : hea9549@postech.ac.kr
password : csed101good

--------------CHECKING--------------
ERROR! invalid password!

Do you want try again? (y/n)
```

그림 8. 비밀번호 틀림

(3) 재입력

사용자가 입력한 이메일 또는 패스워드가 틀린 경우, 적절한 에러메시지 출력 후 다시 시도하겠냐는 문구를 출력한다. 사용자가 y를 선택하면 다시 이메일과 비밀번호를 입력 받아 로그인을 시도한다(그림 9). 사용자가 n을 선택하면 프로그램을 종료한다(그림 10).

```
---LOGIN TO ACCESS HEA SYSTEM---

email : hea9549@postech.kr
password : hpeoas9t5e

--------------CHECKING--------------
ERROR! This domain is not allowed!

Do you want try again? (y/n) y

---LOGIN TO ACCESS HEA SYSTEM---

email:
```

그림 9. 재시도 - y 입력

```
---LOGIN TO ACCESS HEA SYSTEM---

email : hea9549@postech.kr
password : hpeoas9t5e

--------------CHECKING--------------
ERROR! This domain is not allowed!

Do you want try again? (y/n) n

계속하려면 아무 키나 누르십시오 . . .
```

그림 10. 재시도 - n 입력

■ Problem 2: 카드 뒤집기 게임 (45점)

[문제]

- 2차원 배열을 이용하여 카드 뒤집기 게임을 구현해 보자.

[목적]

- 2차원 배열의 선언과 사용을 익힌다
- 함수에서 2차원 배열을 매개변수로 사용하는 방법을 익힌다

[주의사항]

(1) 소스코드는 "assn2_2.c" 로 저장할 것.

(2) 보고서는 "assn2.docx", "assn2.hwp" 또는 "assn2.pdf"로 저장 한다. (보고서는 통합하여 작성)

(3) 이 과제는 2차원 배열을 선언하고 사용하기 위한 과제이므로, 카드 게임판은 반드시 2차원 배열로 선언 후 사용해야 한다. (마지막 페이지의 힌트를 참고할 것)

(4) 전역 변수, goto 문, string 관련 함수, 구조체는 사용할 수 없으며, 포인터의 경우 수업시간에 다룬 내용에 한해서 사용이 가능하다.

(5) 모든 기능을 main 함수에서 구현한 경우 감점 처리한다. 기능적으로 독립됐거나 반복적으로 사용되는 기능은 사용자 함수를 정의해서 구현하여야 한다.

(6) 프로그램 구현 시, main() 함수를 호출하여 사용하지 않는다. 즉, 소스 코드 내에 main(); 이라고 호출하지 않는다.

(7) 문제의 출력 형식은 채점을 위해 아래의 실행 예시와 반드시 같게 작성해 주세요. (보드를 그리기 위해서 사용하는 문자는 -, + , ¦ 로 통일, 공백의 개수는 자유, 불일치시 감점 있음)

(8) 프로그램에서 랜덤 시드는 프로그램 시작 시 main() 에서 srand(time(NULL)); 함수를 한번만 호출하도록 하여 한번만 초기화 한다.

(9) 채점을 위해 랜덤을 활용하는 '보드판 초기화', '플레이어 순서 정하기' 는 반드시 보드 초기화 후 플레이어 순서를 정하는 방식으로 코드를 진행한다.

[게임 규칙]

(1) 보드의 크기는 4 X 4인 정사각형으로, 1~8까지의 숫자 두개 씩(총 16개)를 이용하여 각 칸에 랜덤하게 설정하여 초기화한다.

(2) 초기화 한 후 플레이어에게 보드판에 적혀있는 숫자를 3초간 보여주고 모두 가린다. (3초간 지연하는 기능은 마지막 페이지의 힌트를 참고할 것)

(3) 게임에 참여하는 두 플레이어는 번갈아 가며 자기 차례에 가려져 있는 두 칸을 연다.

(4) 열어본 두 칸의 숫자가 같은 경우 해당 플레이어는 1점을 획득하고, 순서를 한번 더 진행하게 된다. 다를 경우 3초간 보여주고 다시 가리고, 상대방의 순서가 된다.

(5) 오직 가려진 칸만 열 수 있다.

(6) 모든 칸이 열릴 때까지 게임을 진행한다.

(7) 모든 칸이 열렸을 때 점수가 높은 사람이 이기게 된다.

[설명 및 요구사항]

(1) 카드 뒤집기 게임을 2인용 게임으로 구현하도록 한다.

(2) 게임을 시작할 때 플레이어1과 플레이어2 중 누가 먼저 시작 할지는 매 경기마다 임의로 결

정되게 한다.

(3) 출력되는 보드의 좌표는 아래와 같다.

(0, 0)	(0, 1)	(0, 2)	(0, 3)
(1, 0)	(1, 1)	(1, 2)	(1, 3)
(2, 0)	(2, 1)	(2, 2)	(2, 3)
(3, 0)	(3, 1)	(3, 2)	(3, 3)

(4) 플레이어는 자기 순서에, 보드의 좌표에 해당되는 4개의 정수 (1번 칸(x1, y1), 2번 칸(x2, y2))를 입력한다. 틀린 범위의 값과 같은 잘못된 입력은 없다고 가정한다. 단, 플레이어가 실수로 이미 열린 칸을 선택한 경우에는, 다시 좌표를 입력 받도록 한다.

(5) 입력한 두 칸의 숫자를 확인하여 같은 숫자이면, 해당 플레이어의 점수를 1점 올리고 두 칸을 영구적으로 공개한다.

(6) 입력한 두 칸의 숫자를 확인하여 다른 숫자이면, 두 칸을 3초간 보여주고 다시 가린다.

(7) 모든 칸의 숫자가 공개될 경우 경기가 끝난다. 경기가 끝날 때 어떤 플레이어가 이겼는지 혹은 비겼는지에 대해 메시지를 출력한다. (‘Player 1 wins, ‘Player 2 wins’, ‘Draw’ 중 출력)

(8) 출력되는 보드판은 가려진 칸을 ‘*’, 열린 칸을 설정된 숫자로 표시한다.

(9) 한 경기가 끝나면 경기를 계속할지 묻는다. 사용자가 ‘y’를 입력하면 새 경기를 시작하고 ‘n’를 입력하면 ‘Good bye’를 출력하고 프로그램을 종료한다. 잘못된 입력은 고려하지 않는다.

(10) 이 문제를 해결하기 위해 반드시 아래 기능에 해당하는 **사용자 정의 함수**를 정의하고 사용해야 한다. 아래 기능 외의 필요한 함수를 정의해서 사용할 수 있다.

- **print_board: 보드판을 출력하는 기능**
- **check_input: 보드판의 두 칸을 열 수 있는지 확인하는 기능**
- **init_board: 보드판을 초기화하는 기능**

[실행 예시 및 설명]

(1) 초기 실행 화면

보드의 칸에 랜덤하게 숫자를 설정하고(1~8숫자 2개씩) 3초간 사용자에게 보여준다.

```
Memorize the board number!!

    0   1   2   3
  +---+---+---+---+
0 ¦ 1 ¦ 4 ¦ 6 ¦ 3 ¦
  +---+---+---+---+
1 ¦ 7 ¦ 3 ¦ 2 ¦ 6 ¦
  +---+---+---+---+
2 ¦ 5 ¦ 1 ¦ 2 ¦ 5 ¦
  +---+---+---+---+
3 ¦ 4 ¦ 8 ¦ 7 ¦ 8 ¦
  +---+---+---+---+
```

(2) 3초뒤 화면을 모두 지우고 가린 보드를 보여준 뒤 플레이어 입력을 받는다. (지우기 기능은 assn1에서 사용한 ‘화면 지우기’를 이용할 것)
두 플레이어 중 먼저 게임을 시작할 플레이어가 매 경기마다 임의로 결정되게 한다.
아래는 player 1이 먼저 게임을 시작하는 경우이며, 가린 보드가 출력되고 player 1의 입력을 기다린다.

```
     0   1   2   3
   +---+---+---+---+
 0 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 1 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 2 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 3 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+

[player 1] :
```

(3) player 1이 (0, 0), (0, 1) 의 칸을 오픈했다. 오픈 한 칸의 숫자를 보여주고 불일치 했기때문에 3초 뒤에 화면을 모두 지우고 가린 보드를 출력한다. 그 후 플레이어 2의 차례가 된다.

```
 3 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+

[player 1] : 0 0 0 1

     0   1   2   3
   +---+---+---+---+
 0 ¦ 1 ¦ 4 ¦ * ¦ * ¦
   +---+---+---+---+
 1 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 2 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 3 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+

Mismatch!
```

(3초뒤 화면 지우기 후 아래 출력)

```
     0   1   2   3
   +---+---+---+---+
 0 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 1 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 2 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 3 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+

[player 2] :
```

(4) player 2가 (0, 0), (2, 1) 의 칸을 오픈했다. 두 칸은 모두 1로 일치하므로 해당 칸을 영구히 오픈하고, player 2가 1점을 획득한다. 그 후 player 2가 한번 더 플레이한다.

```
 3 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+

[player 2] : 0 0 2 1

     0   1   2   3
   +---+---+---+---+
 0 ¦ 1 ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 1 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 2 ¦ * ¦ 1 ¦ * ¦ * ¦
   +---+---+---+---+
 3 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+

player 2 get score!!
[player 1] 0 : 1 [player 2]

player 2 get one more chance!!
[player 2] :
```

(5) player 2 가 (0, 0), (3, 1) 의 칸을 오픈했다. 이미 (0,0)은 오픈 되어있기 때문에 다시 입력 받는다.

```
     0   1   2   3
   +---+---+---+---+
 0 ¦ 1 ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 1 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+
 2 ¦ * ¦ 1 ¦ * ¦ * ¦
   +---+---+---+---+
 3 ¦ * ¦ * ¦ * ¦ * ¦
   +---+---+---+---+

player 2 get score!!
-------------------------
[player 1] 0 : 1 [player 2]
-------------------------

player 2 get one more chance!!
[player 2] : 0 0 3 1

already open... try again..
[player 2] :
```

(6) 위와 같은 방식으로 틀리면 플레이어 1, 2 순서를 번갈아가며 모든 칸이 오픈될 때까지 게임을 진행한다.

(7) 아래는 모든 칸이 오픈되고 게임이 끝난 상황이다. player 1이 5:3으로 승리하였다.

```
     0   1   2   3
   +---+---+---+---+
 0 ¦ 1 ¦ 4 ¦ 6 ¦ 3 ¦
   +---+---+---+---+
 1 ¦ 7 ¦ 3 ¦ 2 ¦ 6 ¦
   +---+---+---+---+
 2 ¦ 5 ¦ 1 ¦ 2 ¦ 5 ¦
   +---+---+---+---+
 3 ¦ 4 ¦ 8 ¦ 7 ¦ 8 ¦
   +---+---+---+---+

player 1 get score!!
--------------------------
[player 1] 5 : 3 [player 2]
--------------------------

Game End! Player 1 wins!
Continue? (y/n)
```

(8) 'y'를 입력 받아 새 게임 보드를 다시 만들어 새로운 게임을 시작한다.
(게임진행은 1~7과 같음)

```
Game End! Player 1 wins!
Continue? (y/n) y

New Game!
Memorize the board number!!

     0   1   2   3
   +---+---+---+---+
 0 ¦ 1 ¦ 4 ¦ 6 ¦ 3 ¦
   +---+---+---+---+
 1 ¦ 7 ¦ 3 ¦ 2 ¦ 6 ¦
   +---+---+---+---+
 2 ¦ 5 ¦ 1 ¦ 2 ¦ 5 ¦
   +---+---+---+---+
 3 ¦ 4 ¦ 8 ¦ 7 ¦ 8 ¦
   +---+---+---+---+
```

(9) 경기를 비기고 'n'를 입력 받아 프로그램을 종료했다.

```
player 2 get score!!
--------------------------
[player 1] 4 : 4 [player 2]

Game End! Draw!
Continue? (y/n) n
Good bye

계속하려면 아무 키나 누르십시오 . . .
```

(힌트)

- ***배열 선언 및 사용***

1. 배열의 크기를 선언할 때 변수를 사용하면 안됩니다(int array[i][i];). 충분한 크기로 선언한 뒤(int array[4][4];), 필요한 부분만 사용하면 됩니다.
2. 생성한 카드의 정보를 담고 있는 배열 말고도 다른 배열 변수를 만들어서 선택된 칸의 정보를 기억하는 것이 좋습니다.
3. 배열을 parameter 로 다른 함수에 넘겨주었을 경우, 그 함수에서 배열의 값을 바꾸면 main 함수에서도 바뀐 상태가 유지됩니다. 이는 배열이 포인터 형태를 띄고 있기 때문입니다.

```
#include <stdio.h>
void foo(int x[]);
int main()
{
    int a[10];
    a[0] = 0;
    foo(a);
    printf("%d\n", a[0]); //prints 1
    return 0;
}
void foo(int x[])
{
    x[0] = 1;
    return;
}
```

- ***지연 효과 주기***

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

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

참고) 리눅스 또는 맥에서 프로그래밍을 하는 경우

```
#include <stdio.h>
#include <unistd.h>  // sleep 함수 사용을 위해
int main()
{
    printf("첫 번째 메시지 입니다\n");
    sleep(3);   // 매개변수는 초 단위. 3초 지연
    printf("이 메시지는 3초뒤에 출력됩니다\n");
    return 0;
}
```

- ***어싸인 팁***

1. 프로그램 작성 중 무한 루프에 의해 프로그램이 끝나지 않을 경우, Ctrl+C를 누르세요.