**C++프로그래밍**

**프로젝트**

|  |  |
| --- | --- |
| 프로젝트 명 | *Snake Game* |
| 팀 명 |  |
| 문서 제목 | 결과보고서 |

|  |  |
| --- | --- |
| **Version** | 1.4 |
| **Date** | 26 |

|  |  |
| --- | --- |
| **팀원** | 박우혁 (팀장) |
| 김민수 |
|  |
|  |
|  |

|  |
| --- |
| **CONFIDENTIALITY/SECURITY WARNING**  이 문서에 포함되어 있는 정보는 국민대학교 소프트웨어융합대학 소프트웨어학부 및 소프트웨어학부 개설 교과목 C++프로그래밍 수강 학생 중 프로젝트 “Snake Game”를 수행하는 팀 “xxxxx”의 팀원들의 자산입니다. 국민대학교 소프트웨어학부 및 팀 “xxxxxx”의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다. |

**문서 정보 / 수정 내역**

|  |  |
| --- | --- |
| **Filename** | 최종보고서-프로젝트명.doc |
| **원안작성자** | 박우혁 |
| **수정작업자** |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 수정날짜 | 대표수정자 | Revision | 추가/수정 항목 | 내 용 |
| 2023-05-19 | 박우혁 | 1.0 | 최초 작성 |  |
| 2023-05-19 | 박우혁 | 1.1 | 내용 수정 | 개발내용 추가 |
| 2023-05-26 | 박우혁 | 1.2 | 내용 수정 |  |
| 2023-06-07 | 김민수 | 1.3 | 내용 추가 | 개인 평가 작성 |
| 2023-06-15 | 박우혁 | 1.4 | 내용 추가 |  |
|  |  |  |  |  |
|  |  |  |  |  |

**본 양식은 소프트웨어학부 C++프로그래밍 과목의 프로젝트 결과보고서 작성을 위한 기본 양식입니다. 문서의 필수 항목을 제시하는 것이니 폰트, 문단 구조 등의 디자인 부분은 자유롭게 설정하기 바랍니다. 양식 내에 붉은 색으로 기술한 부분은 지우고 작성하기 바랍니다.**

**목 차**

[1 개요 4](#_Toc43103653)

[2 개발 내용 및 결과물 5](#_Toc43103654)

[2.1 목표 5](#_Toc43103655)

[2.2 개발 내용 및 결과물 6](#_Toc43103656)

[2.2.1 개발 내용 6](#_Toc43103657)

[2.2.2 시스템 구조 및 설계도 6](#_Toc43103658)

[2.2.3 활용/개발된 기술 6](#_Toc43103659)

[2.2.4 현실적 제한 요소 및 그 해결 방안 6](#_Toc43103660)

[2.2.5 결과물 목록 7](#_Toc43103661)

[3 자기평가 8](#_Toc43103662)

[4 참고 문헌 8](#_Toc43103663)

[5 부록 8](#_Toc43103664)

[5.1 사용자 매뉴얼 8](#_Toc43103665)

[5.2 설치 방법 8](#_Toc43103666)

# 개요

|  |
| --- |
| **평가기준 (10점)**  **프로젝트를 완성하기 위해 사용한 개발 방법을 기술하세요.**  **또한 사용하고 있는 외부 라이브러리와 해당 라이브러리를 획득/설치하는 방법을 기술하세요.** |

**프로젝트의 전체적인 구조 및 개발 내용을 명확하게 기술한다.**

스네이크 게임을 만들기 위해 nucrses와 cpp STL 라이브러리를 이용해서 제작한다.

게임은 유저의 입력을 받아 방향을 정하고 일정 시간마다 스네이크를 움직이며 게임이 진행된다.

전체적인 감을 잡기 위해 demo게임을 개발해서 게임을 어떻게 구현해야 할지 구상했다.

그리고 이를 바탕으로 완성된 게임을 만들기 위해 **게임 루프 디자인 패턴**을 사용해서 프로그램의 구성을 입력, 게임진행, 렌더링 크게 세 부분으로 나누었다.

즉, 유저의 방향키 입력을 받는 InputManager

게임의 전반적인 로직을 구현하고 이를 진행하는 StageManager

그리고 진행된 맵 데이터를 바탕으로 이를 ncurses로 그리는 RenderManager

이 3개의 클래스로 구성했다. 그리고 이를 싱글톤 패턴을 적용하기 위해 전역변수로 설정하여 언제든 해당 값을 서로 불러와 사용할 수 있도록 했다. 이를 통해 함수의 매개인자가 복잡해지는 것을 최대한 줄이고자 하였다.

기본적으로 60fps를 카운트해서 프레임 정보와 시간 정보 갱신하여 이를 바탕으로 어떤 주기마다 게임을 진행하고 렌더링 할 것인지 설정했다.

최대한 간단하게 구현하기 위해 ncurses와 기본 STL 라이브러리 외에는 사용하지 않는다.

시간에 따른 게임 렌더링을 구현하기 위해 chrono 라이브러리를 사용했다.

아이템의 랜덤한 위치를 가져오기 위해 vector와 random 라이브러리를 사용했다.

# 개발 내용 및 결과물

## 목표

|  |
| --- |
| **작성요령 (10점)**  **프로젝트의 목표를 기술하세요. 각 단계별 목표를 구체적으로 쓰세요.** |

|  |  |  |
| --- | --- | --- |
| 적용단계 | 내용 | 적용 여부 |
| 1단계 | Map의 구현 | 적용 |
| 2단계 | Snake 표현 및 조작 | 적용 |
| 3단계 | Item 요소의 구현 | 적용 |
| 4단계 | Gate 요소의 구현 | 적용 |
| 5단계 | 점수 요소의 구현 | 적용 |

**프로젝트의 각 적용 단계별 구현 목표를 명확하게 제시한다. 제시한 권고안의 내용을 포함하여, 변경된 부분 등을 구체적으로 단계별 구현 목표를 작성해야 한다.**

<1단계:>

* 1. Snack map을 21x21의 2차원 배열로 구성한다. 값이 의미하는 바는 다음과 같이 정한다.

nothing: 0

snack\_head: -1

snack\_body: 1~INT\_MAX;

Immune Wall: -2

Wall: -3

Gate: -4

Growth: -15

Poison: -25

* 1. Ncurses를 이용하여 해당 2차원 배열 값을 읽어서 화면에 출력한다.
  2. Snack map의 크기를 다양화 하고, 여러 맵을 사용할 수 있도록 구조를 수정한다.

<2단계:>

2-1. 1단계의 맵 위에 Snake를 표시하고, 화살표를 입력 받아 Snake가 움직이도록

프로그램을 완성한다.

2-2. Snake는 규칙 #1을 준수해야 한다.

2-3. Tick에 대한 변화 주기(추가사항)

<3단계:>

3-1. 2단계 프로그램에서, Map위에 Growth Item와 Poison Item을 출현하도록 수정한다.

○ 게임 규칙#2를 준수해야 한다.

3-2. Growth Item과 Poison Item을 Map 배열에 표현할 때 값을 정한다.

○ 화면상에 표현시, 색이나 기호를 달리하여 구분할 수 있도록 한다.

○ Map Data에서 Growth Item은 5, Poison Item은 6과 같이 구분할 수 있도록 한다.

3-3. Snake 와 Item 의 상호작용 처리

+ Item의 종류에 대한 추가 (추가 사항): 새로운 Item에 대해 제안하고, 구현하기.

<4단계:>

4-1. 3단계 프로그램에서, Map의 Wall의 임의의 위치에 한 쌍의 Gate가 출현할 수 있도록 변경하고, 각 Gate에 Snake가 통과할 수 있도록 수정한다.

● 게임 규칙 #3, #4, #5를 준수해야한다.

● Wall(Immune Wall 포함)과 Gate를 Map 배열에 표현할 때 값을 결정한다.

○ 화면상에 표현시, Gate는 Wall과 구분될 수 있도록한다.

○ Map Data에서 Gate는 7과 같이 하여, 다른 요소와 구분할 수 있도록 한다.

4-2. (1) Snake  Wall 의 상호작용 처리 (2) SnakeGate 의 상호작용 처리

(3) Wall에 대한 변화 추가 (추가 사항): 새로운 Wall 의 동작에 대해 제안하고, 구현하기

<5단계:>

5-1. 4단계 프로그램에서, 우측에 게임 점수를 표시하는 화면을 구성한다.

○ 슬라이드 Snake Game의 구조 참고

● 게임 점수는 게임 규칙 #6을 준수한다.

5-2. Mission

○ 1. 구성한 Map의 정의에 고정 값을 주거나,

○ 2. 매 게임마다 임의의 값을 주는 방식으로 처리한다.

● Mission을 달성하면 다음 Map으로 진행하도록 프로그램을 완성한다.

○ Stage는 최소 4개로 구성하고, 각 Stage의 Map은 서로 달라야 한다.

5-3. Score Board의 대한 설계사항

○ (1) Game 상태를 Score Board로 표현

○ (2) Score Board에 대한 변경 (추가 사항)

■ Score Board에 대한 상호작용방법, 혹은

■ Score Board에 표현되는 요소의 추가 등을 제안하고, 구현하기.

## 개발 내용 및 결과물

### 개발 내용

|  |
| --- |
| **작성요령 (10점)**  **프로젝트의 수행의 내용을 구체적으로 기술한다. 세부 목표별로 어떤 결과를 어떤 방법으로 달성하였는지를 자세히 기술한다.** |

게임의 전체적인 동작은 Game Loop Design 패턴을 이용하고자 했다. 이를 위해 60fps를 기준으로 두고 정해 둔 프레임(틱) 마다 게임을 진행시키고 프린트 하고자 했다. 그리고 Game Loop를 작동시킬 전체적인 구조를 game을 진행하는 stageManager, 입력을 받는 inputManager, 출력을 담당하는 renderManger라는 3개의 클래스로 나누어 구현의 복잡도를 줄이고자 했다.

우선 게임의 로직을 구현하는 데 중점을 두기 위해 1단계 Map에 대한 구현은 우선 21x21의 2차원 배열로 간단히 했다. Map에 저장하는 값을 초기 구상으로는 class로 나누어 넣을까 했지만 snake의 움직임을 구현하는 좋은 아이디어 생각나 이를 활용하기 좋게 정수 값을 Map에 그려진 요소로 정하기로 했다.

2단계 Snake 표현 부분은 snake의 이동경로를 배열로 저장하거나, LinkedList로 tail 좌표를 삭제하는 등으로 snake의 body정보를 저장하여 사용하는 방식을 처음 떠올렸으나 더 좋은 아이디어가 떠올라서 그 방식으로 구현하기로 했다.

바로 snake의 body를 snake head가 남기는 일종의 흔적으로 생각하는 것이다. 흔적은 snake의 현재 길이만큼의 생명을 가지면서 생겨나고, snake가 이동하는 틱마다 흔적의 생명이 1씩 줄어들도록 하면 결과적으로 snake의 body가 이동하는 것처럼 보이게 될 것이다. 이렇게 하면 복잡하게 snake의 body위치를 고민할 필요 없이 틱마다 모든 0보다 큰 값을 1씩 줄이도록 하면 구현이 가능하기 때문에 다른 클래스를 둘 필요 없이 map 배열만을 가지고 게임의 구현이 가능해 진다. 이를 위해 map의 다른 요소들은 음수 영역에 할당하여 snake의 body 한 칸의 생명이 직관적으로 map의 양수 값이 될 수 있도록 하였다. 그리고 이를 바탕으로 ncurses를 이용해 map배열의 값을 바탕으로 출력을 하였다.

2단계 Snake 조작 부분은 Game Loop의 while문 안에서 타임아웃을 설정한 ncurses의 getch()를 이용하여 계속 최근 방향키 input을 받도록 했고 가장 최근에 갱신된 input값을 바탕으로 틱이 되어 game이 갱신될 때 사용할 수 있도록 했다. 이를 통해 input을 기다리기 위해 게임 전체가 멈추는 것을 방지 할 수 있었다. 그리고 input값을 바탕으로 head의 위치를 이동시키고 남은 자리에 snake.length의 생명을 가진 body를 생성하고, 모든 양의 정수를 1씩 줄여 body를 이동하는 방식으로 구현했다.

3단계 Growth Item와 Poison Item의 기능은 head가 item을 만났을때 모든 body의 생명을 1씩 늘리거나 줄이는 방식으로 간단히 구현 할 수 있었다. Item을 일정 주기마다 리스폰하는 기능을 처음에는 snake body 처럼 생명 값을 주고 틱마다 줄이려 했으나 게임의 틱과 상관 없이 실제 시간 (ex: 5초)에 따라 동작하면 되기에 게임 진행시간을 추적하는 변수를 두어 그 주기마다 갱신 하는 방식으로 해결했다.

4단계 Gate의 구현은 앞서 snake body를 head와 분리해서 생각했기 때문에 head만 잘 들어가고 나오도록 하면 가능한 쉬운 문제였다. Gate에 head가 진입하면 다음 좌표를 다른 Gate로 정한 뒤 가능한 출구를 제시된 규칙에 따라 순서대로 구현했다. Gate의 리스폰은 Item과 동일한 구조로 실제 시간을 기준으로 하되 snake가 통과 중에 Gate가 변경되는 것을 막기 위해 snake의 길이를 기준으로 한 Flag를 snake body처럼 틱마다 감소 시키는 것으로 통과 중 상태라는 것을 알리도록 했다.

5단계 Mission과 ScoreBoard의 구현은 이미 만들어 둔 로직에서 카운트를 하는 변수를 넣어서 해당 행위가 일어날 때 변수를 증가시켜 값을 추적하고 이를 읽어서 출력했다. 그리고 Mission을 달성하면 stage\_complete\_flag를 true로 설정하여 게임이 다음 stage로 가도록 했다.

앞서 간단히 구현했던 이차원 배열 map을 다양한 크기로 동작할 수 있도록 구조를 수정하고, stage에 관한 mission정보 등을 담을 수 있도록 map class를 만들었다. Map 제작을 쉽게 하기 위해서 {} 할당을 사용하였다. 콘솔 크기를 조정하는 것이 어렵고 이를 초과하면 화면이 출력되지 않는 한계가 있었기 때문에 기본 콘솔 크기를 초과하지 않는 최대 배열 크기를 정해 놓고 {}할당으로 최대 크기 이내로 원하는 값을 적고 실제 map 크기를 width와 height 값으로 전달하여 맵을 생성하도록 했다.

이 map들을 여러 개 담고 활용할 수 있도록 mapManager도 추가하였다.

### 시스템 구조 및 설계도

|  |
| --- |
| **작성요령 (30점)**  **프로젝트의 각 세부 목표의 주요 기능(알고리즘 등)에 대해서 기술한다. 세부 목표별로 수정한 프로그램 소스 파일을 나열하고, 해당 파일에서 세부 목표를 달성하기 위해 작성한 클래스/함수에 대해 나열하고, 각 요소에 대해 간략한 설명을 작성한다. 또한 각 요소의 개발자를 명시한다.** |

### 활용/개발된 기술

|  |
| --- |
| **작성요령 (10점)**  **프로젝트 수행에 사용한 외부 기술/라이브러리를 나열하여 작성한다. 각각 기술을 이 프로젝트에 적용할 때, 도움 받거나 해결하고자 하는 기능에 대해 상세히 설명한다.**  **NCURSES / STL 라이브러리 등을 포함하여 설명한다.**  **또한, 이 프로젝트를 수행하면서, 새롭게 고안한 알고리즘 등이 있다면 설명한다.** |

### 현실적 제한 요소 및 그 해결 방안

|  |
| --- |
| **작성요령 (5점)**  **제안된 프로젝트의 단계 별 수행에 있어, 제한 요소를 찾아 작성한다. 해당 제한 요소를 해결하기 위해서 어떤 방법으로 해결하였는지 작성한다.** |

### 결과물 목록

|  |
| --- |
| **작성요령 (5점)**  **결과물 목록을 작성한다. 목록은 제출하는 파일과 각 파일의 역할을 간략히 설명한다.** |

# 자기평가

|  |
| --- |
| **작성요령 (5점)**  **프로젝트를 수행한 자기 평가를 서술한다. 팀원 개개인의 자기 평가가 포함되어야 하며, 본인의 역할, 프로젝트 수행 시 어려운 점, 도움이 되었던 점, 이 프로젝트 운영에 개선이 필요하다고 생각하는 점을 충분히 서술한다.** |

# 참고 문헌

**참고한 서적, 기사, 기술 문서, 웹페이지를 나열한다.:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 번호 | 종류 | 제목 | 출처 | 발행년도 | 저자 | 기타 |
|  | 서적 |  |  |  |  |  |
|  | 기사 |  |  |  |  |  |

# 부록

|  |
| --- |
| **작성요령 (15점)**  **프로젝트의 결과물을 사용하기 위한 방법에 대해서 작성하세요.** |

## 사용자 매뉴얼

**프로젝트 실행 후 사용안내, 따라하기 등 포함**

## 설치 방법

**프로그램 컴파일 및 실행 방법 포함**

**game 폴더 안에서 make 실행, root에 만들어진 snake-game 실행.**