# CSED101. Programming & Problem solving

# Fall, 2016

## Programming Assignment #2

## (60 points)

**이솔애(solae@postech.ac.kr)**

- ***Due***: **2016.10.19 23:59**
- ***Development Environment:*** **Windows Visual Studio 2015**

- ***제출물***
  - **C Code files (*.c)**
    - 확장자를 포함한 소스파일의 이름은 "**assn2.c**"로 할 것.
    - 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.
    - **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.
  - **assn2.c**로 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.

## ■ Problem: 청기 백기 게임

**(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, goto 문, 배열 및 포인터는 사용할 수 없다.
- 사용자로부터 행동 입력 시 ***e, d, i, k*** 이외의 입력에 대해서는 고려하지 않아도 된다. ***e, d, i, k***는 소문자로만 입력한다. 또한 사용자 입력 시간엔 한 개의 키를 한 번만 누른다고 가정한다. 한 입력 시간에 여러 키를 입력하는 상황에 대해서는 고려하지 않아도 된다.
- 이번 과제는 반복문 사용 방법을 익히는 문제이므로 반복문을 사용하라고 되어 있는 부분은 반드시 반복문을 사용하도록 한다.
- **문제의 출력 형식은 채점을 위해 아래의 실행 예시와 최대한 비슷하게 작성해 주세요.**

**(설명)**
청기 백기 게임은 컴퓨터가 지시하면, 사용자가 제한 시간 내에 행동을 취하는 식으로 진행된다. 사용자는 제한시간 내에 알맞은 행위를 취해야 하며, 맞을 경우 Score가 1점 올라가고, 틀릴 경우 Life가 한 개 줄어든다. Life는 처음에 5개가 주어지며, Life가 모두 줄어들면 게임이 종료된다. 즉, 다섯 번째로 틀리면 바로 게임이 종료된다.

컴퓨터는 임의로 다음과 같은 지시를 한다.
: **청기 올려, 청기 내려, 백기 올려, 백기 내려, 청기 올리지마**

사용자는 컴퓨터가 지시한 대로 행동해야 한다. 아래와 같이 문자를 매칭해서 사용한다.
: **청기 올려(e), 청기 내려(d), 백기 올려(i), 백기 내려(k), 청기 올리지마(' '(공백))**

사용자는 컴퓨터의 '청기 올려' 지시에 'e'키를, '청기 내려' 지시에 'd'키를, '백기 올려' 지시에 'i'키를, '백기 내려' 지시에 'k'키를 입력하면 올바른 행동을 취한 것으로 간주한다. '청기 올리지마' 지시에는 제한 시간 동안 아무 키도 입력하지 않으면 올바른 행동을 취한 것으로 간주한다.

## I. 메인함수

본 과제는 메인 함수에 switch case문을 반드시 사용하여 메뉴 선택을 하도록 한다.

## II. 초기 선택 메뉴

프로그램 시작 시, Figure 1처럼 3가지 선택 사항이 있는 메뉴를 출력하고, 사용자로부터 3가지 메뉴 중 하나를 입력 받도록 한다.

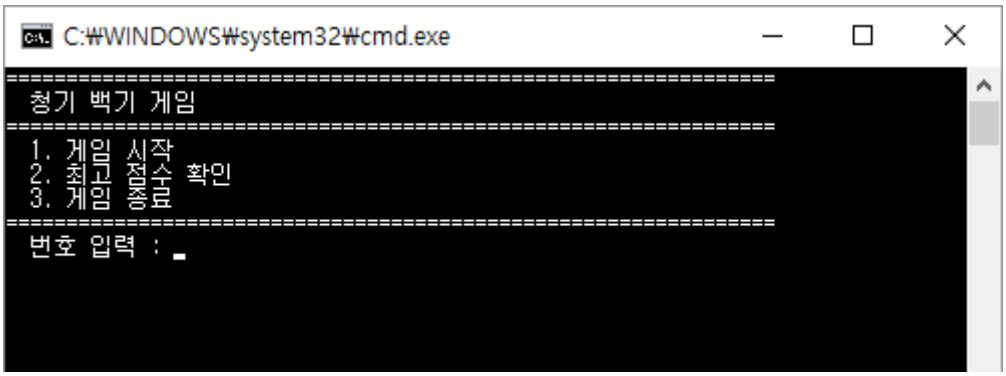



**Figure 1 초기 실행 화면**

- 구분선을 출력하기 위해서는 반드시 아래의 함수를 정의하고 사용해야 한다. 이 함수는 게임 화면 출력에서도 사용된다.
  - ***drawLine*** : 64개의 '='을 출력한다. (for 문을 사용하여 출력할 것)

숫자 1, 2, 3 외의 값이 입력되면 "잘못된 값을 입력하셨습니다. 다시 입력해 주세요."라는 메시지를 출력하고 메뉴가 다시 나타나도록 한다.

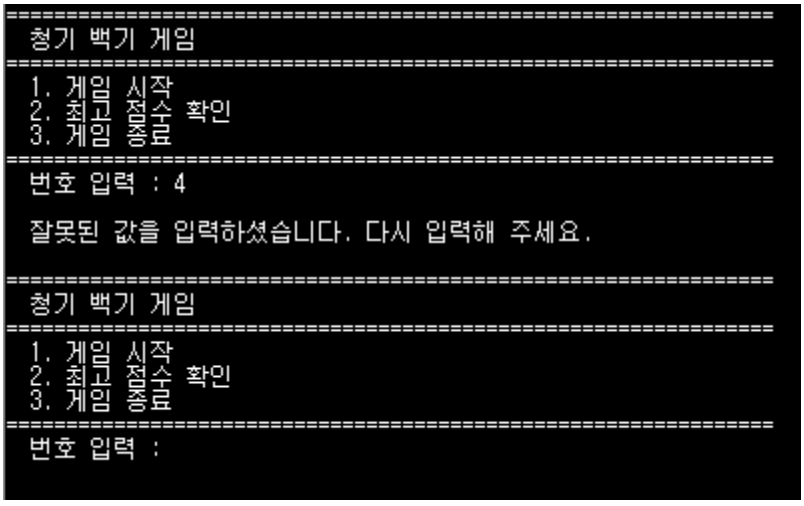



**Figure 2 잘못된 입력 값에 대한 예시**

Figure 2는 해당 메뉴 번호에 없는 숫자 4를 입력했을 때의 화면으로 에러 메시지가 출력되고 초기 메뉴가 다시 출력되었다.

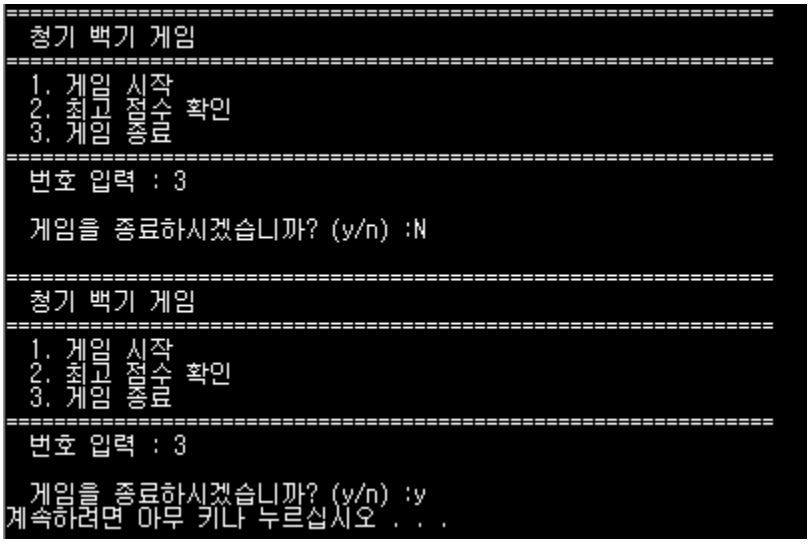



**Figure 3 게임 종료를 위해 3을 입력했을 때의 예시**

Figure 1에서 3을 입력하면 Figure 3과 같이 "게임을 종료하시겠습니까? (y/n)"이라는 메시지를 출력하며 사용자의 입력을 기다린다. 이 때, 사용자가 y를 입력하면 게임이 종료된다. 사용자가 n을 입력하면 Figure 1의 메뉴가 다시 나타난다. (Y/N은 대소문자 상관없이 입력 받을 수 있어야 하며, Y/y, N/n 외의 입력은 고려하지 않는다.)

- 초기 메뉴를 출력하기 위해서는 반드시 아래의 함수를 정의하고 사용해야 한다.
  - ***showMenu*** : 초기 메뉴 화면을 출력하고, 사용자로부터 메뉴 번호를 입력 받아 반환

## III. 게임 시작

청기 백기 게임을 시작하기 위해서 먼저 사용자로부터 게임 난이도(Level)를 입력 받도록 한다. 난이도는 1~3단계가 있으며 Level 1은 3초, Level 2는 2초, Level 3은 1초의 제한 시간이 주어진다. Figure 1에서 숫자 1을 입력했을 때, Figure 4와 같이 게임의 난이도를 입력 받기 위한 메시지가 출력되고, 사용자로부터 입력을 받을 준비를 한다. 난이도의 범위를 벗어난 숫자 입력에 대해서 특별한 에러 메시지 없이 아래와 같이 다시 입력 받도록 한다.

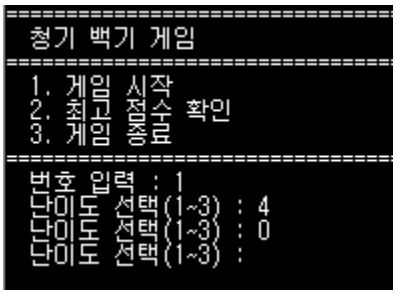


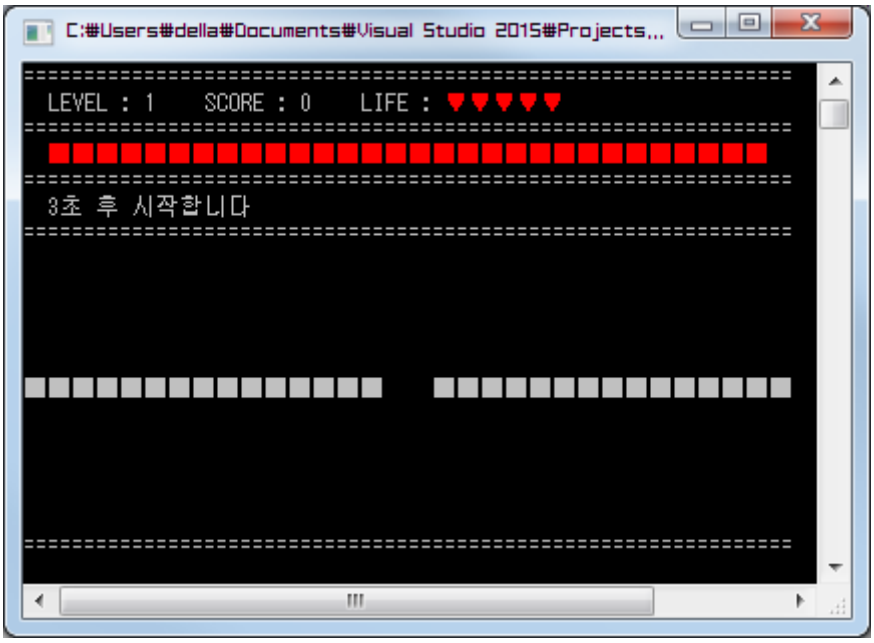

**Figure 4 난이도 입력 예시**

- 난이도 설정을 위해서는 반드시 아래의 함수를 정의하고 사용해야 한다.
  - ***levelSetting*** : 사용자로부터 범위 내의 난이도를 입력 받고, 입력 받은 난이도를 반환

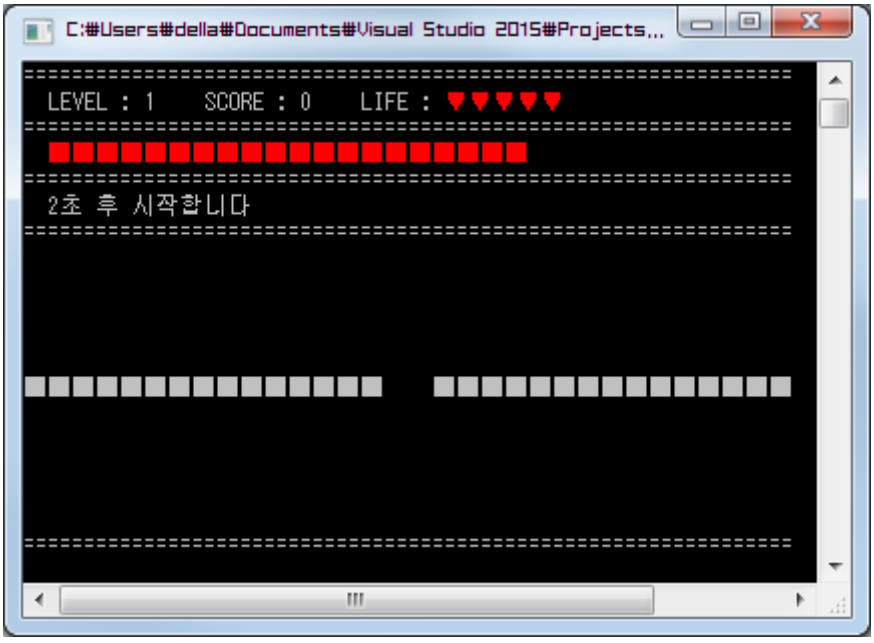
범위 내의 난이도를 입력하면 게임 시작을 위한 3초 카운트다운 화면이 Figure 5와 같이

출력된다. 게임을 위해 카운트다운 화면이 출력 될 때부터는 화면이 지워진 후 다시 출력되도록 한다. 이 부분 구현에 대해서는 **Tips**의 ***'화면 지우기' 설명을 참조***한다. 카운트 다운을 위한 1초간 정지는 Sleep 함수를 사용하거나 (Windows.h에 정의, 사용방법: Sleep(밀리초);), **Tips**의 ***'일시 정지' 설명을 참조***한다. 게임 화면은 지워지고 다시 출력되기를 반복한다.

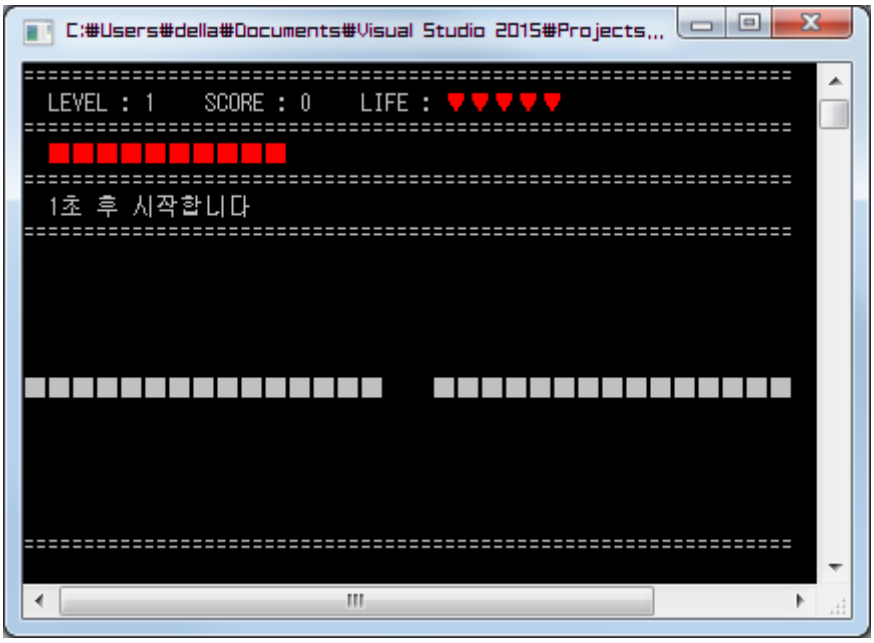
- 게임 시작을 위해서는 반드시 아래의 함수를 정의하고 사용해야 한다.
  - ***startGame*** : 난이도를 매개변수로 전달 받고, 게임 진행 후 점수를 반환.



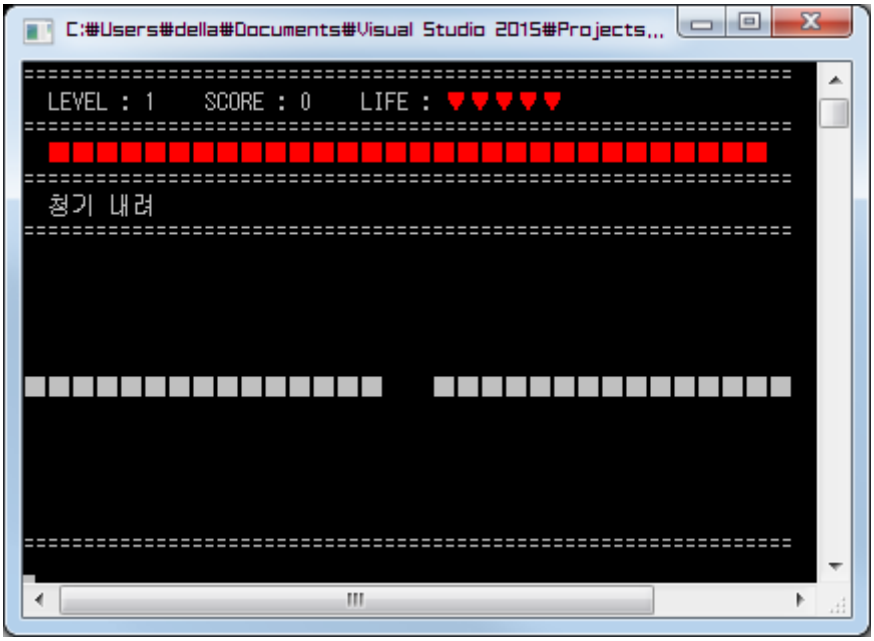

**(a) 카운트다운 시작**




**(b) 카운트다운 시작 후 1초 경과한 화면**




**(c) 카운트다운 시작 후 2초 경과한 화면**




**(d) 카운트다운 시작 후 3초 경과 후, 게임 시작 화면**

**Figure 5 청기 백기 게임 시작 전 3초 카운트 다운 화면 예시 및 게임 시작 화면**

Figure 5의 예시처럼 카운트 다운이 끝나면 게임이 시작된다. Figure 5 (d)는 본격적으로 게임이 시작되는 화면으로 카운트 다운 화면과 화면 구성이 같다.

**게임 화면**은 Figure 6과 같은 구성을 가진다. 5개의 상태를 줄(line) 단위로 표시하며 각 줄은 구분선('=' 64개로 구성)으로 구분된다. (초기 선택 메뉴의 ***drawLine*** 함수 사용)

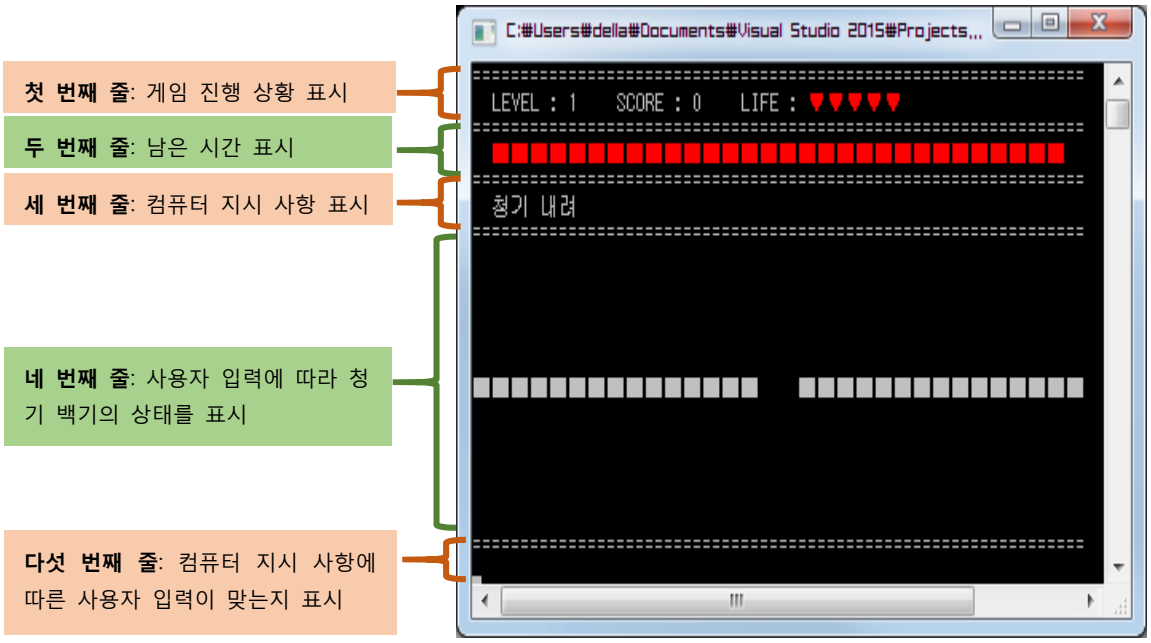



**Figure 6 게임 화면의 구성**

- 게임 화면을 출력하기 위해 반드시 아래의 함수를 정의하고 사용한다.
  - ***display*** : Level, Life, Score, 진행시간, 제한시간, 지시상태, 유저 입력 키 등을 매개변수로 전달 받아 아래에 구현하는 함수들을 이용해 위 Figure들처럼 출력한다.

화면 글자 색상은 Figure들에 나타난 것과 동일하게 구성을 하도록 해야 하며, 글자에 색상을 입히는 구현 방법은 **Tips**의 ***'색상 입히기'를 참조*****한다**.

**첫 번째 줄**은 Level, Score, Life가 출력되도록 한다. 사용자가 컴퓨터의 지시를 따르면 1점, 따르지 않으면 Life가 한 개가 감소된다. Figure 7은 Score와 Life 변화 예시 화면이다. Figure 8에서도 컴퓨터의 '청기 내려' 지시를 따르지 않고 시간이 초과되어 Life가 줄어드는 것을 볼 수 있다.


**Figure 7 Score와 Life 변화 예시**

- Level, Score, Life를 출력하기 위해서는 반드시 아래의 함수를 정의하고 사용해야 한다.
  - ***showState*** : Level, Score, Life를 매개변수로 전달 받아 화면에 이를 출력하도록 한다. (Life는 반드시 for문을 사용하여 출력 할 것.)

**두 번째 줄**은 남은 시간을 나타낸다. (단, ■의 총 개수는 30개이다.) 게임 시작 전에 사용자가 입력한 난이도에 따라 제한 시간이 설정되며 1초가 줄어들 때마다 시간 게이지가 알맞은 비율로 줄어들도록 한다. Figure 7은 Level 1, Figure 8는 Level 2, Figure 10은 Level 3일 때의 시간 게이지가 줄어드는 화면의 예시로 사용자의 입력 없이 시간이 초과되어 생명이 줄어든 경우이다.

- 진행 시간을 출력하기 위해서는 반드시 아래의 함수를 정의하고 사용해야 한다.
  - ***showTime*** : 진행시간, 제한시간을 매개변수로 전달 받아 화면에 알맞은 개수의 ■를 출력하도록 한다. (반드시 for문을 사용하여 출력 할 것.)

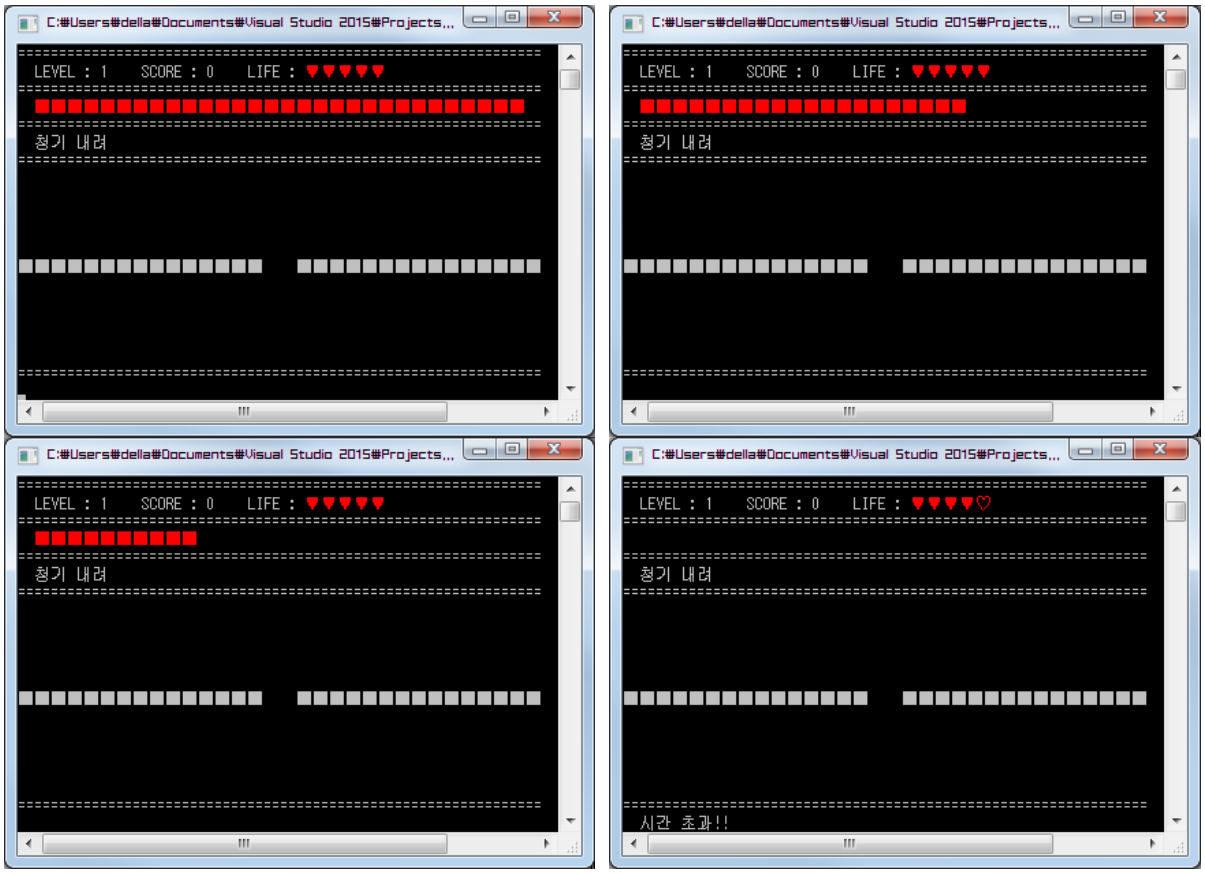



Figure 8 Level 1에서의 시간 게이지 변화 예시


Figure 9 Level 2에서의 시간 게이지 변화 예시


**Figure 10 Level 3에서의 시간 게이지 변화 예시**

**세 번째 줄**은 컴퓨터의 지시를 출력한다. 단, Figure 5에서처럼 게임시작 시에는 카운트 다운 문구에 해당하는 "3초 후 시작합니다"를 출력하도록 한다. 게임이 시작되면 "***청기 올려***', "***청기 내려***', "***백기 올려***', "***백기 내려***', "***청기 올리지마***"를 랜덤으로 출력하도록 한다. (단, 컴퓨터는 프로그램을 실행할 때마다 랜덤하게 지시를 하도록 구현해야 한다.)

- 컴퓨터의 지시사항을 생성하기 위해 반드시 아래의 함수를 정의하고 사용한다.
  - ***getInstruction*** : Random 함수를 이용해 '***e***', '***d***', '***i***', '***k***', ' ' 중 하나를 반환한다.

- 컴퓨터의 지시를 출력하기 위해서는 반드시 아래의 함수를 정의하고 사용해야 한다.
  - ***showInstruction*** : 지시 상태를 매개변수로 전달 받아 화면에 알맞은 지시를 출력한다.

**네 번째 줄**은 사용자의 행위를 시각화한다. 사용자가 아무 것도 하지 않았을 때는 Figure 8, 9, 10과 같이 회색 막대기만 출력하도록 한다. Figure 11은 사용자가 '**e**'키를 눌러 "*청기 올려*"를 수행했을 때, Figure 12는 사용자가 '**d**'키를 눌러 "*청기 내려*"를 수행했을 때, Figure 13은 사용자가 '**i**'키를 눌러 "*백기 올려*"를 수행했을 때, Figure 14는 사용자가 '**k**'키를 눌러 "*백기 내려*"를 수행했을 때이다.

- 깃발을 출력하기 위해서는 반드시 아래의 함수를 정의하고 사용해야 한다.
  - ***showFlag*** : 사용자의 입력을 매개변수로 전달 받아 화면에 알맞은 깃발 상태를 출력한다. 한 깃발의 ■ 개수는 가로 8개, 세로 5개, 깃대는 ■ 15개로 이루어져 있다. 단, 깃발 출력 시, 반드시 2중 for 문을 사용하여 출력하도록 한다.

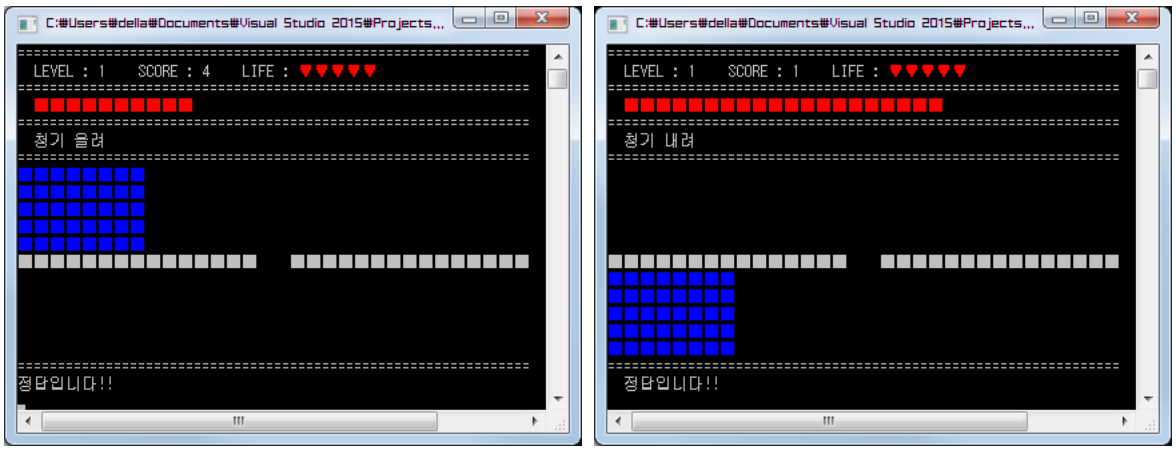



**Figure 11 청기 올려** **Figure 12 청기 내려**

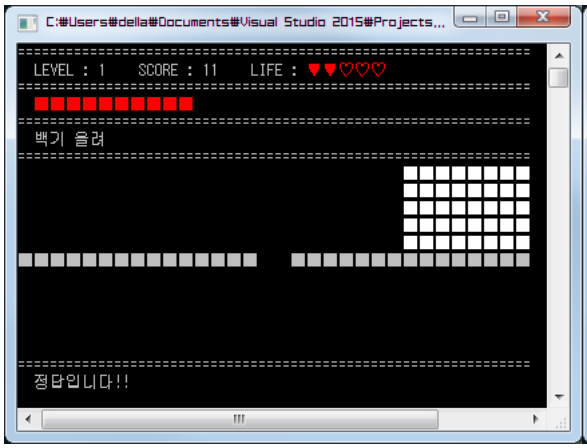



**Figure 13 백기 올려**

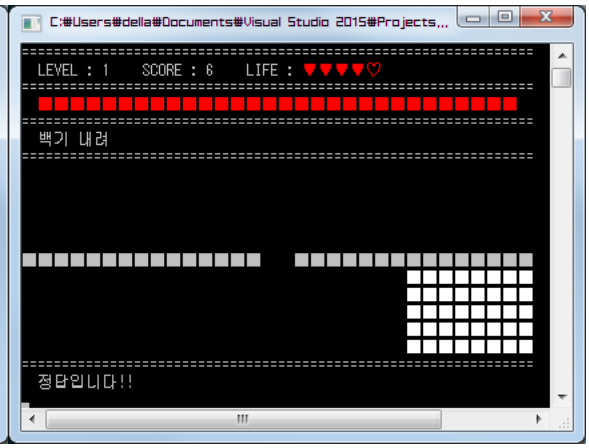



**Figure 14 백기 내려**

**다섯 번째 줄은** 세 번째 줄에 출력된 컴퓨터의 지시에 대하여 사용자가 제한 시간 내에 맞게 행동을 했는지, 즉 지시에 맞는 입력을 했는지를 판단하여 그 결과를 표시한다.

먼저, Figure 15 (a)처럼 컴퓨터의 지시사항이 출력되며 사용자의 입력을 기다리게 된다. 제한 시간 동안 1초마다 시간 게이지가 줄어 들어 다시 출력되며, 사용자 입력을 기다린다. 이 때, 사용자로부터 제한 시간 내에 아무 때나 키보드 입력을 받아서 처리할 수 있도록 해야 한다. 이를 위해 **Tips**(마지막 쪽)의 '***대기 없이 키보드 입력***'과 **Tips**(마지막 쪽)의 '***일시 정지***'부분을 참조한다.

사용자의 입력이 주어진 경우, 바로 그 결과를 Figure 15 (b)처럼 출력하도록 한다. 남은 시간은 키를 입력했을 때의 남은 시간과 동일하게 출력한다. 결과 화면이 출력될 때는 2초동안 결과화면을 출력한 후, 다음 지시사항을 받도록 한다.

사용자가 컴퓨터의 **지시에 대하여 맞게 행동**한 경우, Figure 15처럼 "**정답입니다!!**"가 출력되고, Score가 1 증가한다. **지시와 다르게 행동**한 경우, Figure 16처럼 "**틀렸습니다!!**"가 출력되고 Life가 한 개 감소된다. **시간이 초과된 경우**는 Figure 8, 9, 10처럼 "**시간 초과!!**"가 출력되고, Life가 한 개 감소한다.

- 1초 동안 사용자의 입력을 받기 위해 반드시 아래의 함수를 정의하고 사용해야 한다.
  - ***getUserKey*** : 사용자가 1초 이내에 '***e***', '***d***', '***i***', '***k***' 키를 입력하면 입력한 키를 반환하고, 그렇지 않으면 ' '를 반환한다.


(a) 사용자 입력 대기 화면 (b) 사용자 입력에 대한 결과 화면

**Figure 15 사용자가 정답 키를 눌렀을 때**


**Figure 16 사용자가 오답 키를 눌렀을 때**

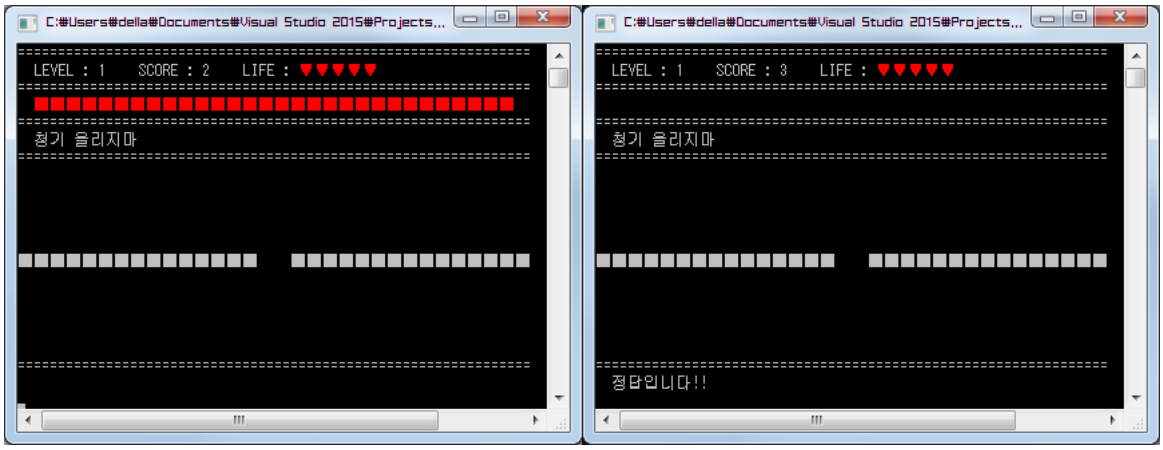



**Figure 17 '청기 올리지마' 정답 처리 화면**

Figure 17은 컴퓨터의 '청기 올리지마' 지시에 대해서 사용자가 제한시간 동안 아무 키도 입력하지 않아서 정답이 맞은 경우에 대한 출력화면으로 "정답입니다!!"출력과 score가 1 증가한 것을 볼 수 있다.

- 정답, 오답, 시간 초과 처리를 위해 아래의 함수를 정의하고 사용한다.
  - ***judge*** : 지시상태, 유저 입력 키, 남은 시간을 매개변수로 전달 받아 게임 상황을 반환한다. 0은 계속 진행, 1은 오답, 2는 정답, 3은 시간 초과로 매칭한다.

## IV. 게임 종료

Life가 모두 소진되면 게임이 종료된다. 게임 화면을 지우고 아래와 같이 "GAME OVER", "몇 점으로 끝났습니다." 라는 문구를 출력한 후, 메뉴를 다시 출력한다. Figure 18은 Life가 한 개 남은 상황에서 오답을 입력해 Life가 모두 소진되어 게임이 종료된 상황이다. 11점으로 게임이 끝나 게임 화면이 지워지고 11점으로 끝났다는 문구와 메뉴가 다시 출력되었다.


**Figure 18 게임 종료 화면 예시**

## V. 최고 점수 확인

메뉴에서 2를 입력했을 때, 최고점수는 몇 점인지 출력하고 메뉴를 다시 출력하도록 한다. Figure 19는 게임이 한번도 진행되지 않았을 경우의 최고 점수 확인 화면 예시이다. Figure 20은 10점이 최고 점수일 때, 2점으로 끝나도 최고점수를 10점으로 출력하는 예시이다.

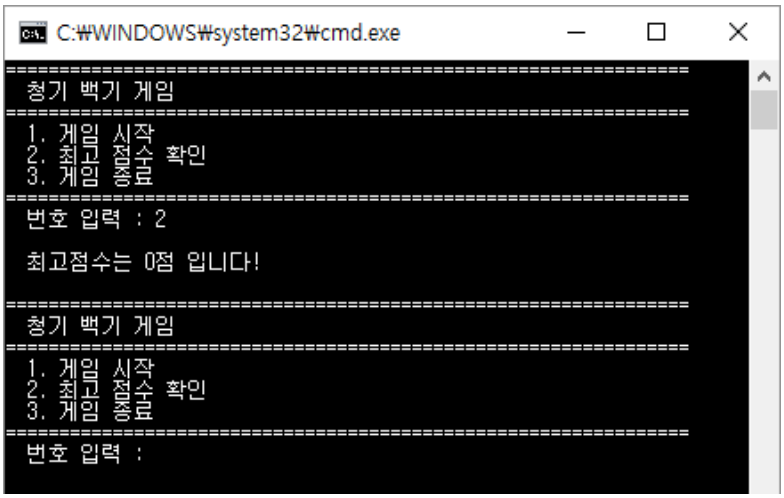



**Figure 19 게임이 한번도 진행되지 않았을 경우 최고 점수 확인 화면 예시**

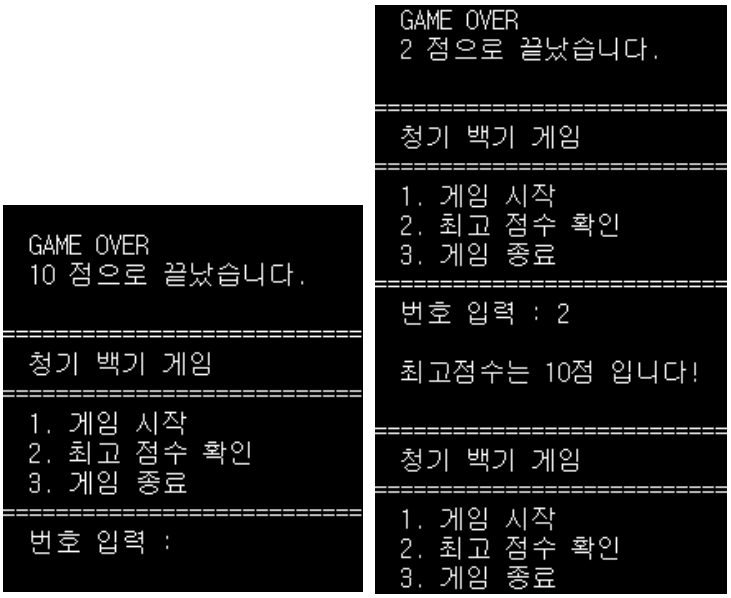



**Figure 20 게임이 몇 번 진행된 후의 최고 점수 확인 화면 예시**

## 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;
}
```

- **색상 입히기**

<Windows.h>를 포함한 뒤, 아래 코드와 같이 SetConsoleTextAttribute() 함수를 실행해보고 색 지정이 어떻게 이뤄지는지 파악한다.

```c
#include <stdio.h>
#include <Windows.h>
int main()
{
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 9); // 파랑
	printf("이 텍스트는 파란색입니다. \n");
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 12); // 빨강
	printf("이 텍스트는 빨간색입니다. \n");
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 7); // 밝은 회색
	printf("이 텍스트는 회색입니다. \n");
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 15); // 흰색
	printf("이 텍스트는 흰색입니다. \n");
	return 0;
}
```

- **일시 정지**

<time.h>를 포함한 뒤, 변수 sec의 값을 바꾸어보면서 결과가 어떻게 출력되는지 확인해본다.

```c
#include <stdio.h>
#include <time.h>
int main()
{
        clock_t st;
        int sec = 1;
        printf("HELLO WORLD\n");
        st = clock();
        while(clock() - st < sec*1000) {}
        printf("HELLO WORLD2\n");
}
```

**clock_t clock(void)**: 프로그램 실행 후 경과된 시간을 리턴하는 함수

- 이 함수는 시스템으로부터 현재 시간을 millisecond 단위로 가져오며 1초는 1000 밀리세컨드가 된다.
- <time.h>에 포함되어 있음

clock_t 형은 long 형으로 time.h 에 정의되어 있다.

- **대기 없이 키보드 입력**

scanf를 사용하여 키보드 입력을 받게 되면 입력 라인에서 프로그램이 멈추게 된다. 본 프로그램은 게임의 형식을 취하고 있기 때문에 실시간 입력이 가능해야 한다. 아래의 두 함수를 응용하여 키보드 입력을 받을 수 있다.

**int _kbhit(void)**: 키 입력을 체크하는 함수

- 키 입력이 있으면 0 이외의 값을, 아니면 0을 리턴 한다.
- <conio.h>에 포함되어 있음

**int _getch(void)**: 화면에 출력 없이 키보드로부터 하나의 문자를 입력 받는 함수

- 엔터를 누르지 않고도 키보드 입력을 받을 수 있고, 누른 키가 화면에 출력되지 않기 때문에 게임을 엔터키 없이 진행할 수 있다.
- 누른 키보드의 값을 리턴한다.
- <conio.h>에 포함되어 있음