CSED101. Programming & Problem solving

Spring, 2018

Programming Assignment #2

(60 points)

최연규(ygchoi1024@postech.ac.kr)

- ***Due***: **2018.04.06 23:59**
- ***Development Environment***: **Windows Visual Studio 2017**

- ***제출물***
  - **C Code file (.c)**
    - 확장자를 포함한 소스파일의 이름은 assn2.c로 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 (보고서 명 미 준수 시 감점)
    - 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: 가위바위보 달리기

(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, 총 계단 수, 가위바위보를 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)

이 게임은 2인으로 진행이 되며, 본 과제에서는 컴퓨터와 사용자가 플레이를 하게 된다.
컴퓨터와 사용자가 서로 맞은편 언덕 위 계단에서 가위바위보를 하면서 규칙에 따라 계단을 오르거나 내려가게 된다. 이때, 먼저 상대편의 시작점에 도착하는 플레이어가 이기는 게임이다.

승패를 위한 가위바위보는 "가위바위보! 하나 빼기 일!" 규칙을 사용하도록 한다.

(1) 가위바위보!

게임을 위해 각 플레이어(컴퓨터, 사용자)들은 가위바위보 중에 2개의 후보를 선택해서 먼저 낸다. (이 때, 선택한 2개의 후보는 같을 수 없다고 가정한다.)

(2) 하나 빼기 일!

플레이어들은 서로 선택한 2개의 후보를 확인 한 후, 각각 자신이 선택한 후보 2개 중 최종 하나를 선택해서 낸다.

(3) 승패 결정 및 계단 이동 규칙

"하나 빼기 일"을 통해 최종 선택한 후보로 가위바위보의 승부를 결정한다.

1) 가위로 이긴 경우: 1계단 이동
2) 바위로 이긴 경우: 2계단 이동
3) 보로 이긴 경우: 3계단 이동
4) 비긴 경우: 플레이어들의 위치 변동 없음

게임을 위해 아래의 기준으로 가위바위보를 숫자와 매칭해서 사용하도록 한다.

- **가위, 바위, 보 숫자 매칭**
  : 가위(1), 바위(2), 보(3)

I. 초기 선택 메뉴

프로그램 시작 시, 그림 1처럼 3가지 선택사항이 있는 메뉴를 출력하고, 사용자로부터 3가지 선택사항 중 한가지를 입력 받도록 한다.

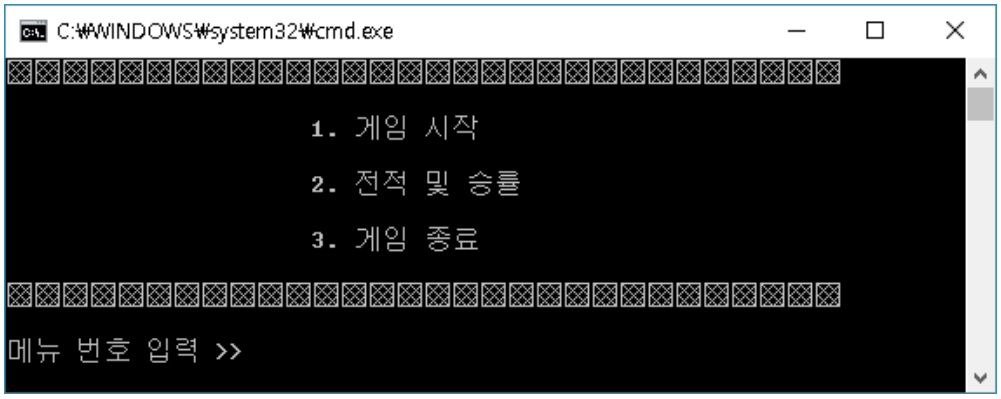


그림 1. 초기 메뉴 화면 예시

- 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 아래 위의 구분선을 출력하기 위해 반드시 아래의 함수를 정의하고 사용해야 한다.
  - ***draw_line()***: 40개의 "▨"을 출력한다. (for 문을 사용하여 출력할 것)

  ※ "▨" 문자 입력은 'ㅁ'키 입력 후, 한자키를 눌러 해당 문자 선택

숫자 1, 2, 3 외의 값이 입력되면 그림 1이 그대로 다시 출력되며 재입력을 받도록 한다. 메뉴가 다시 출력될 때, 화면이 지워진 후 메뉴가 출력되도록 한다. 이 부분 구현에 대해서는 Tips(마지막 쪽)의 '화면 지우기' 설명을 참조한다. (이 과제의 모든 출력화면은 화면을 지운다고 명시한 경우, 그에 맞게 지운 후 출력한다)

그림 1에서 3을 입력하면 그림 2와 같이 "게임을 종료하시겠습니까? (y/n)"라는 메시지를 출력하며 사용자의 입력을 기다린다. 이 때, 사용자가 y를 입력하면 "게임을 종료합니다..."라는 메시지 출력 후 프로그램을 종료한다. 사용자가 n을 입력하면 그림 1의 초기 메뉴 선택 화면으로 돌아간다. (Y/N은 대소문자 상관없이 입력 받을 수 있어야 하며, Y/y, N/n 외의 입력은 고려하지 않는다.)

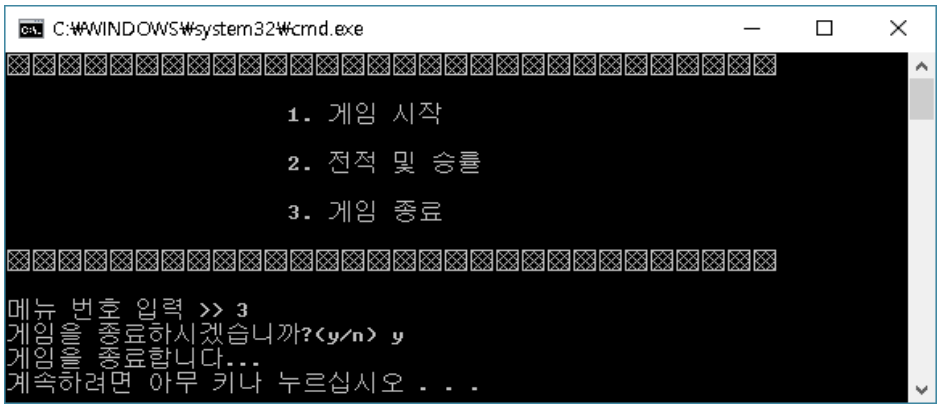


그림 2. 게임 종료 화면 예시

II. 게임 시작

그림 1의 메뉴에서 1을 선택하면, ***game_start()*** 함수가 호출되며 게임이 시작된다. 모든 게임은 game_start() 함수 내에서 진행되어야 한다. game_start()함수 내에서 game_start()함수를 다시 호출하는 일이 없도록 한다. game_start()함수를 통하여 플레이어와 컴퓨터 중 누가 이겼는지 정수로 반환하도록 한다. (예를 들면, 플레이어가 컴퓨터의 시작점에 먼저 도착한 경우 1을 반환, 컴퓨터가 플레이어의 시작점에 먼저 도착한 경우 0을 반환)

이 게임은 먼저 사용자로부터 이동할 계단의 개수를 입력 받는 것에서부터 시작한다. 화면 지우기 후, 그림 3과 같이 계단의 개수를 입력 받기 위한 텍스트가 출력되고 사용자로부터 입력을 받을 준비를 한다. 단 계단의 개수는 **10개~30개로 한정**한다. 범위를 벗어난 입력에 대해서는 아래의 예시처럼 다시 입력 받도록 한다.

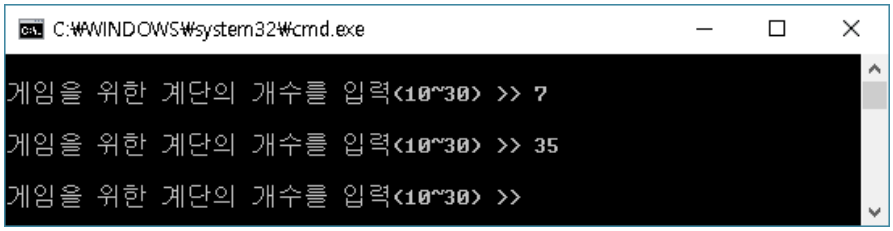


그림 3. 계단 개수 입력 예시

범위 내의 계단의 개수를 입력하면 화면 지우기 후 컴퓨터와 플레이를 할 화면이 그림 4와 같이 출력된다

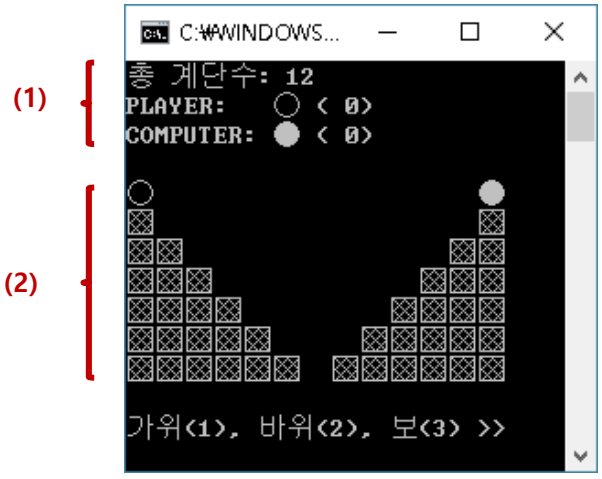


그림 4. 계단이 생성된 화면 예시(12개의 계단을 입력한 경우)

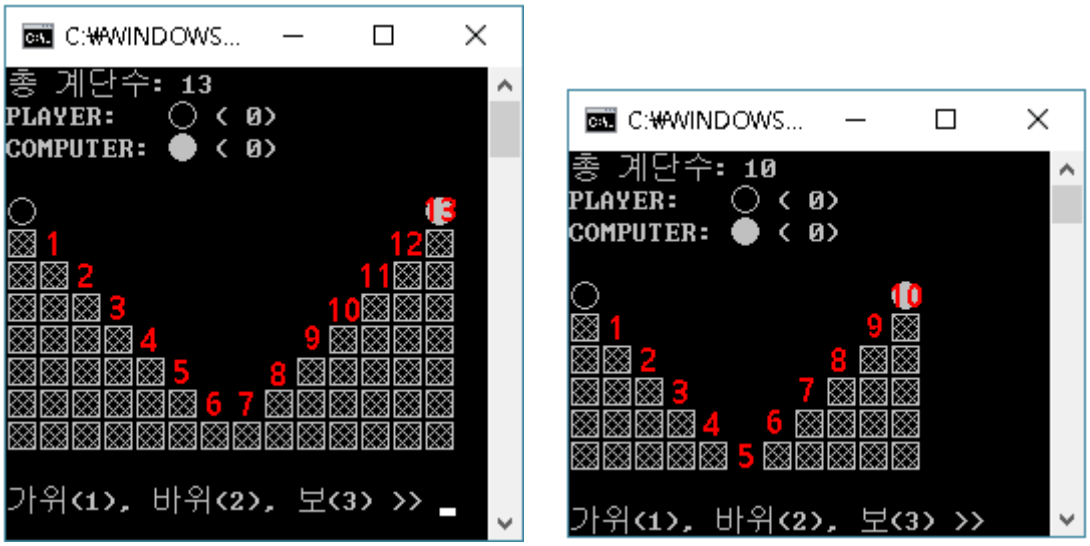
◆ **그림 4의 (1)**

- 사용자가 입력한 총 계단 수
- PLAYER의 위치를 표시하는 기호 "○"와 현재까지 이동한 계단 수
- COMPUTER의 위치를 표시하는 기호"●"와 현재까지 이동한 계단 수

◆ **그림 4의 (2)**

PLAYER는 왼쪽 계단 위에서 COMPUTER는 오른쪽 계단 위에서 게임을 시작한다. PLAYER는 "○", COMPUTER는 "●" 기호로 현재 위치를 나타내고 있다. 현재는 둘 다 이동한 계단수가 0인 상태이다. 게임 도중에 같은 계단에 서게 되면 기호 "◑"로 표시하도록 한다.

게임을 위해 이동해야 할 계단의 수는 홀수일 때와 짝수일 때의 구성이 다르니 그림 5를 참고하여 작성하도록 한다. 그림 5의 빨간색으로 표시된 숫자는 플레이어가 이동하게 될 순서이며 플레이어가 현재 위치에서 한 계단을 이동하게 되면, 1이라고 적힌 위치로 이동하게 된다.



(a) 총 계단수가 홀수인 경우 (b) 총 계단수가 짝수인 경우

그림 5. 총 계단 수 예시 화면

➢ 계단을 출력하기 위해서는 다음과 같은 함수를 정의한 후 사용한다.

- ***print_stairs()***: 컴퓨터와 플레이어의 현재 이동한 계단 수와 이동해야 할 총 계단 수를 매개변수를 전달받아 화면에 현재 진행상황을 출력한다. "가위바위보! 하나 빼기 일!"을 통해 승부가 결정될 때마다 해당함수를 호출하여 현재 진행상황을 지속적으로 출력한다. (중첩 for문을 사용하여 구현할 것!)

그림 4의 상태에서 플레이어는 숫자 1(가위), 2(바위), 3(보) 중 2개의 후보를 선택하여 입력한다. 입력 시, 2개의 숫자를 공백으로 구분하여 입력한다. 이 때, 플레이어가 선택한 2개의 후보는 같지 않아야 한다. (컴퓨터 또한 마찬가지이다.) 플레이어의 입력 중, 선택 범위를 벗어나거나 동일한 후보를 선택한 경우의 실행 예시는 그림 13을 참조하도록 한다.

컴퓨터는 Random 함수를 기반으로 1, 2, 3 중 2개의 후보를 선택하게 된다. (단, 컴퓨터는 프로그램을 실행할 때마다 랜덤하게 가위바위보를 내도록 구현해야 한다.)

➢ 컴퓨터의 2개의 후보를 생성하기 위해서 다음과 같은 함수를 정의한 후 사용한다.

- ***void generate_two_numbers(int *n1, int *n2):*** 1, 2, 3 중 겹치지 않는 2개의 후보 선택

플레이어의 후보 2개를 입력 받으면 그림 6과 같이 플레이어와 컴퓨터가 각각 무엇을 냈는지 출력해 준다. 컴퓨터의 후보가 보(3), 바위(2), 플레이어의 후보가 바위(2), 가위(1)임을 출력한 상태이다.

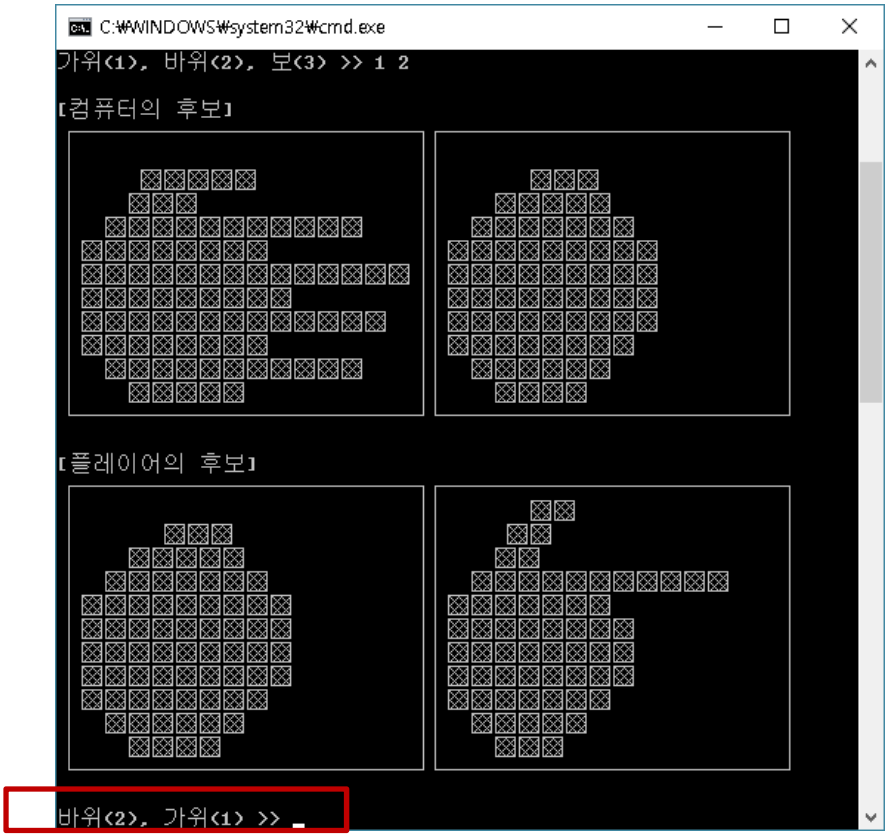


그림 6. 두 플레이어의 후보 출력 예시

➢ 가위바위보 출력을 위해 아래의 함수를 정의하고 사용한다.

- ***print_rock_scissors()***
  그림 6의 [플레이어의 후보] 출력과 같이 바위, 보의 이미지를 출력
- ***print_paper_rock()***
  그림 6의 [컴퓨터의 후보]의 출력과 같이 보, 바위의 이미지를 출력
- ***print_scissors_paper()***
  그림 14의 [플레이어의 후보]의 출력과 같이 가위, 보 이미지를 출력
- ***print_RPS()***
  플레이어(또는 컴퓨터)의 두 후보를 매개변수로 전달받아 위에서 정의한 3개의 함수를 호출하여 출력

다음으로 "하나 빼기 일"을 위해서 플레이어 후보 중 2개를 선택하도록 텍스트가 출력되고 플레이어 입력을 기다린다. 그림 6은 플레이어의 후보가 "바위(2), 가위(1)"인 경우로 후보에 해당하는 텍스트만 출력됨을 알 수 있다. (출력 순서는 상관없다.)

플레이어가 2개의 후보를 제외한 다른 값을 입력한 경우, 아래와 같은 메시지 출력 후, 다시 입력 받도록 한다.

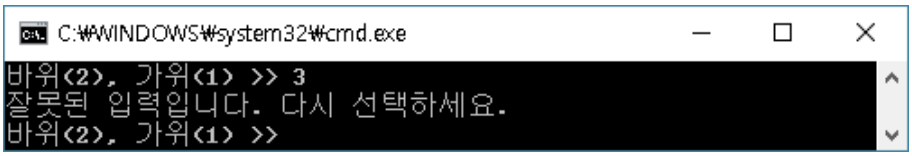


그림 7. 플레이어 후보 입력 오류 예시

컴퓨터는 2개의 후보 중 다음의 규칙을 따라 최종 후보를 선택하도록 한다.

- 플레이어와 2개의 후보 모두 일치하는 경우
  2개의 후보 중 이기는 후보를 선택하도록 한다.
  예) 2개의 후보가 가위, 바위인 경우 바위를 선택
- 나머지 경우
  2개의 후보 중 랜덤하게 하나를 선택하도록 한다.

그림 8은 플레이어는 2개의 후보 중 가위(1)를 선택, 컴퓨터는 자신의 2개의 후보 중 랜덤하게 보(3)를 선택한 상황으로 그 결과를 판단하여 아래에 "플레이어가 이겼습니다!!"라는 메시지가 출력되었다.

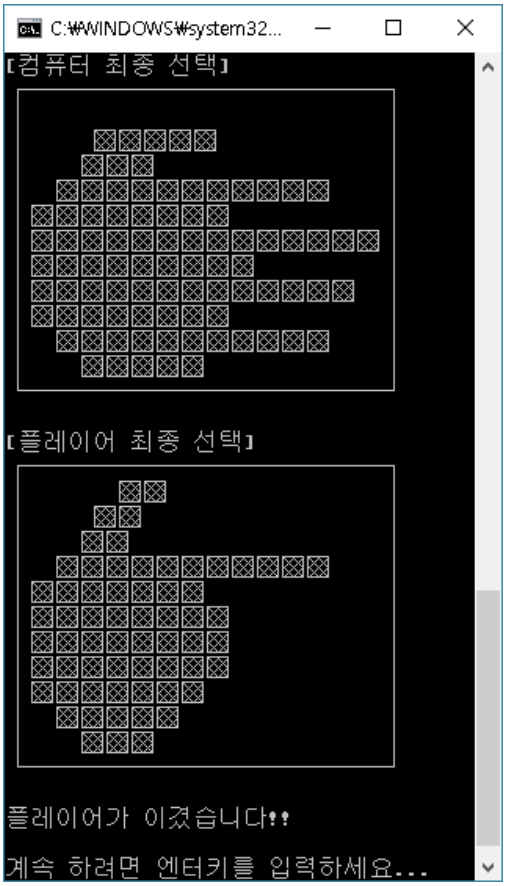


그림 8. "하나 빼기 일"의 결과

- 게임 승패 판단을 위하여 아래의 함수를 정의하고 사용한다
  - ***win_lose_draw():*** 컴퓨터와 플레이어가 최종으로 선택한 후보를 매개변수로 전달받아서 플레이어의 이김(WIN), 짐(LOSE), 비김(DRAW)를 판단하여 이긴 경우는

1을, 진 경우는 -1을, 비긴 경우는 0을 반환한다. (3가지 경우를 구분할 수 있는 다른 정수 값으로 반환해도 무방하다.)

플레이어가 엔터를 입력하면 화면 지우기 후 바로 다음 판을 진행한다. 이때, 플레이어와 컴퓨터의 현재 계단에서의 위치를 숫자와 그림으로 그림 9와 같이 표시한다. 앞에서 명시한 계단 이동 규칙 (가위(1)로 이긴 경우 1칸 이동, 바위(2)로 이긴 경우 2칸 이동, 보(3)로 이긴 경우 3칸 이동)을 적용하면, 플레이어가 가위(1)로 이겼으므로 한 칸 아래로 이동했음을 볼 수 있다.

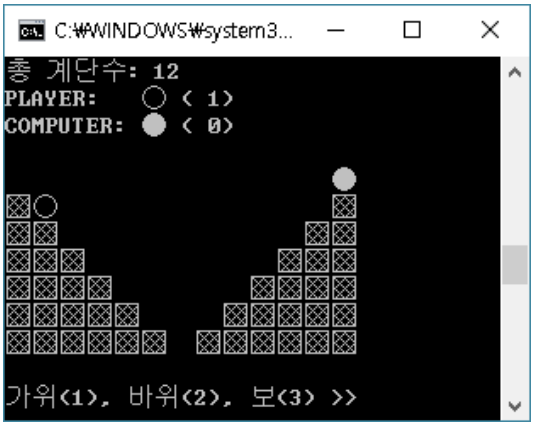


그림 9. 플레이어가 가위(1)로 이긴 경우

그림 10은 플레이어의 위치 이동 1, 컴퓨터의 위치 이동 0인 상태에서 컴퓨터가 바위(2)로 이겨 두 계단 이동한 경우의 예시이다. 승부의 결과는 "컴퓨터가 이겼습니다!!"가 출력된 후, 플레이어가 엔터를 입력하면 그림 10의 오른쪽 예시처럼 승부의 결과가 반영되어 출력된다.

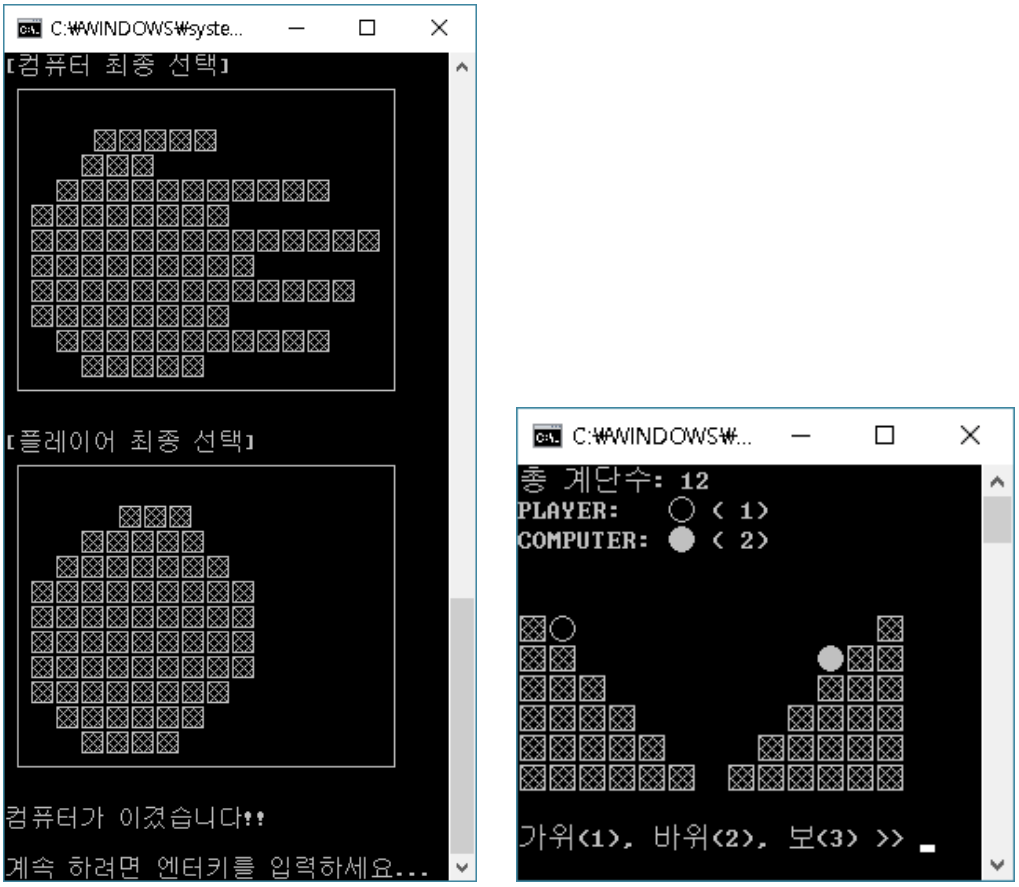


그림 10. 컴퓨터가 이긴 경우

그림 11은 플레이어 위치 이동 1, 컴퓨터의 위치 이동 2인 상태에서 둘 다 보(3)를 내어 비긴 경우로 계단에서의 위치 이동은 없다.

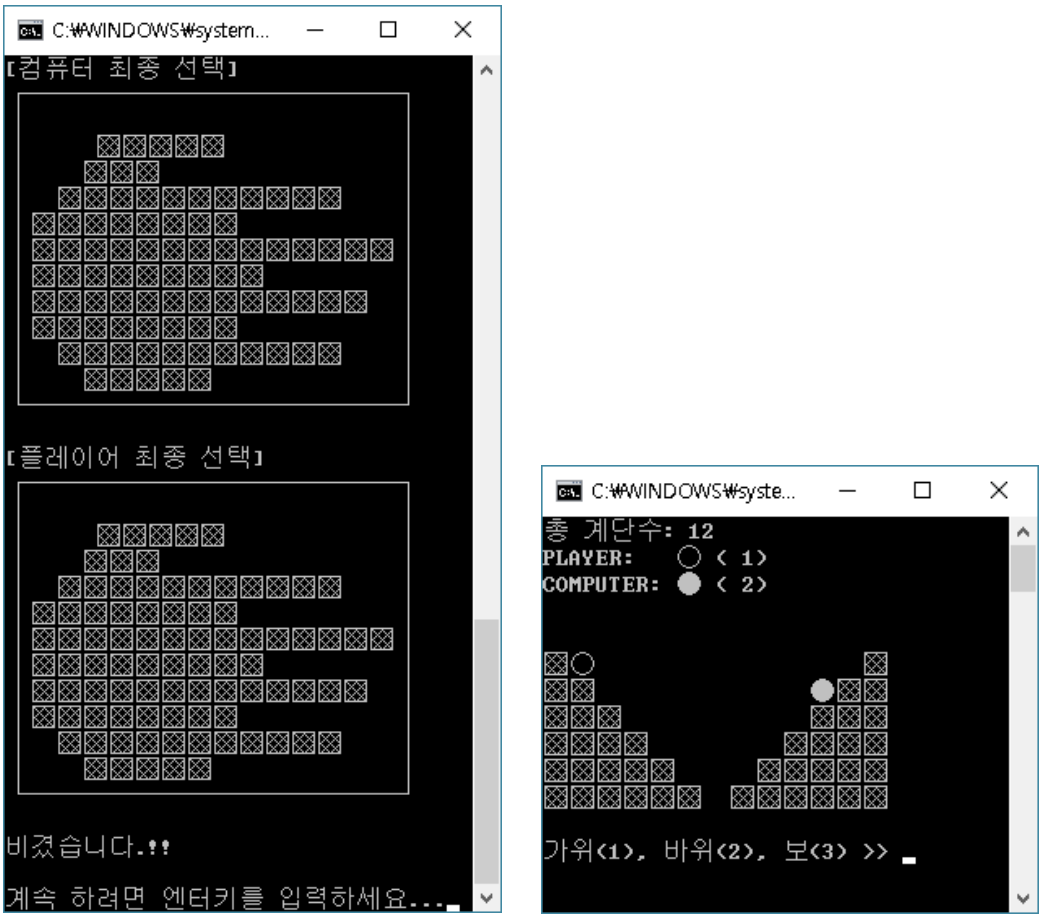


그림 11. 비긴 경우

그림 12는 게임 중간에 플레이어와 컴퓨터가 같은 계단 위치에 서게 된 경우로 기호 "◑"로 위치가 표시됨을 보여 준다.

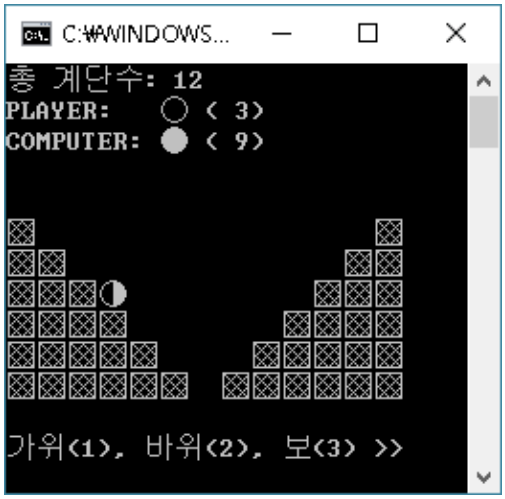


그림 12. 게임 중간에 계단의 위치가 같은 경우

플레이어가 1(가위), 2(바위), 3(보)에서 벗어난 범위의 숫자 또는 동일한 후보를 선택하여 입력하면 특별한 에러 메시지 없이 그림 13처럼 다시 입력을 받도록 한다.

그림 13. 플레이어의 2개의 후보 선택 입력 에러 예시

III. 게임 종료

게임이 끝나는 경우는 아래의 2가지 경우이다.

(1) 게임 포기

플레이어는 게임 중간에 포기할 수 있는 기회가 주어지는 데, 그림 14처럼 "하나 빼기 일"단계에서 플레이어 입력으로 0을 입력한 경우이다. "이번 게임을 포기하시겠습니까?(y/n)" 라는 메시지가 출력되고 플레이어의 입력을 기다린다.

이 때, 플레이어가 y를 선택하면 플레이어 패배가 기록되며, 화면 지우기 후 초기 메뉴 선택(그림 1) 화면으로 돌아간다. 플레이어가 n을 선택하면 그림 14처럼 다시 "하나 빼기 일"단계의 플레이어 입력을 기다린다. (단, y, Y, n, N 외의 입력은 고려하지 않는다.)

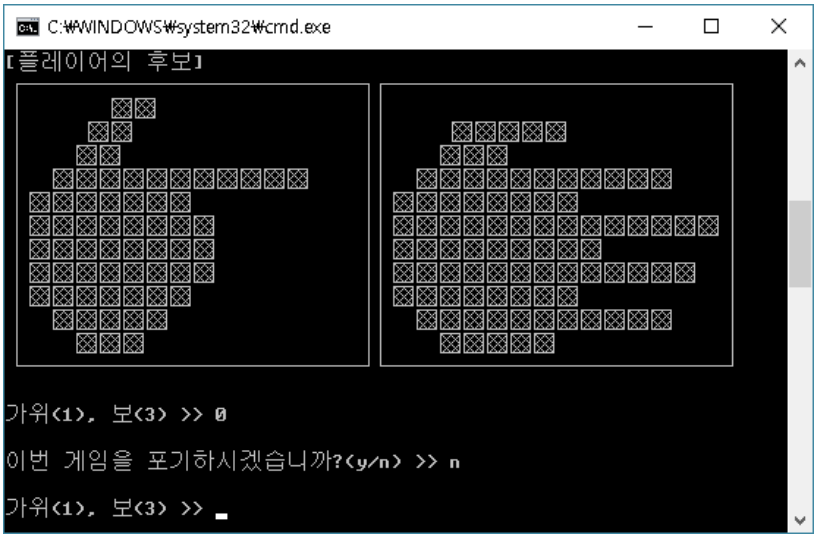


그림 14. 플레이어가 0을 선택한 경우(게임 포기)

(2) 게임 참가자(플레이어, 컴퓨터)중의 하나가 먼저 상대방의 시작점에 도착한 경우

그림 15는 12개의 계단 중 플레이어는 10번째 계단에 위치, 컴퓨터는 11번째 계단에 위치한 상태(그림 15의 왼쪽 예시)에서 "가위바위보 하나 빼기 일"이 진행되어, 플레이어가 마지막 보(3)로 이긴 결과를 화면 지우기 후 그림 15의 오른쪽 예시처럼 보여준다.

플레이어는 10번째 계단에서 3계단을 이동해야 하지만 총 12개의 계단이 존재하므로 플레이어의 최종 위치는 12번째 계단으로 컴퓨터의 게임 시작위치이다. 그리고, 그 아래 최종 결과 "플레이어 최종 승리"를 출력한다.

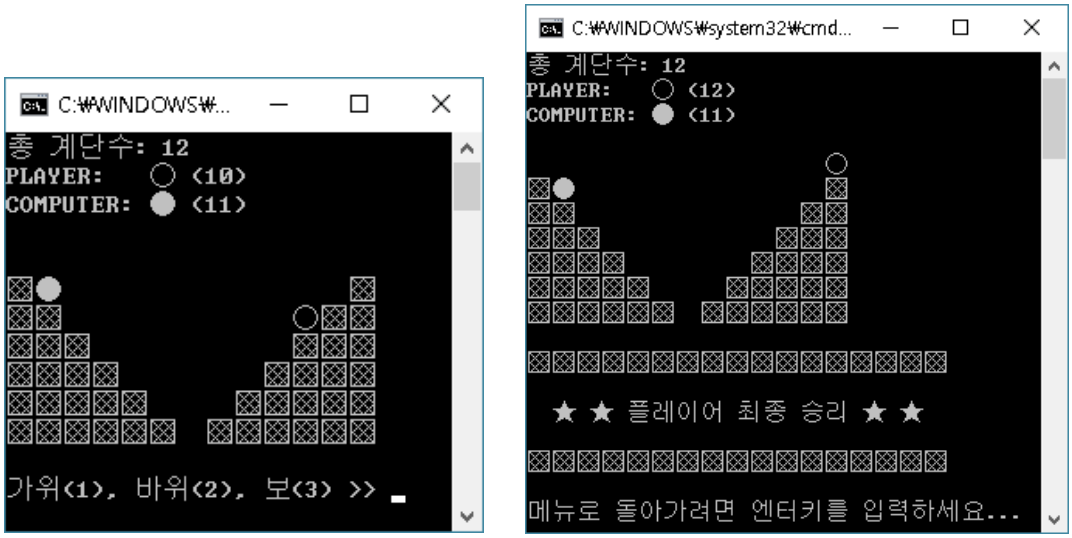


그림 15. 게임 종료 예시 (플레이어 승)

아래 그림 16은 컴퓨터가 먼저 플레이어의 시작 위치에 도착한 경우로 컴퓨터가 승리한 경우에 출력되는 예시이다. 엔터를 입력하면 화면 지우기 후 그림 1의 초기화면 메뉴로 돌아 갈 수 있도록 구현한다.

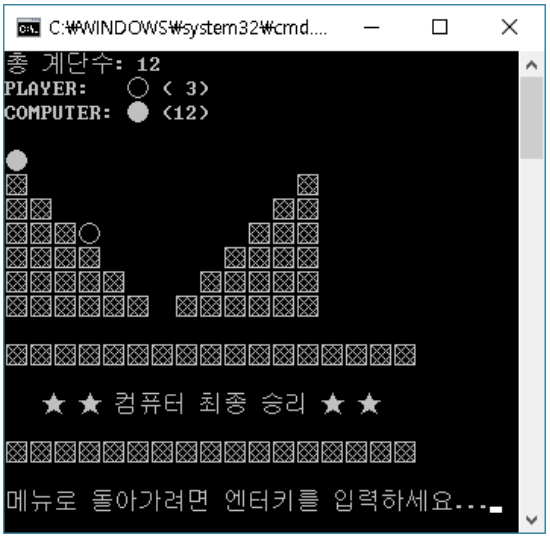


그림 16. 게임 종료 예시 (컴퓨터 승)

IV. 전적 및 승률 확인

그림 1의 메뉴에서 2를 선택하면 화면 지우기 후 그림 17과 같이 전적 및 승률 화면을 출력한다. 그림 17은 게임이 한번도 진행되지 않은 경우의 승률 확인 화면이다. 전적 및 승률 확인 화면에서 엔터키를 누르면 화면 지우기 후 메인 메뉴로 돌아가도록 한다.

- 전적 및 승률 확인을 위해서는 다음과 같은 함수를 정의하고 사용한다.
  - ***show_record()***: 플레이어의 승/패를 매개변수로 전달받아 승/패와 그에 따른 승률을 계산하여 출력

그림 17은 게임이 한 판도 진행되지 않은 경우로 승리/패배/승률 모두 0으로 출력된다.

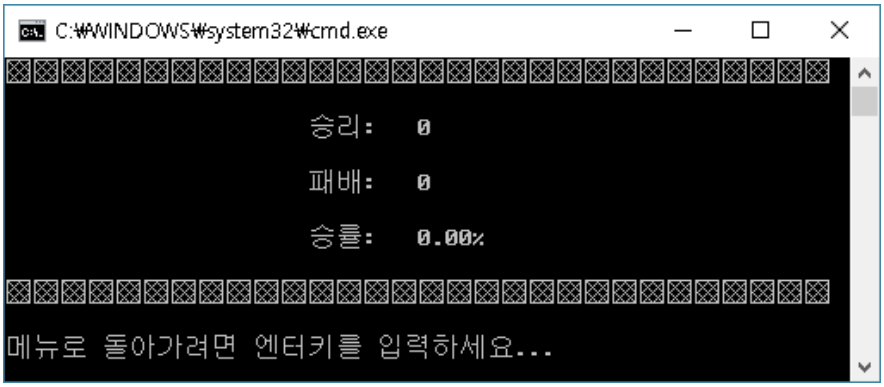


Figure 17. 전적 및 승률(한 번도 진행하지 않았을 때)

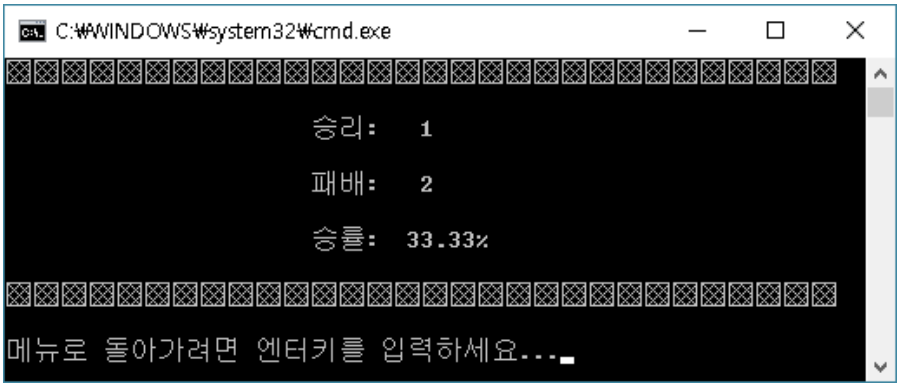


그림 18. 전적 및 승률(게임을 세 번 진행한 상황)

그림 18은 현재까지 세 판의 게임이 진행된 상황을 나타내고 있으며, 플레이어가 1번 승리, 2번 패배, 따라서 승률은 33.33%로 산출된 결과값을 출력하고 있다.

Tips

■ 화면 지우기

<stdlib.h>를 포함한 뒤, system 함수를 사용하여 현재 콘솔 창에 출력된 내용을 지운다. 아래 코드를 그대로 실행해본 뒤, ''system("cls");"를 주석 처리하여 실행해봄으로써 차이를 파악한다.

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

int main()
{
	printf("HELLO WORLD\n");
	system("cls");
	return 0;
}
```