

챕터14. 콘솔 프로그래밍

1절. 콘솔 제어

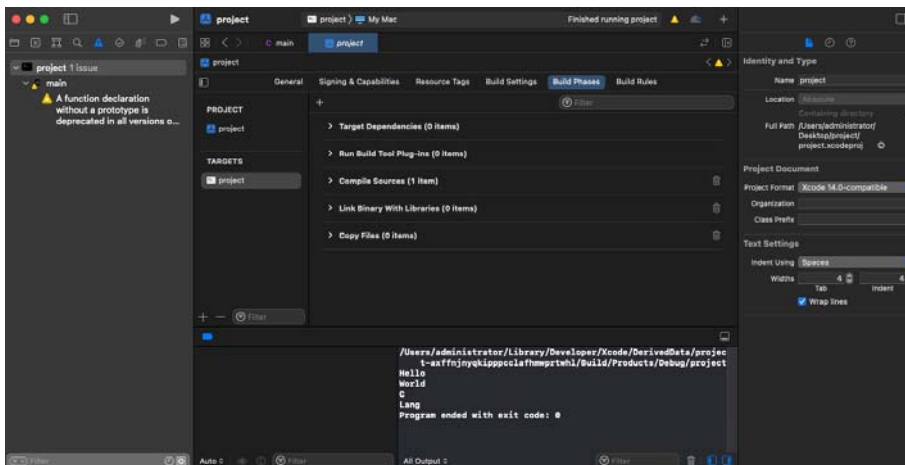
핵심 키워드	ncurses, 콘솔 색, 커서
여기서는 무얼 배울까	<p>콘솔은 문자만으로 모든 정보를 표현하지만, 문자 그 자체가 나타내는 의미만을 사용하지는 않는다. 컴퓨터는 문자의 위치와 색, 특수 기호, 심지어는 문자로 이루어진 이미지까지 다양한 방법을 통해 사용자에게 정보를 제공한다. 이번 절에서는 콘솔에 추가적인 정보와 함께 시각적으로 아름답게 콘솔을 꾸밀 수 있는 색과 커서 제어를 배우고 이를 실습한다.</p>

1. ncurses

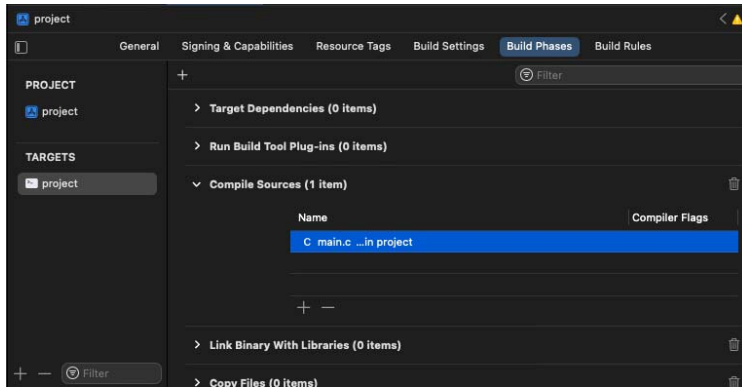
리눅스에는 콘솔의 제어를 손쉽게 도와주는 라이브러리인 ncurses가 있다. macOS의 Xcode는 ncurses를 사용할 수 있도록 해당 라이브러리가 기본적으로 내장되어 있고, ncurses.h 헤더 파일을 include하면 여기에 포함된 다양한 함수들을 사용할 수 있다. 이러한 함수들은 한 번의 호출로 다양한 콘솔의 제어를 도와준다.

(1) ncurses를 포함하여 빌드

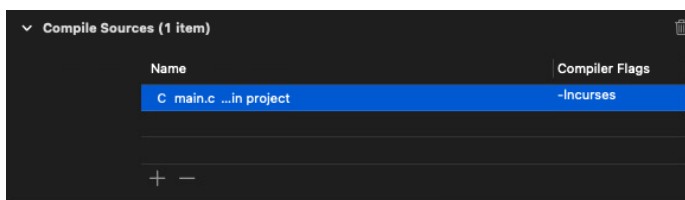
Xcode 프로젝트는 기본적으로 ncurses가 포함되지 않는다. 따라서 먼저 ncurses 라이브러리를 사용한다는 것을 지정해주어야 빌드하는 과정에서 ncurses를 함께 링크하여 실행 파일을 만든다.



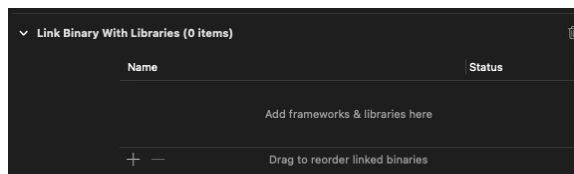
① 네비게이터의 프로젝트명 → TARGETS → 프로젝트명 → Build Phases를 누른다.



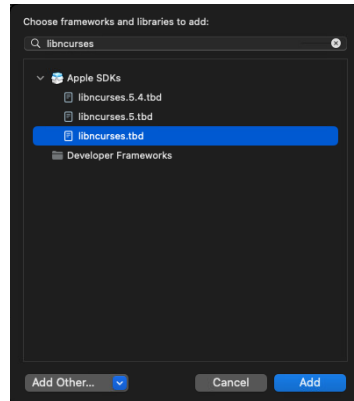
② Compile Sources에서 ncurses.h의 함수가 포함되는 파일을 선택한다.



③ Compiler Flags에 -Incsures를 입력한다.



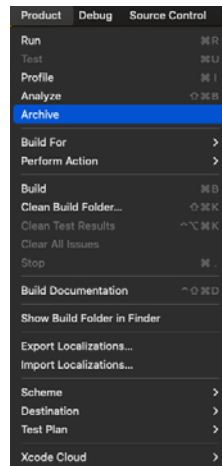
④ Compile Sources 아래의 Link Binary With Libraries를 연 다음 +를 누른다.



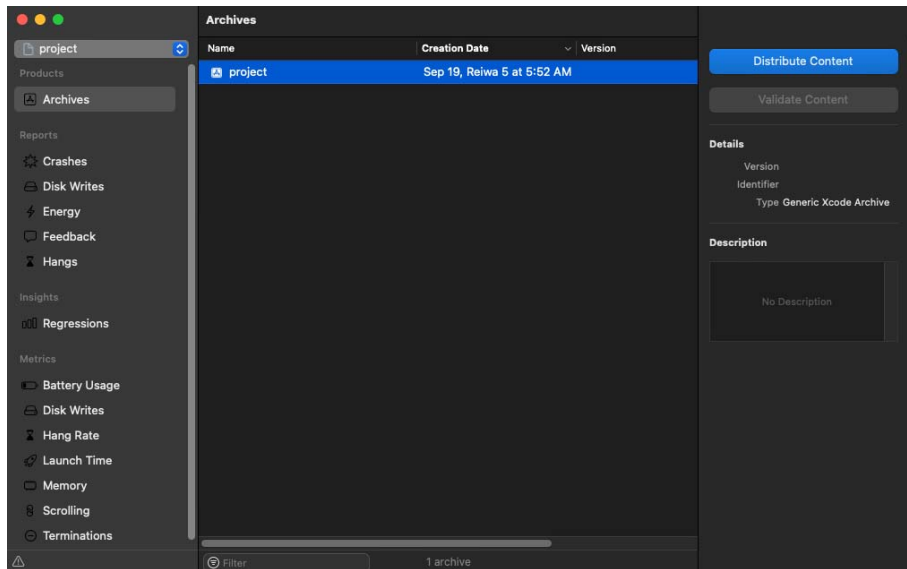
⑤ libncurses를 검색하여 libncurses.tbd를 추가한다.

다음 과정을 따랐다면 이제 프로젝트에 ncurses 라이브러리가 포함되어 함께 빌드된다. 하지만 이 상태만으로는 빌드된 프로그램을 실행시킬 수 없으므로 실행 파일을 만드는 과정도 함께 설명한다.

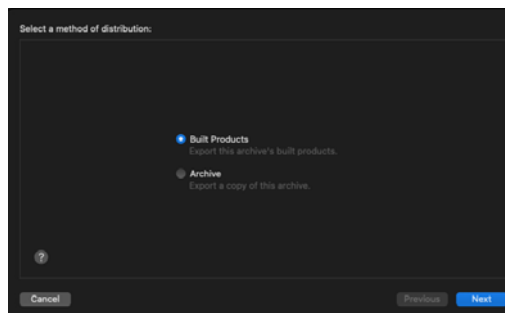
DELL 2023/09/19 20:01
TIP
 실행 없이 빌드만 수행하는 단축키는 **Command + B**입니다.



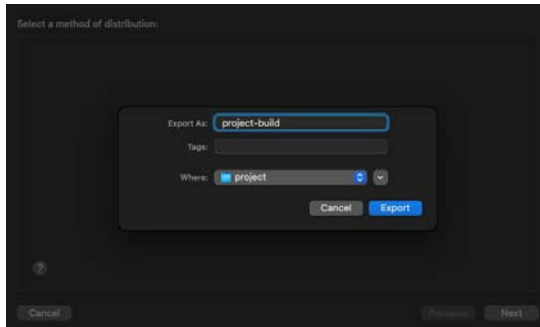
① 상단 메뉴의 Product → Archive를 선택한다.



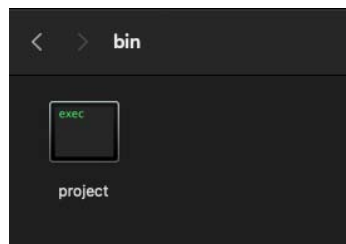
② 우측 상단의 **Distribute Content**를 클릭한다.



③ **Build Products**를 선택한 후 **Next**를 클릭한다.



④ 실행 파일이 저장될 폴더의 이름과 경로를 설정한 후 **Export**를 누른다.



⑤ 지정한 경로로 이동하여 bin 폴더에 도달하면 빌드의 결과인 실행 파일이 존재한다. 이 실행 파일을 실행하면 C 언어로 만든 프로그램이 실행된다.

(2) 스크린 생성과 출력

손으로 익히는 코딩
<pre>#include <ncurses.h> #include <stdio.h> int main() { initscr(); printw("Hello, World!"); refresh(); getch(); endwin(); }</pre>

ncurses로 구성된 프로그램은 스크린을 초기화하는 `initscr()`, 프로그램이 끝날 때 실행되는 `endwin()`으로 이루어진다.

손으로 익히는 코딩

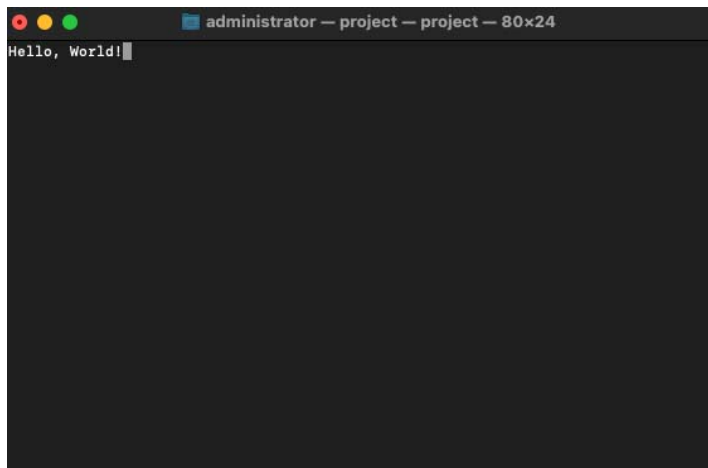
<pre>printw("Hello, World!"); refresh();</pre>
--

ncurses는 기존의 `printf()` 대신 `printw()`로 문자를 기록하고 `refresh()`로 기록한 문자를 출력한다. `refresh()`가 없다면 화면에 문자가 보이지 않는다.

손으로 익히는 코딩

<pre>getch();</pre>

출력한 이후에는 입력을 받는 함수 하나가 필요하다. ncurses는 마치 GUI 운영체제에서 새로운 창을 키듯 빈 화면을 연 다음 그 화면에 문자를 출력한다. 프로그램이 종료되면 ncurses에 의해 열린 창이 즉시 닫혀 결과가 보이지 않는다.



프로그램을 실행하면 빈 화면에 Hello, World!가 출력되고, 이 상태에서 아무 키나 누르면 프로그램이 종료되어 출력된 문자가 사라진다.

2. 콘솔 색 제어와 문자 속성

색을 포함한 문자의 속성은 문자에 담긴 의미와 더불어 사용자가 한눈에 추가적인 정보를 알 수 있게 한다. 붉은 글자는 위험을, 푸른 글자는 안전을 나타내는 것처럼 말이다. ncurses는 문자와 배경 색을 묶어 색을 변경하거나, 굵음과 같은 문자의 속성을 지정할 수 있다.

(1) 색상

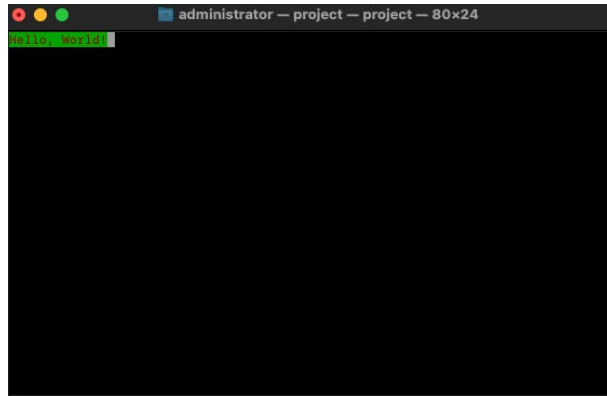
코드 소개
<pre>start_color(); init_pair(번호, 문자색, 배경색); attron(COLOR_PAIR(번호))</pre>

색을 바꾸려면 프로그램의 시작 부분에서 start_color()를 실행한 후 init_pair()를 통해 특정한 문자, 배경 색 조합에 번호를 부여한다. attron()은 COLOR_PAIR를 통해 init_pair()로 만들어진 색 조합을 사용한다.

정수	열거형
0	COLOR_BLACK
1	COLOR_RED
2	COLOR_GREEN
3	COLOR_YELLOW
4	COLOR_BLUE
5	COLOR_MAGENTA
6	COLOR_CYAN
7	COLOR_WHITE

init_pair()로 사용할 수 있는 색은 8가지로, 각각의 색은 정수 또는 열거형으로 지정할 수 있다.

손으로 익히는 코딩
<pre>#include <ncurses.h> #include <stdio.h> int main() { initscr(); start_color(); init_pair(1, COLOR_RED, COLOR_GREEN); attron(COLOR_PAIR(1)); printw("Hello, World!"); refresh(); getch(); endwin(); }</pre>



다음 예제를 출력하는 콘솔에 초록색의 배경에 빨간 글씨로 Hello, World!가 출력된다. 콘솔 색은 문자를 출력하는 시점에 지정되므로 줄 바꿈을 통해 다음 줄로 넘어간다면 오른쪽 부분은 색이 변하지 않는다.

(2) 굵음

특정한 글자를 강조하는 용도로 굵음, 볼드를 적용할 수 있다. 색을 적용하는 것과 비슷하게 `attron()`을 이용한다.

손으로 익히는 코딩

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

int main()
{
    initscr();

    attron(A_BOLD);
    printw("Hello");
    attroff(A_BOLD);
    printw(", World!");
    refresh();

    getch();
    endwin();
}
```


A_BOLD는 함수 호출 이후부터 출력하는 모든 문자에 굵음을 적용한다. 특정 옵션 적용을 끝내길 원한다면 attroff()로 종료한다.



다음 코드를 실행하면 Hello만 굵게 표시되고, 나머지 글자인 World!는 일반적인 두께로 표시된다.

3. 커서 제어

(1) 커서 이동

콘솔에서 문자를 출력하는 것은 콘솔의 커서가 위치하는 곳에 문자를 쓰고, 커서를 한 칸 이동하는 것을 의미한다. 그러므로 글을 쓰듯 왼쪽에서 오른쪽으로 문자를 출력하는 것이 아닌 내가 원하는 위치에 문자를 출력하기 위해선 커서를 옮겨야 한다.

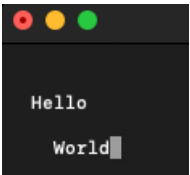
코드 소개
<code>move(행, 열);</code>
<code>mvprintw(행, 열, 문자열);</code>

ncurses는 move()와 mvprintw()로 커서의 위치를 이동시킨다. move()는 커서의 위치만을 이동시키고, mvprintw()는 커서를 이동시킨 후 문자열을 출력한다.

손으로 익히는 코딩
<pre>#include <ncurses.h> #include <stdio.h> int main() { initscr(); move(2, 2); printw("Hello"); mvprintw(4, 4, "World"); refresh(); getch(); endwin(); }</pre>

```
}
```

위 예제는 2, 2 위치에 Hello를 출력하고 4, 4 위치에 World를 출력한다.



항상 문자가 왼쪽 위부터 출력되던 것과는 달리, 이제 원하는 위치에 문자를 출력할 수 있게 되었다. 문자의 출력 위치를 지정할 수 있다는 것은, 이미 출력된 문자를 자유롭게 수정하고, 문자의 위치 자체에 특별한 의미를 부여하여 사용할 수 있다는 것을 말한다.

(2) 커서 숨기기

커서는 사용자가 콘솔의 어느 부분에 출력할지를 알려주는 역할을 하지만, 이를 알 필요가 없을 때가 있다. 이 경우 커서가 깜빡이면서 그 위치를 표시하는 것은 오히려 사용자가 거슬러할 수도 있다.

코드 소개
<code>curs_set(FALSE);</code>

curs_set()은 커서를 보이게 하거나 보이지 않게 설정한다. FALSE는 커서가 보이지 않고, TRUE는 커서가 보인다.

손으로 익히는 코딩
<pre>#include <ncurses.h> #include <stdio.h> int main() { initscr(); curs_set(FALSE); printf("Hello, World!"); refresh(); getch(); }</pre>

```
endwin();
}
```

실행 결과
Hello, World!

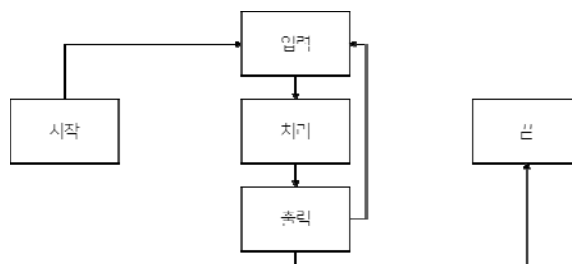
curs_set()으로 커서를 숨기면 더 이상 깜빡이는 커서 없이 문자만이 화면에 표시된다.

2절. 게임 루프와 입력 처리

핵심 키워드	대화형 프로그램, 게임 루프, getch(), nodelay()
여기서는 무얼 배울까	
<p>먼 옛날부터 지금에 이르기까지 인류는 다양한 오락을 발명하고 행해왔다. 컴퓨터의 역사 역시 오락과의 관계를 뗄 수는 없을 것이다. 디지털 컴퓨터가 등장하고서부터 지금까지 다양한 게임들이 개발됐고, 여기에는 콘솔에서 플레이하는 콘솔 게임들도 있다. 이번 절에서는 콘솔 게임을 만들기 위한 기본적인 지식을 배우고 이를 바탕으로 하는 예제들을 살펴본다.</p>	

1. 게임 루프

모든 프로그램은 시작과 끝이 있다. 프로그램이 시작된 후 사용자로부터 입력을 받아 적절한 처리를 하고, 그에 맞는 결과를 출력한다. 입력과 처리, 그리고 출력으로 이루어진 기본적인 프로그램은 이와 같은 형태를 가지고 있지만 여러분들이 사용하는 많은 프로그램들은 프로그램이 실행되고 나서 입력과 처리, 출력이 무한히 반복되는 형태였을 것이다.



위 이미지처럼 프로그램이 시작되고 나서 사용자의 입력을 처리하고, 그에 맞는 결과를 출력하는 일련의 작업이 무한히 반복되게 하려면 프로그램 내에 프로그램이 종료되기 전까지 반복되는 반복문이 필요하다. 이 반복문은 입력과 처리, 출력을 반복함으로써 사용자가 프로그램과 마치 대화하듯 입력과 출력을 주고받거나, 연산이 필요할 때마다 새로운 프로그램을 실행할 필요 없이 기존의 프로그램을 바로 사용할 수 있게 한다.



계산기 역시 이러한 사이클을 무한히 반복한다. 버튼을 클릭하면 클릭한 버튼에 맞게 숫자를 추가하거나 계산하고, 그에 맞는 결과를 계산기의 화면에서 보이도록 한다.

디지털 게임도 프로그램의 일종이므로 사용자와 프로그램이 대화형으로 진행하기 위해선 프로그램 내부에 계속해서 반복되는 반복문이 필요하다. 게임에서는 이 반복을 게임 루프라고 한다. 게임 루프는 여러 형태가 있지만 기본적으로는 입력, 처리, 출력을 반복하고, 이 세 단계를 합쳐 프레임 또는 턴이라고 말한다.

(1) 입력

게임을 비롯한 대화형 프로그램은 매 프레임마다 입력 여부를 확인한다. 만약 입력이 있다면 입력에 맞는 추가적인 처리를 할 수 있는데, 여기에는 키를 누르거나 뺐는지, 다중 키를 입력했는지 등이 있다.

(2) 처리 (업데이트)

입력이 끝났다면 그에 맞는 처리를 수행한다. 대화형 프로그램은 사용자의 입력을 기다리지 않는다는 특징이 있기에 처리 단계에서 입력이 없을 수도 있다. 이러한 경우 아무런 처리를 하지 않을 수도, 또는 이전의 입력에 영향을 받은 처리를 마저 수행할 수도 있다. 이러한 특징 때문에 처리를 업데이트라고 부르는 경우가 많다.

(3) 출력 (렌더링)

입력과 처리 이후에는 사용자에게 그 결과를 알려주는 출력 작업이 있다. 출력의 형태에는 문자, 소리 등 여러 가지가 있지만 대표적인 것은 사용자의 화면에 결과를 그려내는 것이다. 컴퓨터의 화면에 무언가를 그려내는 것은 렌더링이라고 하며, 게임 루프에선 출력 대신 렌더링이란 단어를 사용해 입력, 업데이트, 렌더링의 세 단계로 표현하기도 한다.

DELL 2023/08/31 13:40
기초 용어 정리
대화형 프로그램
 프로그램이 사용자와 입력과 출력을 대화하듯 주고받는 형태의 프로그램

DELL 2023/09/01 19:17
기초 용어 정리
게임 루프
 게임에서 입력, 업데이트, 렌더링의 세 단계가 반복되는 것

2. ncurses의 입력

자신이 입력한 것을 엔터를 쳐서 확정 짓기 전까지 수정할 수 있다는 것은 많은 장점이 있다. 그러나 만약 여러분들이 게임을 만들고자 한다면, 혹은 단축키와 같은 기능을 만들고자 한다면 엔터 없이 키를 입력하는 순간 어떠한 연산이 이루어지는 것이 더 나을 수 있다. ncurses는 보다 다양한 종류의 입력을 위한 여러 함수를 제공한다.

(1) getch()

getch()는 사용자로부터 키보드 입력을 받는 즉시 값을 반환하는 함수이다.

손으로 익히는 코딩
<pre>#include <ncurses.h> #include <stdio.h> int main() { char str[128]; initscr(); curs_set(FALSE); keypad(stdscr, TRUE); while (1) { int x = getch(); sprintf(str, "%d\n", x); printw(str); refresh(); } endwin(); }</pre>

입력과 실행 결과
a
97

위 예제는 키보드를 입력하는 순간 그 키에 해당하는 코드를 출력한다. 예제에 포함된 keypad()는 키보드의 문자 입력과 함께 방향키와 같이 문자가 아닌 키도 입력받을 수 있게

한다.

코드 소개

<code>noecho();</code>

키보드로 입력한 문자가 화면에 출력되지 않도록 하려면 출력 전 `noecho()`를 호출한다. 위 예제에서 문자를 입력받기 전 해당 함수를 호출하면 문자 없이 문자에 해당하는 코드만 출력됨을 확인할 수 있다.

(2) 입력을 기다리지 않기

우리가 지금까지 사용했던 모든 입력 함수들은 사용자로부터 입력이 오길 기다리고, 기다리는 중에는 다른 연산을 수행하지 않는다. `ncurses`는 몇 가지 함수를 통해 입력의 유무를 검사하고, 입력이 없다면 입력을 더 기다리지 않고 자신이 해야 하는 연산을 수행하도록 할 수 있다.

코드 소개

<code>nodelay(stdscr, TRUE);</code>

`nodelay()`를 키면 `getch()`가 입력을 기다리지 않는다. 만약 입력이 존재한다면 그에 해당하는 코드가 반환되고, 입력이 없다면 `ERR`가 반환된다.

손으로 익히는 코딩

<pre>#include <ncurses.h> #include <stdio.h> int main() { char str[128]; int cnt = 0; initscr(); curs_set(FALSE); noecho(); keypad(stdscr, TRUE); nodelay(stdscr, TRUE); while (1) {</pre>

```

int x = getch();
if (x != ERR)
{
    sprintf(str, "%d\n", cnt);
    printw(str);
    refresh();
}
cnt++;
}

endwin();
}

```

위 예제는 프로그램이 시작한 후부터 지금까지 몇 프레임이 지났는지를 cnt에 저장하고, 키보드 입력이 들어올 때마다 cnt를 출력한다. nodelay()로 인해 키보드 입력이 없더라도 반복문의 cnt++가 계속해서 호출되므로, 입력할 때마다 큰 폭으로 cnt가 증가함을 확인할 수 있다.

연습문제 : Windows 버전과 동일

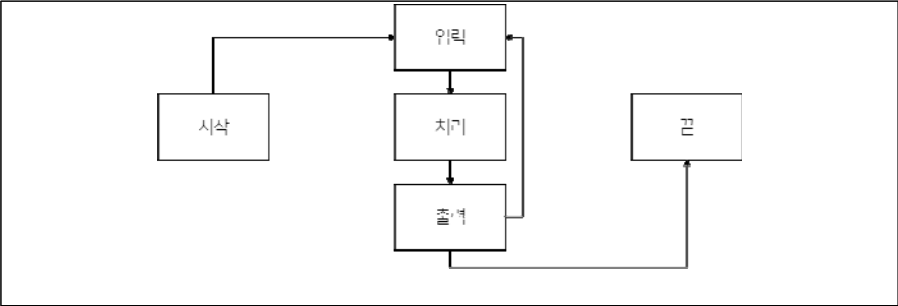
한 장으로 요약 정리

키워드로 정리하기

- 리눅스와 macOS는 **ncurses** 라이브러리를 통해 콘솔을 제어하여 CLI 프로그램을 손쉽게 만들 수 있다.
- ncurses는 프로그램의 시작을 **initscr()**, 끝을 **endwin()**로 나타낸다. ncurses의 출력은 **printw()**를 이용하며 화면 출력을 위해선 **refresh()**가 필요하다.
- ncurses로 콘솔 색을 바꾸고 문자에 속성을 설정하는 함수는 **attron()**이며 커서를 이동시키는 함수는 **move()**, **mvprintw()**가 있다.
- **getch()**는 ncurses가 제공하는 입력 함수로 **noecho()**, **keypad()**, **nodelay()** 등과 함께 사용할 수 있다.
- 대화형 프로그램과 같이 사용자와 컴퓨터가 지속해서 입출력을 주고받는 경우 입력, 처리, 출력의 사이클을 내부적으로 반복하여 실행한다.
- 게임의 경우 특히 처리를 업데이트, 출력을 렌더라고 부르는 경우가 많으며, 이러한 반복을 **게임 루프**라고 한다.

그림으로 정리하기

오늘날의 많은 프로그램은 내부적으로 입력, 처리, 출력의 사이클을 반복하는 반복문이 있고, 이를 간단히 그림으로 나타내면 이렇다.



[3쪽] [메모:3] DELL

2023/09/19 20:01

TIP

실행 없이 빌드만 수행하는 단축키는 **Command + B**입니다.

[12쪽] [메모:1] DELL

2023/08/31 13:40

기초 용어 정리

대화형 프로그램

프로그램이 사용자와 입력과 출력을 대화하듯 주고받는 형태의 프로그램

[12쪽] [메모:2] DELL

2023/09/01 19:17

기초 용어 정리

게임 루프

게임에서 입력, 업데이트, 렌더링의 세 단계가 반복되는 것