

콘솔 서바이벌(Console survival)-최종 보고서

낭만 팀장 양준서팀- 양준서, 김영인, 황윤희, 장임욱

## 1. 개발 배경 및 필요성

평소에 머릿속에서 구상하고 있던 아이디어를 혼자 개발하기에는 많은 어려움이 있었다. 그래서 팀원들이 구상하고 있던 아이디어를 조각조각 모아 이번 프로젝트를 기회로 활용하여 구현해보기로 했다.

## 2. 개발 환경 및 목표

개발 환경은 콘솔 기반으로 C++(Visual Studio)를 활용하였다. 우리들의 목표는 게임을 완성시키는 것뿐만 아니라 플레이어들의 피드백을 수용해 더 좋은 게임으로 발전시키는 것이다.

## 3. 기존에 존재하는 관련 내용들

플레이어 움직임 동작을 다음과 같은 느낌으로 구현할 것이다.

<https://youtu.be/x8sX5Xhjv1Y?si=N2355lHTgQRF65Ky>

플레이어 움직임 처리& 인 게임 화면은 다음과 같은 화면을 생각하고 있다.

[https://youtube.com/shorts/2skDKT\\_al2s?si=zTrsia6k4J2QeOfH](https://youtube.com/shorts/2skDKT_al2s?si=zTrsia6k4J2QeOfH)

## 4. 게임 플레이 방법

- 키보드 ↑ ↓ ← → 방향키로 플레이어를 조작할 수 있습니다.
- 매 턴마다 랜덤으로 숫자가 부여되고, 해당 숫자만큼 이동할 수 있습니다.
- 이동이 끝나면, 마지막으로 움직인 방향으로 자동으로 공격이 나갑니다.
- 적의 공격은 \*\*플레이어가 움직였을 때만\*\* 동작합니다.

(즉, 플레이어가 가만히 있으면 적도 공격하지 않습니다!)

- 제한된 시간 동안 몬스터를 처치하며 최대한 많은 점수를 획득하세요.

## 5. 스케줄 및 역할 분담

중간고사 이전 기획을 완료하였고 중간고사 이후 개발을 시작하여 약 8~13주차까지 개발을 진행하였다. 이후 최종 보고서 마감일인 6/8까지 계속 기능 보완 및 밸런스 디자인하였다.

황윤희- 플레이어 움직임 및 공격 기능 구현, UI 수정 및 보완

양준서- 게임 시작화면, 랭킹 시스템, 타이머, 맵 드로우, 게임 방법 페이지

김영인- 전체적으로 코드 개발을 담당하였고 팀원들이 각자 구현한 기능을 한 코드로 합치는 일을 함. +타이머, 몬스터 생성 및 몬스터 공격, 플레이어 공격과 몬스터 상호작용, 몬스터 공격과 플레이어 상호작용.

## 6. 주요 기능 및 함수

### main.cpp

-프로그램 시작점으로, 전체 게임 루프를 실행하고 각 모듈을 초기화/호출하는 역할을 담당합니다.

### map.cpp

- 게임 맵 출력 및 UI 관련 기능 담당
- 타이머/점수/랭킹 출력, 몬스터/플레이어/총알 표시 기능 포함
- 타이틀 효과 및 클리어 효과도 담당

### Monster.cpp

- 몬스터 생성, 이동, 공격 기능 담당
- 몬스터는 일정 주기로 출현하며, 이동 후 플레이어를 공격하는 로직 포함

### Timer.cpp

- 게임 제한 시간 기능 담당
- 타이머 시작/중지/잔여 시간 조회 기능 포함

### attack.cpp

- 몬스터의 총알(공격)을 발사하는 기능을 담당
- 총알 위치 이동 및 몬스터와의 충돌 처리

### player.cpp

- 플레이어 이동 처리
- 플레이어 위치 초기화 및 점수 관리(가산/초기화) 기능 포함

### -멀티스레드(std::thread) 사용 이유

본 프로젝트에서는 타이머, 몬스터 생성/이동/공격, 플레이어 조작, 맵 그리기 등 여러 기능이 실시간으로 동시에 실행되어야 자연스러운 게임 플레이가 가능하다.

이를 위해 `std::thread`를 도입하여 `timer_thread`(타이머 기능)와 `monster_thread`(몬스터 동작)를 독립 thread로 실행하였다.

이를 통해 메인 루프에서는 플레이어 입력과 맵 출력에 집중하고, 타이머 및 몬스터 동작은 별도 thread에서 병렬로 처리하였다. 결과적으로 한 번의 실행으로 여러 기능들이 한꺼번에 실행 가능하게 되었다.

### -mutex 사용 이유

게임 환경에서 콘솔 출력 시 출력 경합(충돌) 문제가 발생하였다. 예를 들어 점수, 타이머, 게임 메시지가 동시에 출력될 때 출력 위치가 겹치거나 화면이 깜빡이는 현상이 나타났다.

초기에는 `bool` 변수를 사용하여 출력 타이밍을 조절하려 했으나, 출력 항목이 많아지면서 코드 관리가 어려워졌다.

이에 `std::mutex`를 `output_mutex`라는 이름으로 선언하고, `lock_guard<std::mutex> lock(output_mutex)` 구문을 활용하여 출력 블록을 보호하도록 구현하였다.

결과적으로 콘솔 화면 출력이 안정적이고 깔끔하게 유지되어 오류를 줄일 수 있었다.

## 7. 최종 결과물 사진



## 8. 개발 과정에서 어려운 점과 해결방법

### 기술적 문제 ①: 콘솔 출력 시 문자열 줄 맞춤 깨짐 문제

#### • 문제 발생 원인:

콘솔에서 출력 시 영문자는 1칸, 한글 및 일부 특수문자는 2칸으로 표시됨. 하지만 기본 출력 함수는 이를 구분하지 않기 때문에 표나 UI를 출력할 때 줄 맞춤이 깨지는 현상이 발생.

#### • 해결 과정:

문자열의 실제 표시 폭을 계산하는 `getDisplayWidth(const string& text)` 함수를 구현하여 출력 전 문자열 폭을 정확히 계산하도록 개선.

이 함수는 각 문자에 대해 상위 비트를 검사하여 1칸 또는 2칸으로 누적하여 너비를 계산함.

#### • 적용 예:

메뉴 출력, 게임 정보 UI, 플레이어 상태 표시 등에서 문자열이 한글과 영문 혼용 시에도 안정적으로 정렬되도록 적용.

#### • 결과:

게임 플레이 화면에서 줄 맞춤과 UI 정렬이 정상적으로 출력됨.

### 기술적 문제 ②: 콘솔 출력 시 출력 충돌(겹침) 문제

#### • 문제 발생 원인:

게임 화면에서 남은 시간, 몬스터 위치, 몬스터 공격 효과 등 다양한 정보를 동시

에 출력해야 했다.

하지만 콘솔에서는 각 기능의 출력이 별도의 스레드에서 동작하거나 빠르게 반복 출력되는 경우가 많아, 출력 시점이 겹치면 기존 출력 내용을 덮어쓰는 문제가 발생하였다.

특히, 몬스터 생성 위치에 남은 시간이 출력되거나, 몬스터 공격 자리에 남은 시간이 덮어쓰지는 현상이 반복적으로 발생하였다.

- **해결 과정:**

모든 출력 작업에 **\*\*mutex(뮤텍스)\*\***를 적용하여 출력 순서를 통제하였다.

mutex lock을 통해 한 번에 하나의 기능만 출력하도록 설정하였고, 출력 중 다른 기능은 대기하도록 구성하였다.

이를 통해 동시에 여러 출력이 이루어지지 않도록 하여 출력 충돌 문제를 해결하였다.

- **적용 예:**

남은 시간 표시, 몬스터 위치 출력, 몬스터 공격 효과 출력 등 콘솔에서 시각적으로 지속적으로 변화하는 모든 출력 기능에 mutex를 적용하여 안정적으로 표시되도록 구현.

- **결과:**

게임 플레이 도중 화면 출력이 깔끔하게 유지되고, 각 기능의 출력 내용이 서로 덮어쓰지 않도록 정상적으로 표시됨.

## 9. 기대 효과 및 활용 분야

- 유니티, godot등의 게임 엔진을 활용해 2D나 3D게임으로 구체화하여 수익을 낼 수 있다.

- 게임을 만들기 전 기본적인 알고리즘과 기획 등 틀에 대해 이해할 수 있고 게임을 만듦과 동시에 C++언어에 대한 능력이 향상될 수 있다. 또한 팀 프로젝트인만큼 팀워크의 중요성과 관련 경험을 쌓을 수 있었다.