

오픈소스 SW개론

과제 5: Tetris Game

조장 17011679 홍혜인

17011653 김지수

17011657 이승민

목차

I. 프로젝트 개요	1p
가) 프로젝트 주제	1p
나) 주제 선정 이유	1p
다) 프로젝트 목표	1p
라) 프로젝트 환경	2p
II. 선택한 오픈소스 SW에 대한 간단한 설명	3p
가) Console 테트리스 게임	3p
나) 다운로드 경로 및 출처	3p
다) 라이선스	4p
III. 프로젝트 수행 내용	5p
가) Github 활용	5p
나) Git 활용	12p
다) 오픈소스 수정	15p
IV. 프로젝트 진행 과정 및 시행 착오	20p
가) 프로젝트 진행 과정	20p
나) 시행 착오	24p
V. 역할 분담	26p
VI. 참고문헌	27p

I. 프로젝트 개요

가) 프로젝트 주제

우리 팀은 테트리스 게임을 프로젝트 주제로 정하였다. 테트리스 게임은 퍼즐 게임으로, 네 개의 사각형으로 이루어진 블록이 랜덤한 모양으로 나타나 바닥과 블록 위에 떨어진다. 이 게임의 목표는 블록을 움직이고 90도 회전하여, 수평선을 빈틈없이 채우는 것이다. 수평선이 만들어지면 이 선은 없어지며 블록이 보드의 꼭대기까지 가득 메워 블록이 더 들어갈 공간이 없게 되면 게임이 끝나게 된다.

나) 주제 선정 이유

테트리스 게임은 남녀노소 모두 쉽게 즐길 수 있는 고전 게임이다. 컴퓨터 게임 뿐만 아니라 휴대폰 게임으로 등장할 만큼 테트리스 게임은 많은 사람들에게 사랑을 받고 있다. 대중적인 게임인 테트리스 게임을 프로젝트 주제로 정하는 것이 사람들에게 흥미를 유발하고 팀원의 이해도를 높일 수 있을 것 같아 테트리스 게임을 프로젝트 주제로 정하였다. 또한 해당 오픈소스 코드는 가독성이 떨어지는 부분이 많아서, 오픈소스SW개론 수업 때 배운 내용을 바탕으로 오픈소스를 수정해보고 싶다는 생각이 들어 프로젝트 주제로 정하였다.

다) 프로젝트 목표

Github 활용	팀원과 소통
	분류 및 관리
	문서화 작업
	배포 관리
	그래프
Git 활용	작업 공간
	commit 메시지
	버전 관리
	Log
오픈소스 코드 개선	가독성
	유지 보수
	오류 해결
	기능 추가

[표 1] 프로젝트 목표

1. Github 활용

Issue 와 Pull request 를 사용하여 팀원과 소통한다. 또한 Issue 와 Pull request 를 분류하기 위해 Label, Milestones, Project 를 사용하여 Issue 와 Pull request 를 분류 및 관리한다. 그리고 기존 오픈소스에 없었던 README 파일을 만들어 사용자에게 높은 접근성을 제공하고 LICENSE 파일을 만들어 원작자와 사용자 간의 조건을 명시한다. 또한 Wiki 을 사용하여 프로젝트를 진행하는데 도움이 되는 자료를 제공하고, Release 을 사용하여 사용자가 원하는 버전 별로 소스 파일을 다운로드 받을 수 있도록 한다. 마지막으로 Network 을

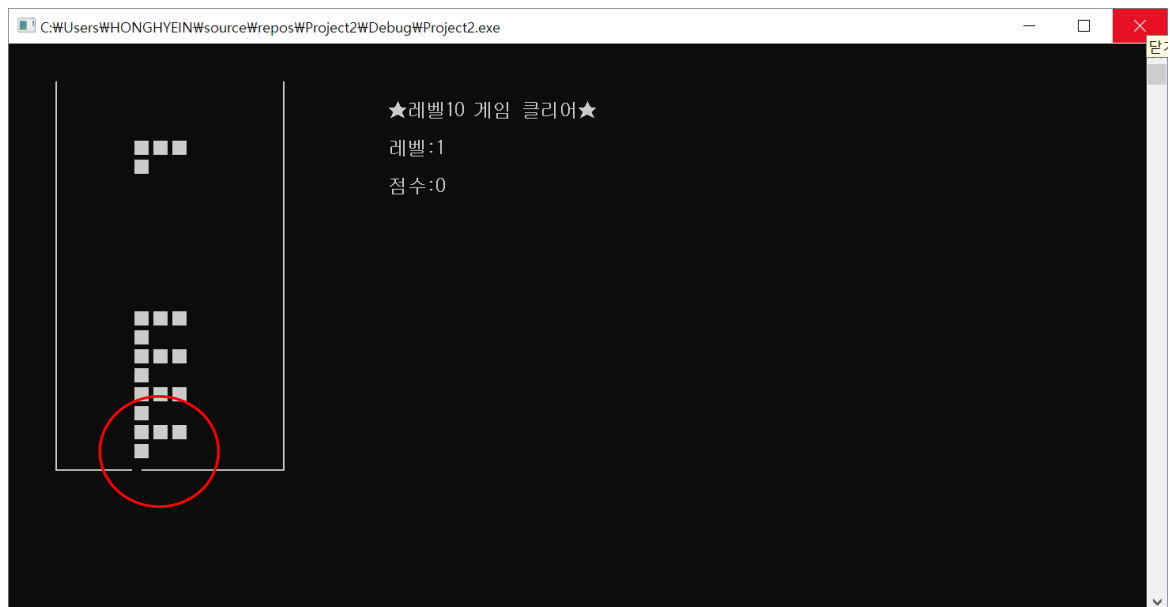
통해 branch 가 어떻게 merge 되고 있고 현재 어떤 상태인지에 대한 정보를 제공받고, Pulse 와 Contributor 을 통해 팀원의 활동율을 객관적으로 비교한 자료를 제공받는다.

2. Git 활용

작업 공간인 branch 를 만들어 안전한 작업 환경에서 오픈소스를 수정할 수 있도록 한다. 또한 commit 메시지에 해당 작업에 대한 설명을 작성하여 팀원의 이해도를 높인다. 마지막으로 tag 을 사용하여 버전을 관리한다.

3. 오픈소스 코드 개선

해당 오픈소스 코드는 사용자가 이해하기에 다소 어렵다. 우리 팀은 해당 오픈소스의 가독성을 높이는 목적으로 프로젝트를 진행할 예정이다. 변수 및 함수 이름을 의미 있게 변경하고 일관된 형식을 유지하여 코드를 읽기 편하도록 수정하고, 머리 주석 및 함수 주석을 작성하여 코드를 읽는 사람의 이해도를 높인다. 그리고 여러 개의 기능을 포함하는 작업을 분할하여 최소화하고 분할 컴파일을 통해 유지 보수를 하기 편하도록 수정한다. 또한 [그림 1]에서 볼 수 있다시피 해당 오픈소스는 랜덤한 모양의 블록을 생성하지 못하고 같은 모양의 블록만 생성하는 문제점과, 블록 바로 밑에 보드가 있을 경우 보드가 출력되지 않는 문제점을 가지고 있다. 이러한 오류가 발생한다는 점을 유념하고 오류를 처리한다. 마지막으로 기존 오픈소스에 추가적인 기능을 추가하여 사용자가 테트리스 게임을 하는데 도움을 줄 수 있도록 한다.



[그림 1] 해당 오픈소스가 가지고 있는 오류

라) 프로젝트 환경

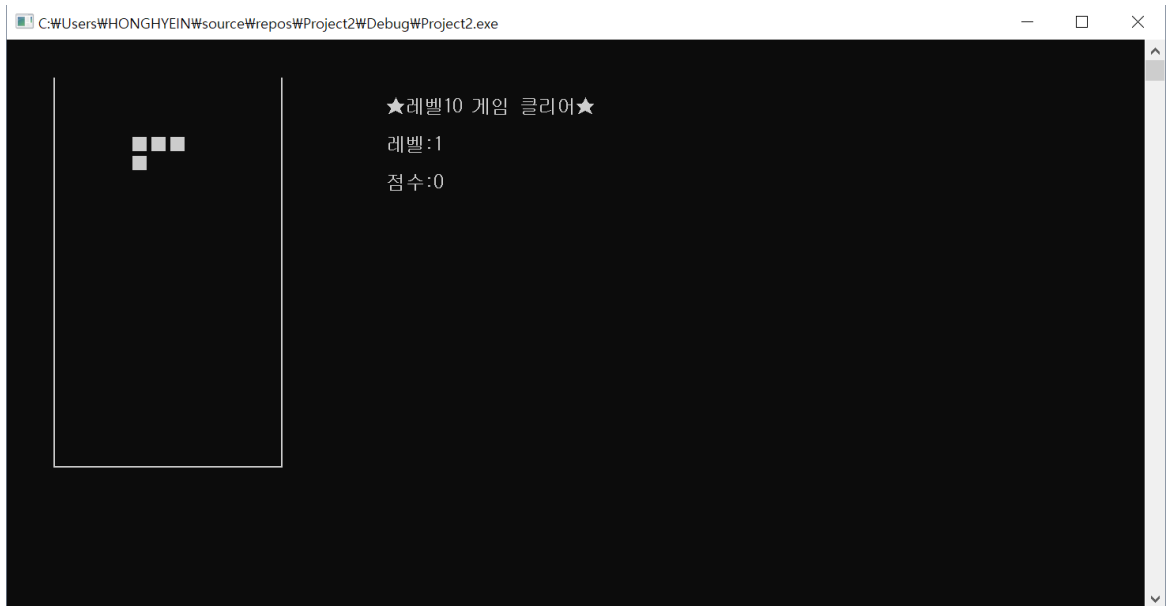
C 언어를 사용하며, Windows의 Visual Studio 2017버전을 실행 환경으로 사용한다. Microsoft Visual Studio Community 2017 버전의 소프트웨어 사용권 계약서의 링크 및 출처는 다음과 같다.

<https://visualstudio.microsoft.com/ko/license-terms/mlt553321/>

II. 선택한 오픈소스SW에 대한 간단한 설명

가) Console 테트리스 게임

해당 오픈소스는 C 언어를 사용하여 테트리스 게임을 콘솔 창에 구현하였다. 일반적인 테트리스 게임과 마찬가지로, 네 개의 사각형으로 이루어진 블록이 랜덤한 모양으로 나타나 바닥과 블록 위에 떨어진다. 플레이어는 블록을 움직이고 90도 회전하여, 수평선을 빈틈없이 채우는 것이 목표이다. 만약 수평선이 만들어지면 이 선은 없어진다. 블록이 보드의 꼭대기까지 가득 메워 블록이 더 들어갈 공간이 없게 되면 게임이 끝나게 된다. 해당 오픈소스는 게임 보드와 테트리스 게임에 사용되는 블록 등 화면에 표시되는 요소를 배열을 사용하여 그려지도록 구현되어 있다. 조작 방법은 방향키 ←, ↓, →를 사용하여 블록을 방향키에 맞는 방향으로 이동시키고, 방향키 ↑를 사용하여 블록을 회전시키는 방법이다. 또한 스페이스바를 눌러 블록을 바닥으로 즉시 이동시킬 수 있다. 하지만 일반적인 테트리스 게임과 달리 해당 오픈소스는 레벨에 따라 블록이 떨어지는 속도를 조절하여 게임의 재미를 배가하였다. 하지만 랜덤한 모양의 블록을 생성하지 못하고 같은 모양의 블록만 생성하는 문제점과, 블록 바로 밑에 보드가 있을 경우 보드가 출력되지 않는 문제점을 가지고 있다.

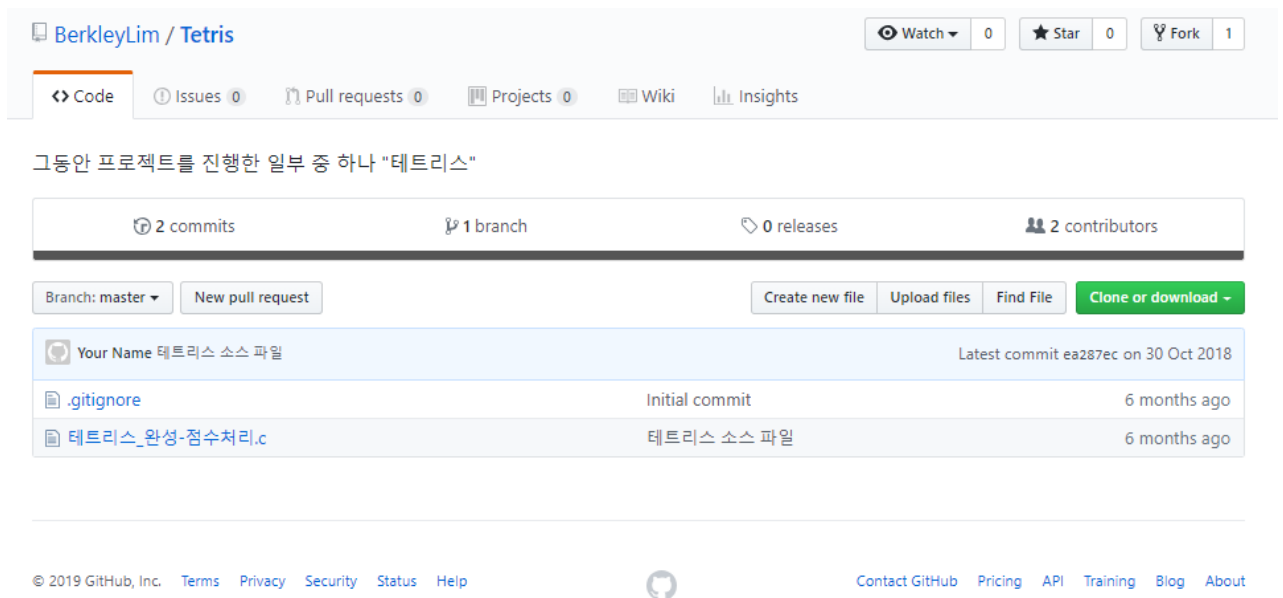


[그림 2] 해당 오픈소스를 실행한 화면

나) 다운로드 경로 및 출처

해당 오픈소스는 Github에서 무료로 다운로드 가능하며 링크 및 출처는 다음과 같다.

<https://github.com/BerkleyLim/Tetris>



[그림 3] 오픈소스 repository

다) 라이선스

해당 오픈소스는 라이선스를 포함하고 있지 않다. 오픈소스를 사용하기 위해 원작자에게 이메일을 보냈지만, 현재까지 답장이 없는 상태이다. 따라서 프로젝트를 통해 팀원들과 라이선스를 만들고 사용 조건을 명시하였다. 라이선스와 관련하여 3. 프로젝트 수행 내용에서 자세하게 언급할 예정이다.

III. 프로젝트 수행 내용

가) Github 활용

- Description와 Topics

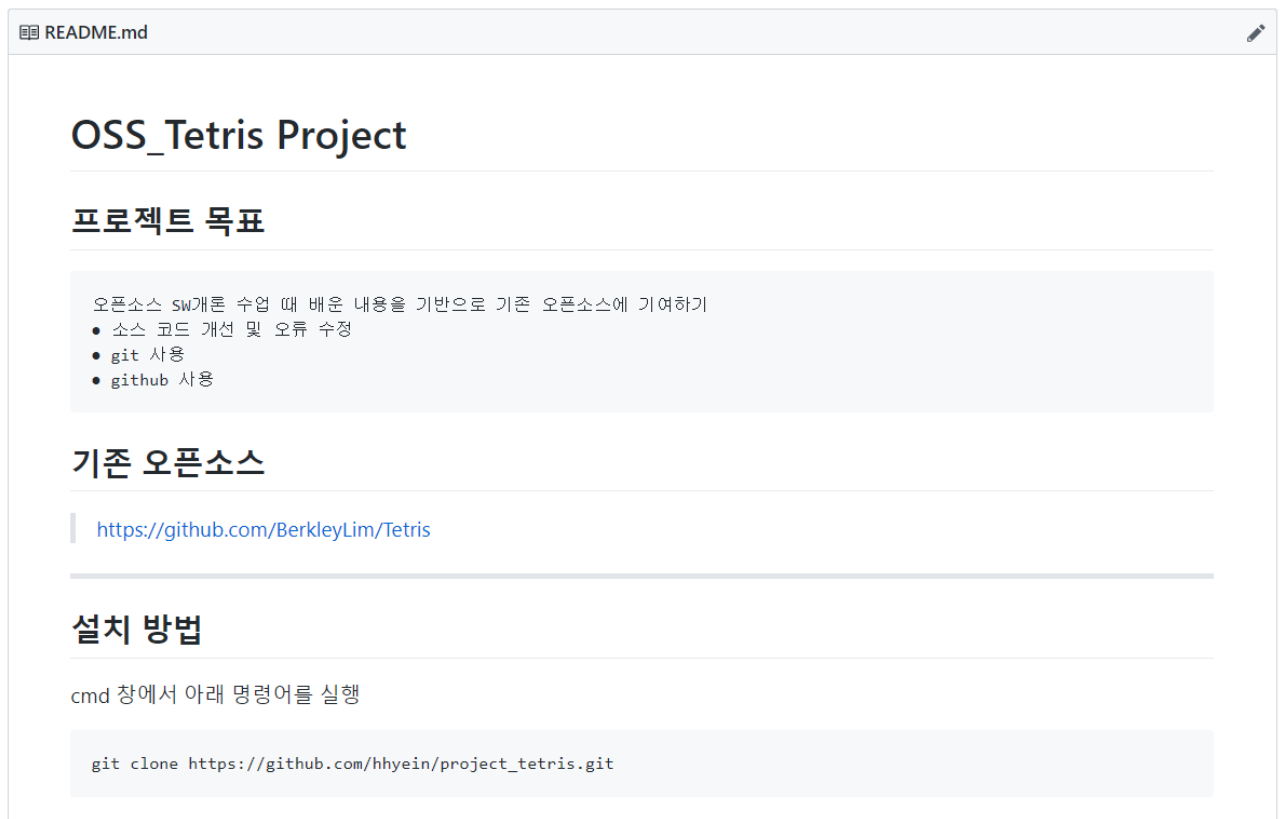
Repository에 대해 간단하게 설명하고 분류를 하기 위해 Description을 작성하고 Topics 기능을 사용하였다. [그림 4]에서 볼 수 있다시피 해당 Repository에 대한 설명을 Opensource Project라고 간단하게 작성하였고, c-language와 tetris-game topic을 추가하여 C 언어와 테트리스 게임에 관심있는 사용자가 해당 Repository에 쉽게 접근할 수 있도록 하였다.



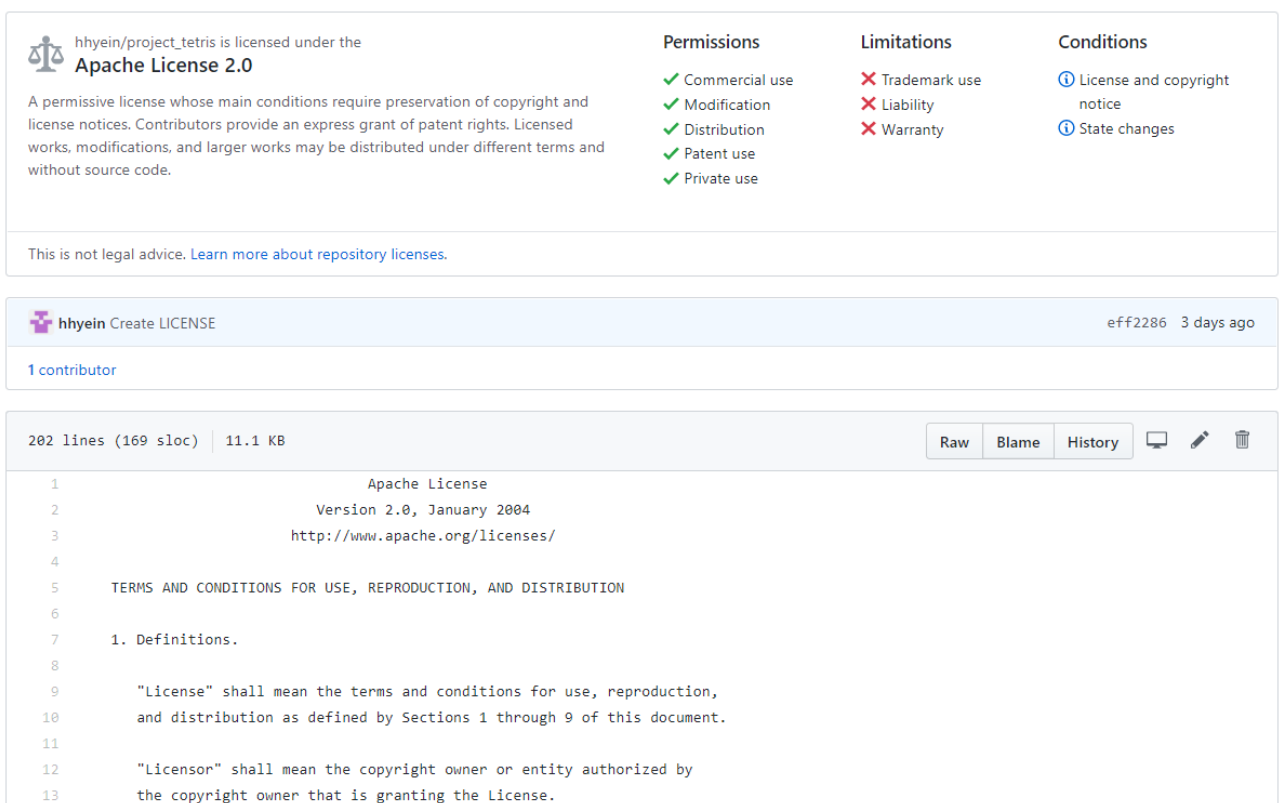
[그림 4] description와 manage topics

- README.md 파일과 LICENSE 파일

기존 오픈소스는 README 파일과 LICENSE 파일을 포함하고 있지 않았다. 따라서 README 파일을 생성하고 해당 프로젝트에 대한 간단한 소개 및 설명을 작성하여 Repository의 첫 화면에 보일 수 있도록 한다. 이를 통해 사용자가 쉽게 프로젝트에 접근하고 이해할 수 있도록 돕는다. README 파일에는 프로젝트 목표, 설치 및 사용 방법, 원작자의 repository 주소, 팀원의 github 주소와 메일에 대한 내용이 있다. 우리 팀은 해당 프로젝트를 진행하면서 원작자와 소통이 되지 않아서 매우 큰 불편함을 느꼈다. 따라서 다른 사용자는 불편함을 느끼지 않도록 팀원의 github 주소와 메일을 추가로 작성하여 사용자가 원활하게 프로젝트에 기여할 수 있도록 도움을 주었다. 또한 연락처를 작성함으로써 해당 프로젝트에 대한 신뢰도를 높였다. 또한 LICENSE 파일을 생성하여 원작자와 사용자 간의 조건을 명시하였다. 많은 LICENSE 템플릿 중에서 우리 팀은 Apache-2.0을 선택하였는데, 만약 Copyleft에 포함된 라이선스를 선택한다면 많은 사용자들이 자신이 수정한 소스 코드를 반환해야 하는 의무 때문에 프로젝트에 크게 기여하지 않을 것 같아 Apache 라이선스를 선택하였다.



[그림 5] README.md 파일

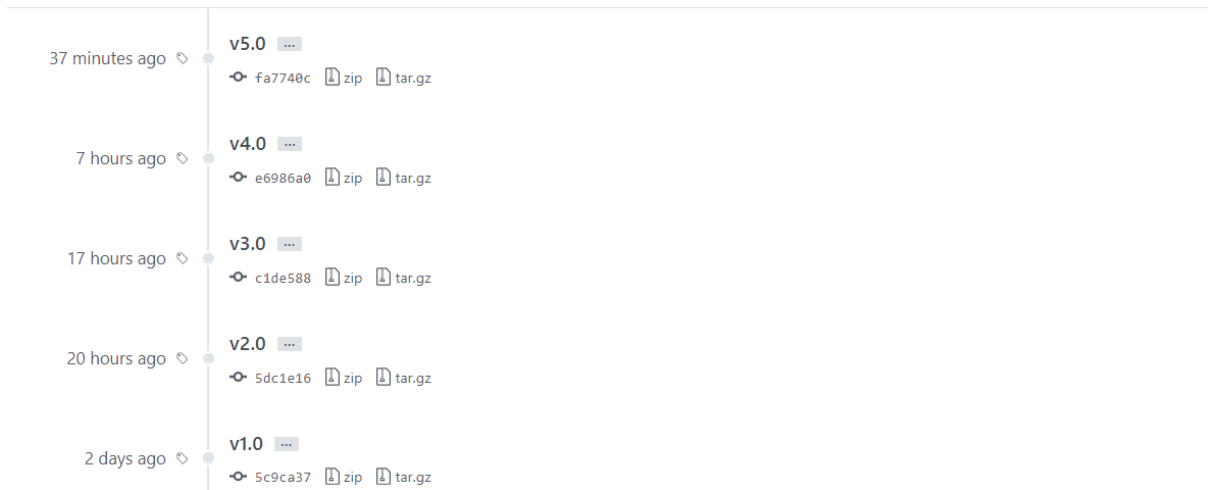


[그림 6] LICENSE 파일

● Release

Release을 사용하여 사용자가 원하는 버전 별로 소스 파일을 다운로드 받을 수 있도록 한다. 해당

프로젝트에서는 5개의 버전이 존재한다. 버전에 대한 설명은 tag에서 자세하게 다를 예정이다.

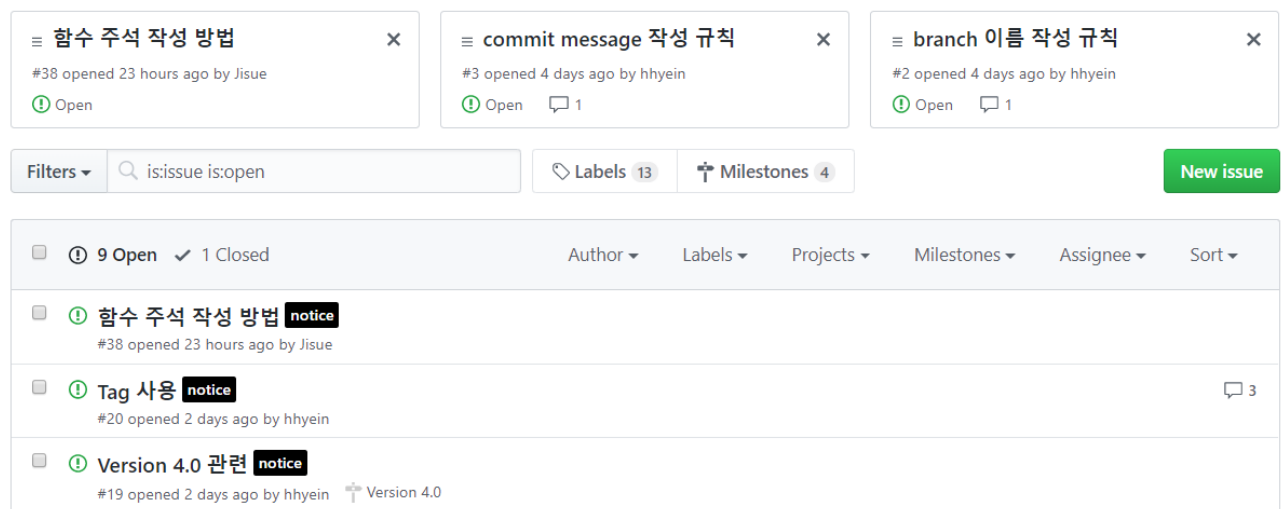


[그림 7] Release

● Issues와 Pull requests

팀원과 소통해야 할 때 Issues와 Pull requests 기능을 사용하였다. 팀원에게 공지하거나 토론해야 할 주제가 생기면 Issues를 생성하였다. 만약 더 이상 필요하지 않은 Issue라면 Open 상태에서 Closed 상태로 변경하여 현재 프로젝트 진행에 필요한 Issues에 대해서만 볼 수 있게 하였다. 특히 [그림 5]에서 볼 수 있다시피 여러 개의 Issues 중에서 중요한 Issues를 한 눈에 확인할 수 있게 해주는 Pinned issues 기능을 매우 편리하게 사용하였다. 또한 소스 코드를 수정한 후 Merge하기 전에 팀원에게 허락을 받거나 피드백을 받아야 하는 경우 Pull request를 작성하였다. Pull request에는 소스 코드를 수정한 이유와 수정 사항에 대한 내용을 작성하였다. 또한 Pull request를 통해 Commit 메시지를 읽어볼 수 있고, Files changed를 통해 기존 소스 코드와 수정한 소스 코드를 비교할 수 있어서 어떤 점이 수정되었는지 쉽게 확인하였다. 해당 프로젝트를 진행할 때는 branch마다 Pull requests를 작성해서 Merge 할 때 충돌이 생길 위험성을 크게 줄였다. 또한 Merge한 branch에 대한 Pull request는 자동적으로 Open 상태에서 Closed 상태로 변경되기 때문에 별도의 관리가 필요 없어서 편리하게 느껴졌다.

Pinned issues



[그림 8] Issues


0 Open	50 Closed	Author	Labels	Projects	Milestones	Reviews	Assignee	Sort
<input type="checkbox"/>	<input checked="" type="checkbox"/>							
<input type="checkbox"/>	<input checked="" type="checkbox"/>	SPT scorelevel 기능 함수 분할	split					
		#61 by hhyein was merged 7 hours ago	Version 4.0					
<input type="checkbox"/>	<input checked="" type="checkbox"/>	SPT - cursor 기능 함수 분할	split					
		#60 by hhyein was merged 7 hours ago	Version 4.0					
<input type="checkbox"/>	<input checked="" type="checkbox"/>	SPT Run 함수 최소화	split					1
		#59 by hhyein was merged 14 hours ago	Version 4.0					
<input type="checkbox"/>	<input checked="" type="checkbox"/>	SPT block.h 생성	split					1
		#58 by hhyein was merged 14 hours ago	Version 4.0					
<input type="checkbox"/>	<input checked="" type="checkbox"/>	EOR 블록 모양 오류 해결	bug					1
		#56 by Jisue was merged 17 hours ago	Version 3.0					
<input type="checkbox"/>	<input checked="" type="checkbox"/>	CLN Detect 함수 주석 작성	comment					1
		#55 by shuffle2479 was merged 20 hours ago	Version 2.0					
<input type="checkbox"/>	<input checked="" type="checkbox"/>	CLN Initial 함수 주석 작성	comment					1
		#54 by shuffle2479 was merged 20 hours ago	Version 2.0					
<input type="checkbox"/>	<input checked="" type="checkbox"/>	CLN_Show_board 함수 주석 작성	comment					1
		#53 by shuffle2479 was merged 20 hours ago	Version 2.0					
<input type="checkbox"/>	<input checked="" type="checkbox"/>	CLN_Get_cursor 함수 주석 작성	comment					2
		#52 by shuffle2479 was merged 20 hours ago	Version 2.0					

[그림 9] Pull request

EOR 블록 모양 오류 해결 #56

Merged Jisue merged 1 commit into EOR from EOR_randomblock 10 seconds ago

Conversation 1
Commits 1
Checks 0
Files changed 1


Jisue commented 2 minutes ago • edited by hhyein

- 변경한 이유
블록모양이 random하게 나오지 않고 한가지만 나옴.
- 변경 사항
기존 블록 모양을 결정하는 코드를 `block = (rand() % 7) * 4` 으로 수정함.

[그림 10] Pull request 예시






● Label, Milestones, Projects

Issues와 Pull requests를 기능과 주제에 따라 분류하기 위해 Label와 Milestones을 사용하였다. Label을 통해 해당 Issues와 Pull requests가 어떤 기능과 주제와 관련되어 있는지 바로 직감할 수 있게 도움을 주었고, 원하는 기능과 주제에 관련된 Issues와 Pull requests를 한 번에 모아서 볼 수 있도록 하였다. 해당 프로젝트에서 사용한 Label은 [그림 11]에서 확인할 수 있다. 또한 Milestones을 사용하여 Issues와 Pull requests를 기준에 따라 분류하고 각각의 진행 상황에 대해 확인하였다. 해당 프로젝트에서는 Milestone을 버전에 따라 5개를 생성하였고, 해당 버전에 포함되는 Issues와 Pull requests를 Milestone으로 분류하고 진행 상황에 대해 확인하였다. 버전에 대한 설명은 tag에서 자세하게 다룰 예정이다. 또한 Issue와 Pull request를 더 세부적으로 관리하고 진행 상황에 대해

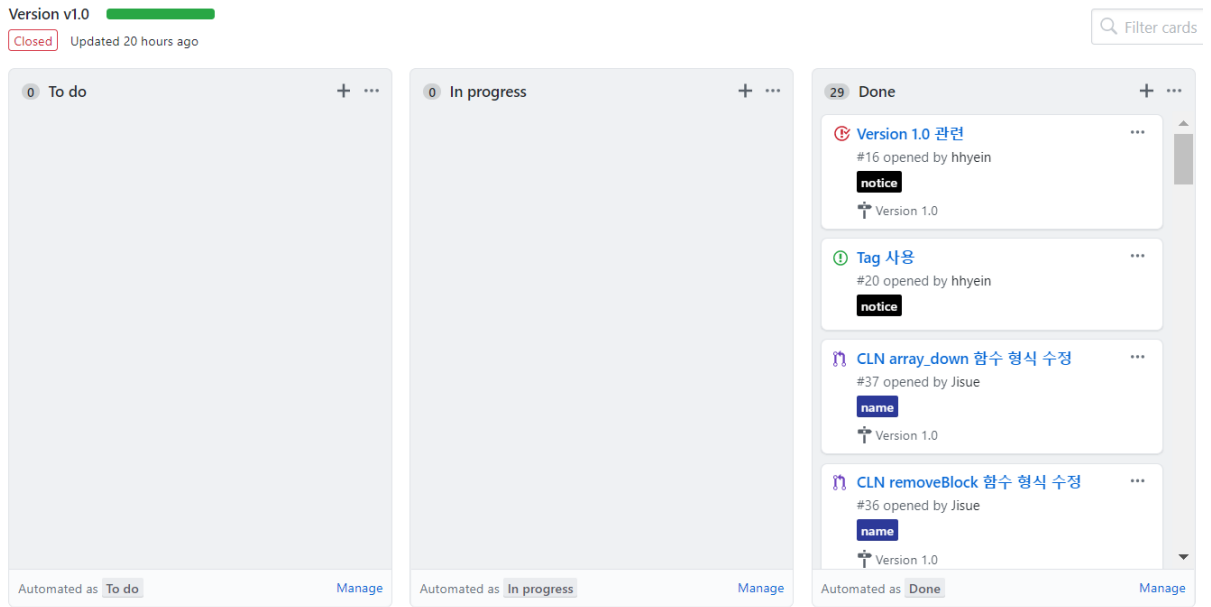
확인하기 위해 Projects을 사용하였다. 특히 Projects은 Milestones와 달리 해야 할 일, 진행중인 일, 끝낸 일로 분류하고 간단한 드래그 만으로 관리할 수 있다는 점이 편리하게 느껴졌다.

8 labels		Sort ▾
bug	Something isn't working	Edit Delete
comment	write or remove comments	Edit Delete
form	change to consistent form	Edit Delete
name	change name clear	Edit Delete
notice	5 open issues and pull requests	Edit Delete
question	Further information is requested	Edit Delete
remove	remove unnecessary code	Edit Delete
split	split multiple function and compilation	Edit Delete

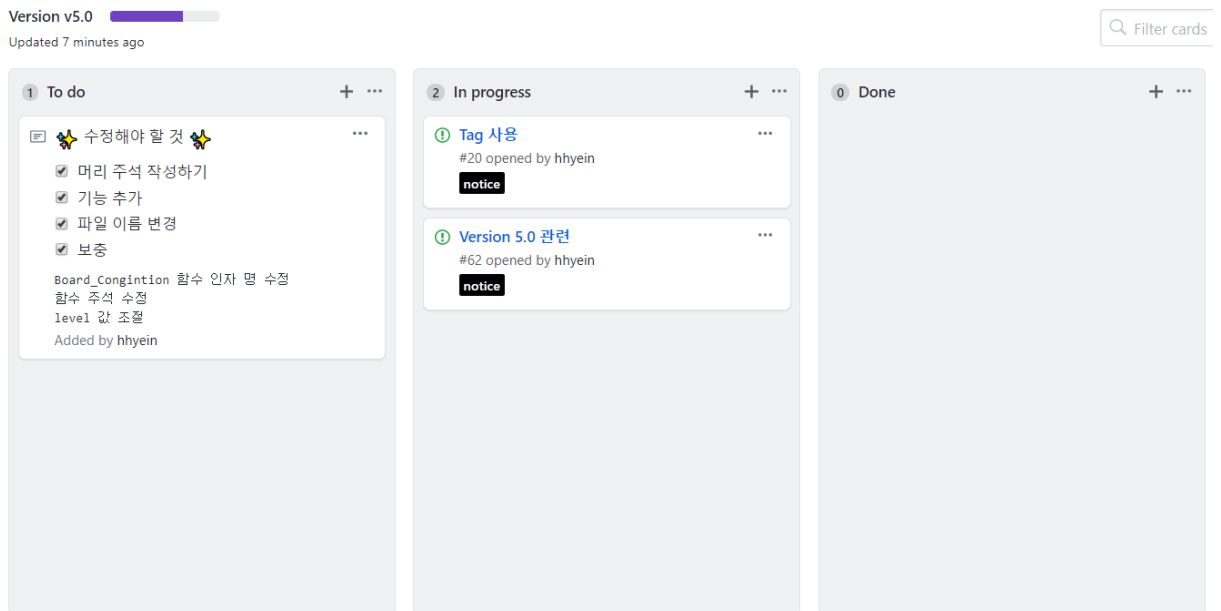
[그림 11] Label

✚ 0 Open ✓ 5 Closed		Sort ▾
Version 5.0 Closed 41 seconds ago ⌚ Last updated 1 minute ago fix and add	 100% complete 0 open 1 closed Edit Reopen Delete	
Version 4.0 Closed a minute ago ⌚ Last updated 1 minute ago Split compilation	 100% complete 0 open 5 closed Edit Reopen Delete	
Version 2.0 Closed 16 hours ago ⌚ Last updated about 16 hours ago write comments	 100% complete 0 open 18 closed Edit Reopen Delete	
Version 3.0 Closed 16 hours ago ⌚ Last updated about 16 hours ago fix errors	 100% complete 0 open 3 closed Edit Reopen Delete	
Version 1.0 Closed 17 hours ago ⌚ Last updated about 17 hours ago process of cleaning code	 100% complete 0 open 29 closed Edit Reopen Delete	

[그림 12] Milestones



[그림 13] Project 예시 1



[그림 14] Project 예시 2

● Wiki

README 파일에 작성하지 못한 내용은 Wiki 홈에 작성하여 해당 프로젝트에 대한 설명을 추가적으로 작성하였다. Wiki 홈에는 프로젝트 주제와 주제 선정 이유에 대한 자세한 설명을 작성하였다. 그리고 프로젝트 목표와 해당 프로젝트에 기여할 수 있는 방법을 구체적으로 작성하여 사용자가 프로젝트에 쉽게 참여할 수 있도록 도움을 주었다. 또한 해당 프로젝트를 진행하면서 도움이 될 만한 정보들은 Wiki를 사용하여 정보를 제공하였다. 해당 프로젝트에서는 tag 사용 방법, 마크 다운 사용 방법, 분할 컴파일에 대한 Wiki를 작성하였다. tag 사용 방법에 대한 Wiki에는 tag를 사용해야 하는 이유와 tag 명령어에 대한 설명이 있다. 또한 README 파일을 작성하거나 Issues와 Pull requests를 작성하려면 마크 다운 문법을 알아야 하므로 마크 다운 사용 방법에 대한 Wiki를 작성하였다. 해당 Wiki에는 기본적인 마크 다운 문법에 대한 설명이 있다. 마지막으로 분할 컴파일에 대한 Wiki에는 분할

컴파일에 대한 정보를 제공하는 주소를 첨부하여 팀원에게 도움을 주었다.

Home

hhyein edited this page 2 days ago · 2 revisions

[Edit](#)[New Page](#)

project_tetris

프로젝트 주제

테트리스 게임을 프로젝트 주제로 정하였다.

테트리스 게임은 퍼즐 게임으로, 네 개의 사각형으로 이루어진 블록이 랜덤한 모양으로 나타나 바닥과 블록이 게임의 목표는 블록을 움직이고 90도 회전하여, 수평선을 빈틈없이 채우는 것이다.

수평선이 만들어지면 이 선은 없어지며 블록이 보드의 꼭대기까지 가득 채워 블록이 더 들어갈 공간이 없게

▼ Pages 4

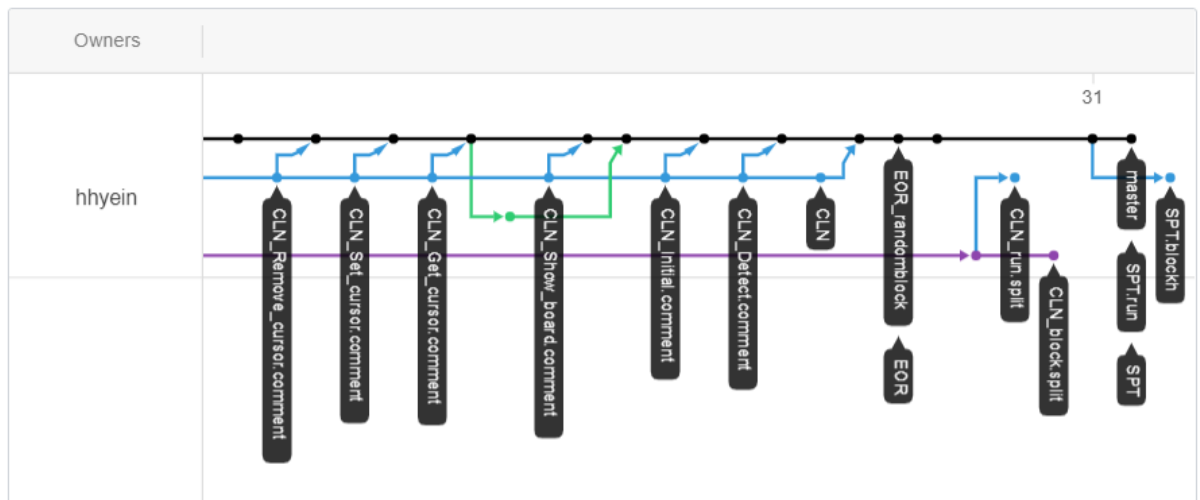
[Home](#)[tag 사용 방법](#)[마크다운 사용 방법](#)[분할 컴파일](#)[+ Add a custom sidebar](#)

Clone this wiki locally

[그림 15] Wiki

- Network

Network을 통해 각각의 branch가 어떻게 merge 되고 있고 현재 branch의 상태가 어떤 지에 대한 정보를 제공받았다. 시각적인 자료인 그래프를 통해 branch 상태에 대한 정보를 제공받을 수 있어서 만약 branch 사용을 잘못된 경우 이 사실에 대해 쉽고 빠르게 깨달을 수 있어서 Network을 유용하게 사용하였다.



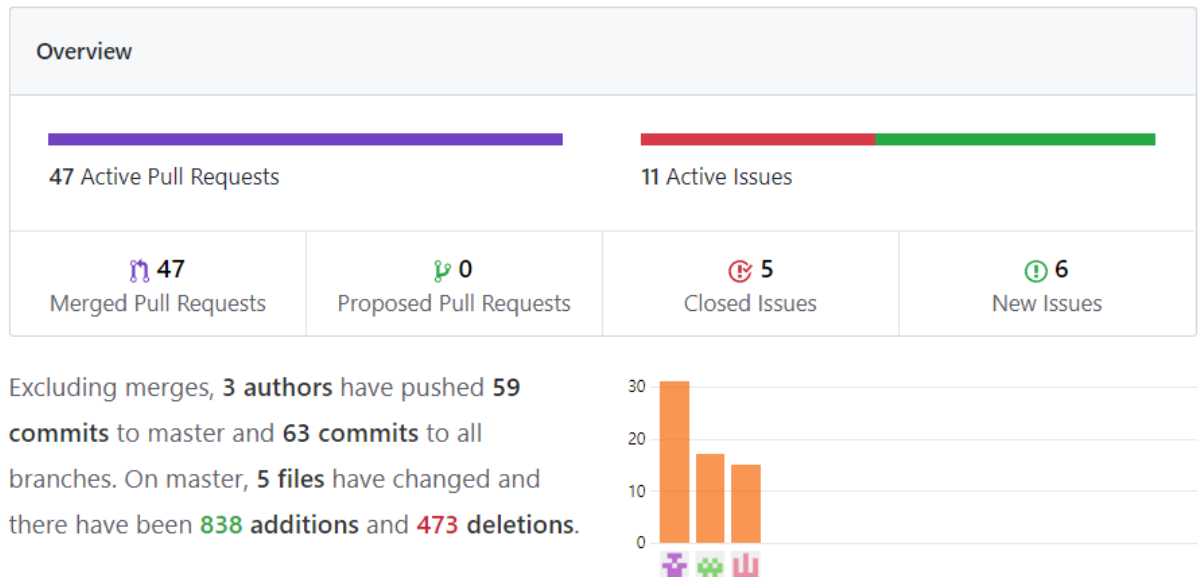
[그림 16] Network

- Pulse와 Contributor

Pulse와 Contributor을 통해 팀원의 활동율을 객관적으로 비교한 자료를 제공받았다. 이를 통해 만약 프로젝트에 활발하게 활동하지 않는 팀원이 있다면 참여를 유도하여 프로젝트 진행이 더디지 않도록 조절하였다. 특히 Pulse은 기간별로 자료를 제공받을 수 있어서 편리하다고 느꼈다.

May 23, 2019 – May 30, 2019

Period: 1 week ▼

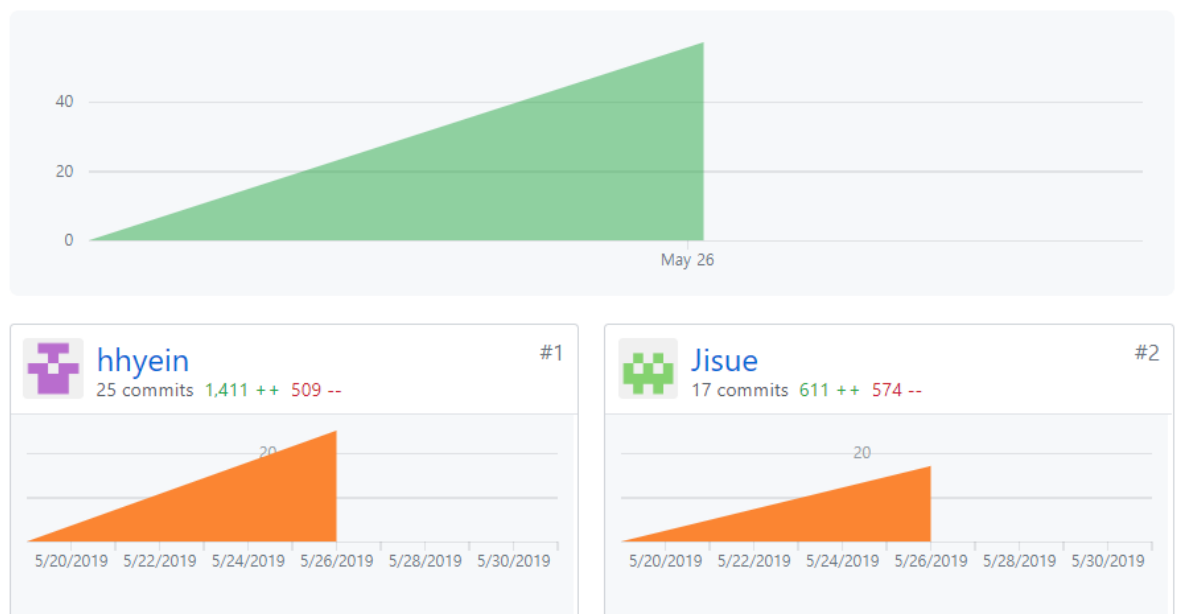


[그림 17] Pulse

May 19, 2019 – May 31, 2019

Contributions: Commits ▼

Contributions to master, excluding merge commits



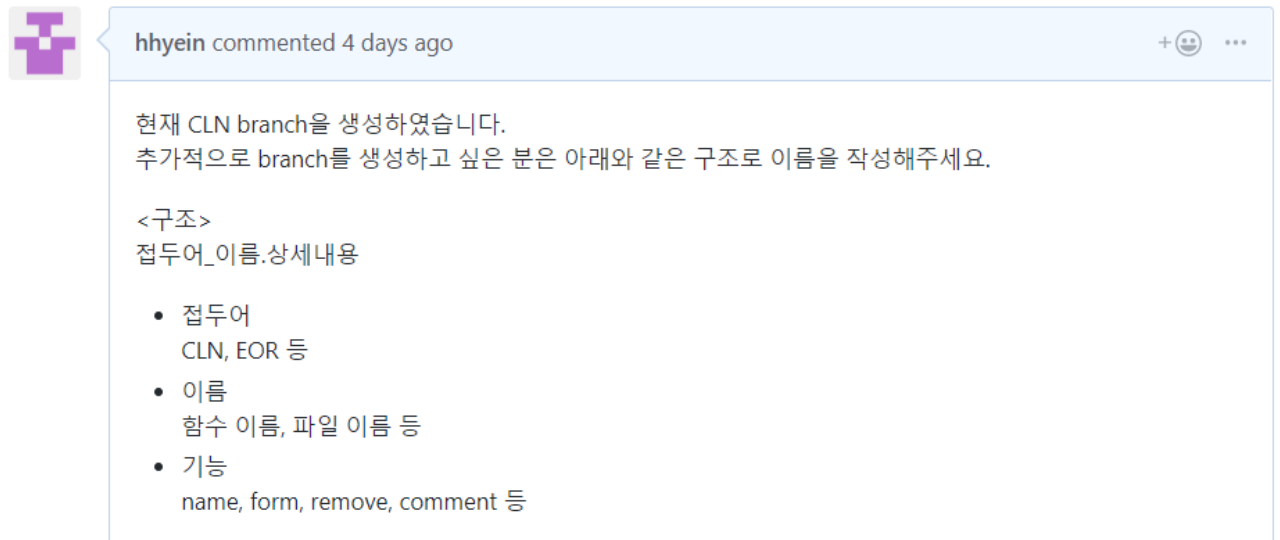
[그림 18] Contributor

나) Git 활용

- branch

작업 공간인 branch를 만들어 안전한 작업 환경에서 오픈소스를 수정하여 프로젝트를 진행하였다. 해당 프로젝트에서는 Master branch 밑에 CLN, EOR, SPT branch를 만들었고 작업을 수행할 때마다 각각의 branch 밑으로 하위 branch를 만들어 오픈소스를 수정하였다. 우선 CLN branch에서는 기존

오픈소스의 가독성이 떨어지는 부분을 수정하는 작업을 수행하였다. CLN branch은 하위 branch가 너무 많이 생겨서 헷갈릴 것 같아 branch 이름에 상세 내용을 붙여 보기 쉽게 분류하였다. 변수 및 함수 명을 의미 있게 변경하는 작업의 branch는 name, 머리 주석 및 함수 주석을 작성하여 코드를 읽는 사람의 이해도를 높이는 작업의 branch는 comment, 일관된 형식을 유지하여 코드를 읽기 편하도록 수정하는 작업의 branch는 form이라는 상세 내용을 붙였다. 또한 EOR branch에서는 기존 오픈소스가 가지고 있는 오류를 고치기 위한 작업을 수행하였다. 마지막으로 SPT branch에서는 여러 개의 기능을 포함하는 작업을 분할하여 최소화하고, 분할 컴파일을 통해 유지 보수를 하기 편하도록 도와주는 작업을 수행하였다. branch 이름 작성 규칙은 [그림 19]에서 볼 수 있다시피 Issues를 통해 팀원끼리 정한 후 공지하였다.



[그림 19] branch 이름 작성 규칙

● commit message

branch를 수정할 때 마다 파일의 변경 사항을 기록하기 위해 commit 메시지를 작성하였다. commit 메시지는 commit 하기 전의 문제점 및 변경 사항에 대한 내용을 작성하였다. commit 메시지는 branch마다 변경 사항에 대한 내용을 읽어볼 수 있기 때문에 모든 branch마다 commit 메시지를 작성하여 프로젝트에 대한 팀원의 이해도를 높였다. commit 메시지 작성 규칙은 [그림 20]에서 볼 수 있다시피 Issues를 통해 팀원끼리 정한 후 공지하였다.



hhyein commented 4 days ago



<구조>

[category] – [simple message]

[detailed description]

- category
commit 성격이 무엇인지 한번에 알 수 있는 단어로 작성
(예) EOR: 에러 수정, ADD: 기능 추가, DOC: 문서화 관련 작업, CLN: 코드 정리, ENH: 개선
- simple message
해당 commit에 대한 간단한 한 줄 설명 작성
- detailed description
commit하기 전 문제점 및 commit 후 무엇이 변했는지 설명

<주의>

과거형으로 작성 금지

마침표로 첫 줄을 끝내지 않기

[그림 20] commit 메시지 작성 규칙

● tag

중요 시점에 tag를 저장하여 매우 편리하게 특정 위치로 이동하거나 운영 및 배포를 하였다. 해당 프로젝트에서는 기능 별로 5개의 중요 시점에 tag를 작성하였고 각각의 tag마다 버전을 관리하였다. 이를 통해 사용자가 자신이 원하는 버전의 소스 파일을 다운로드 받을 수 있도록 하였다. 각각의 중요 시점에 대해서는 아래의 [표 2]을 통해 한 눈에 확인할 수 있다.

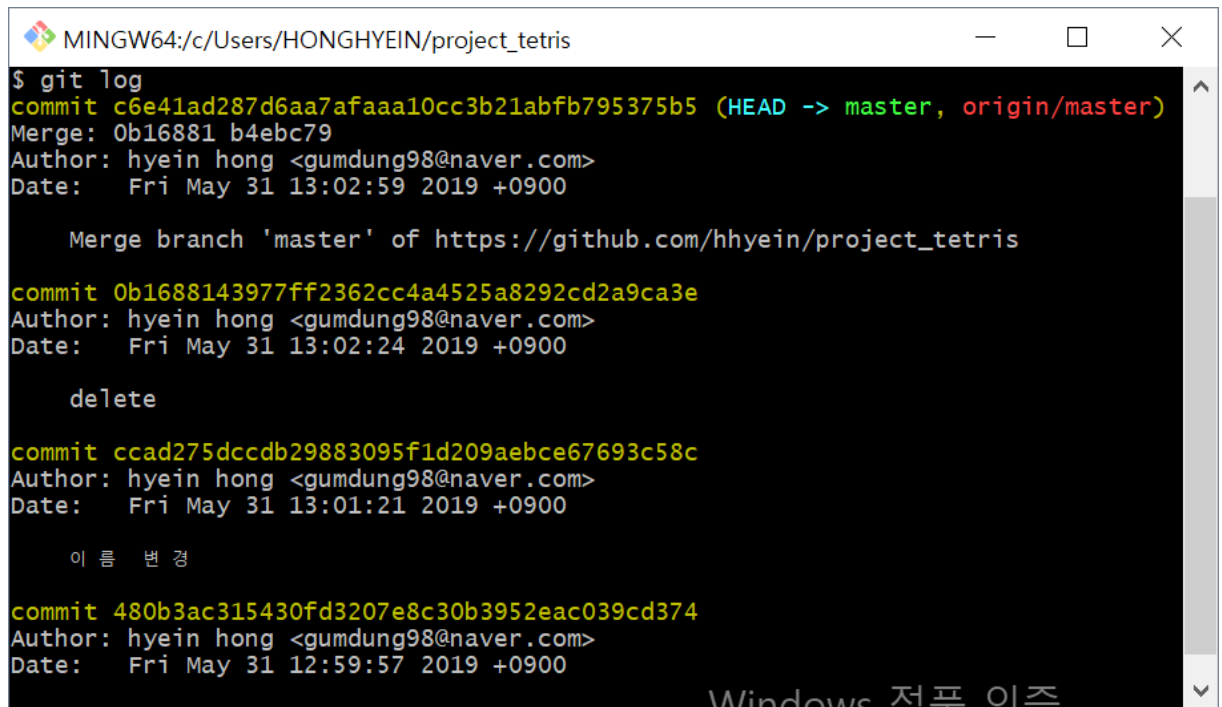
tag	설명
v1.0	오픈소스의 가독성을 높이기 위해 소스 코드를 수정한 버전
v2.0	오픈소스의 가독성을 높이기 위해 주석을 작성한 버전
v3.0	오픈소스의 오류를 수정한 버전
v4.0	기능 최소화 및 분할 컴파일한 버전
v5.0	수정하지 못한 부분이나 보충하고 싶은 부분에 대해 작업한 버전

[표 2] tag 종류 및 설명

● Log

[git log] 명령어는 commit 히스토리를 조회하고 특정 commit의 체크섬을 확인하여 tag를 작성하는 용도로 자주 사용하였다. 하지만 해당 명령어를 실행하면 그래프 형태가 아닌 문장 형태이기 때문에 전체적인 흐름을 파악하는 데에 가독성이 떨어진다는 생각이 들었다. 따라서 [git log --graph] 명령어를 실행하여 현재 repository의 작업 흐름에 대해 그래프로 쉽게 알아볼 수 있었다. 하지만 해당 프로젝트를 진행할 때 충돌을 최대한 줄이기 위해 branch를 상위 branch로 merge할 때마다 최상위

branch인 master에도 즉각 merge하였기 때문에 작업 흐름을 이해하는 데에 큰 도움이 되지 않았다.



```
MINGW64:/c/Users/HONGHYEIN/project_tetris
$ git log
commit c6e41ad287d6aa7afaaa10cc3b21abfb795375b5 (HEAD -> master, origin/master)
Merge: 0b16881 b4ebc79
Author: hyein hong <gumdung98@naver.com>
Date: Fri May 31 13:02:59 2019 +0900

    Merge branch 'master' of https://github.com/hhyein/project_tetris

commit 0b1688143977ff2362cc4a4525a8292cd2a9ca3e
Author: hyein hong <gumdung98@naver.com>
Date: Fri May 31 13:02:24 2019 +0900

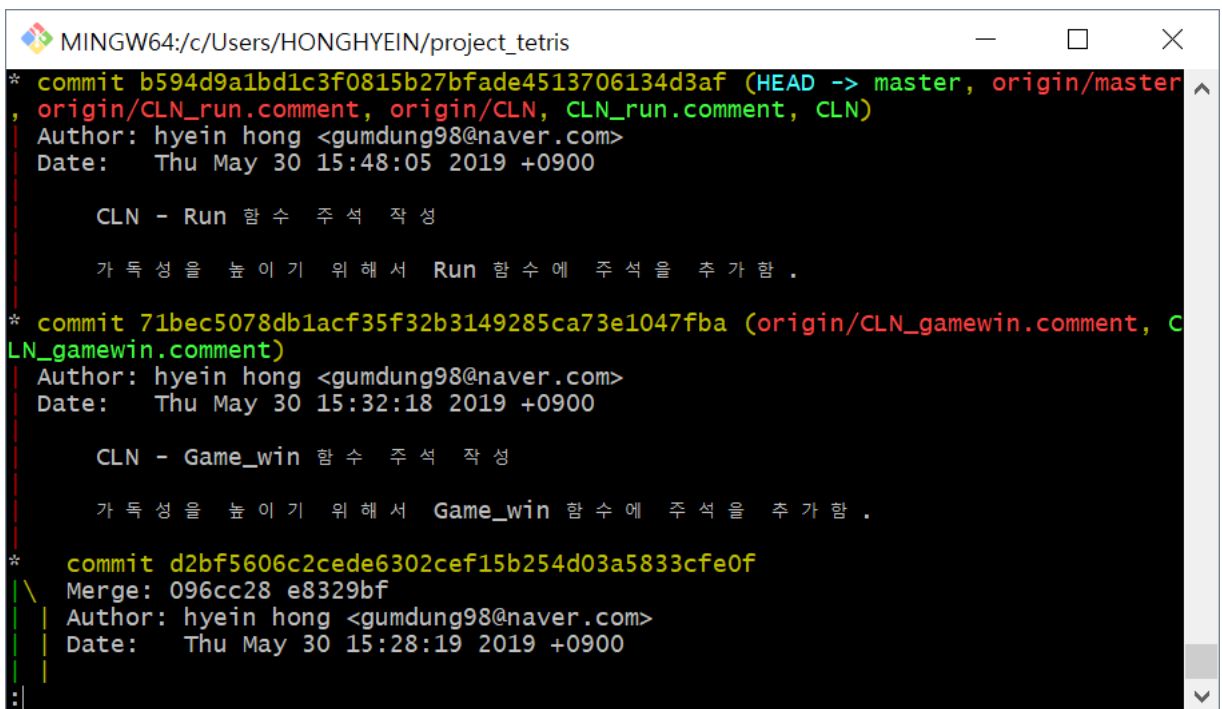
    delete

commit ccad275dccdb29883095f1d209aebce67693c58c
Author: hyein hong <gumdung98@naver.com>
Date: Fri May 31 13:01:21 2019 +0900

    이 름 변 경

commit 480b3ac315430fd3207e8c30b3952eac039cd374
Author: hyein hong <gumdung98@naver.com>
Date: Fri May 31 12:59:57 2019 +0900
```

[그림 21] git log 명령어 실행 예시



```
MINGW64:/c/Users/HONGHYEIN/project_tetris
* commit b594d9a1bd1c3f0815b27bfade4513706134d3af (HEAD -> master, origin/master, origin/CLN_run.comment, origin/CLN, CLN_run.comment, CLN)
| Author: hyein hong <gumdung98@naver.com>
| Date: Thu May 30 15:48:05 2019 +0900
|
| CLN - Run 함수 주석 작성
|
| 가 독 성 을 높 이 기 위 해 서 Run 함수 에 주석 을 추 가 함 .
|
* commit 71bec5078db1acf35f32b3149285ca73e1047fba (origin/CLN_gamewin.comment, CLN_gamewin.comment)
| Author: hyein hong <gumdung98@naver.com>
| Date: Thu May 30 15:32:18 2019 +0900
|
| CLN - Game_win 함수 주석 작성
|
| 가 독 성 을 높 이 기 위 해 서 Game_win 함수 에 주석 을 추 가 함 .
|
* commit d2bf5606c2ced6302cef15b254d03a5833cfe0f
| Merge: 096ccc28 e8329bf
| Author: hyein hong <gumdung98@naver.com>
| Date: Thu May 30 15:28:19 2019 +0900
|
|
```

[그림 22] Log graph

다) 오픈소스 수정

- 함수 및 변수 명 수정

해당 오픈소스는 함수 및 변수 명을 의미 있게 작성하지 않아서 사용자가 소스 코드를 이해하는 데에 어려움을 준다. 따라서 의미 없는 함수 및 변수 명을 적절하게 수정하여 해당 오픈소스의 가독성을 높였다. [그림 23]은 기존 오픈소스의 moveBlock 함수의 일부이다. ww와 k 변수 명은 어떤 기능을

수행하고 어떤 값을 저장한 변수인지 전혀 추측할 수 없다. 또한 var와 tmp 변수는 함수에 선언하였지만 변수를 사용하지 않는다. 따라서 사용자가 변수 명만 보고 해당 변수의 기능 및 역할을 추측할 수 있도록 [그림 24]와 같이 변수 명을 의미 있게 수정하였고, 사용하지 않은 변수를 선언한 코드는 소스 코드를 이해하는 데에 불편함을 주므로 코드를 삭제하였다.

또한 기존 오픈소스는 함수 명의 첫 글자가 대문자가 아니기 때문에 변수 명인지 함수 명인지 분별할 수 없다. 따라서 사용자가 쉽고 빠르게 함수인지 변수인지 분별할 수 있도록 함수 명의 첫 글자를 대문자로 모두 수정하였다. 구체적인 예시는 [표 4]을 통해 확인할 수 있다.

<pre>int ww = 0; int var; int k = 0; int tmp;</pre>	<pre>int last_line = 0; int block_rotation = 0;</pre>
[그림 23] 수정 전 변수 명 예시	[그림 24] 수정 후 변수 명 예시

[표 3] 변수 명 수정 전 후

<pre>int gameWin(void) {</pre>	<pre>int Game_win(void) {</pre>
[그림 25] 수정 전 함수 명 예시	[그림 26] 수정 후 함수 명 예시

[표 4] 함수 명 수정 전 후

● 형식 일관성 유지

해당 오픈소스는 비슷한 기능을 가진 코드임에도 불구하고 형식이 독립적이기 때문에 가독성을 낮추는 부분이 있다. 따라서 형식의 일관성을 유지하도록 코드를 수정하여 해당 오픈소스의 가독성을 높였다. [그림 27]는 기존 오픈소스의 moveBlock 함수의 일부이다. 두 개의 조건문은 비슷한 기능을 가진 코드이지만 형식이 서로 다르다. 따라서 아래의 조건문을 위의 조건문 형식에 맞춰 일관된 형식으로 수정하여 가독성을 높였다.

또한 어떤 조건문에서는 중괄호를 사용하였지만 어떤 조건문에서는 중괄호를 사용하지 않은 부분이 있다. 따라서 중괄호를 사용하지 않은 조건문에 중괄호를 추가해줌으로써 형식을 일관성 있게 유지하였다.

<pre>if (gameWin()) { setCursor(35, 20); printf("GAME WIN"); getchar(); exit(1); } if (gameOver(n)) break;</pre>	<pre>if (Game_win()) { Set_cursor(35, 20); printf("GAME WIN"); getchar(); exit(1); } if (Game_over(block)) { Set_cursor(35, 20); printf("GAME OVER"); getchar(); exit(1); }</pre>
[그림 27] 수정 전 형식 예시	[그림 28] 수정 후 형식 예시

[표 5] 형식 수정 전 후

● 주석 작성

기존 오픈소스에는 머리 주석 및 함수 주석이 작성되어 있지 않다. 따라서 해당 프로젝트에서 머리 주석 및 함수 주석을 작성하여 코드를 읽는 사람의 이해도를 높였다. 머리 주석에는 프로그램에 대한 설명, 사용 방법 및 제한 사항 등에 대한 내용을 작성하였으며, 함수 주석에는 함수의 기능과 인자의 역할에 대한 내용을 작성하였다. 그리고 기존 오픈소스에는 주석이 없어도 함수 및 변수 명을 보고 어떤 기능을 하는지 추측 가능한 작업에 주석을 작성하여 오히려 가독성을 떨어트렸다. 따라서 무분별한 주석을 삭제하여 소스 코드를 읽기 편하게 도움을 주었다. 또한 특별한 작업을 동작하거나 소스 코드만으로 개발자의 의도를 충분히 표현하지 못해서 사용자가 볼 때 이해하기 어려운 소스 코드 부분에는 추가적으로 주석을 작성하였다.

```
/*
파일 명: main.c
작성자: 홍혜인
프로그램 목적: 테트리스 게임을 실행한다.
사용 방식: <-, -> 방향 키로 블록을 이동하고 ↑ 방향 키로 블록을 회전할 수 있다. 또한 스페이스 바로 블록을 보드 하단으로 이동할 수 있다.
사용 파일: block.h, cursor.h, print.h, cursor.c, print.c
제한 사항: 레거시 콘솔 사용의 체크 박스를 해제해야 한다. 그렇지 않으면 보드가 출력되지 않는 오류가 발생한다.
*/
```

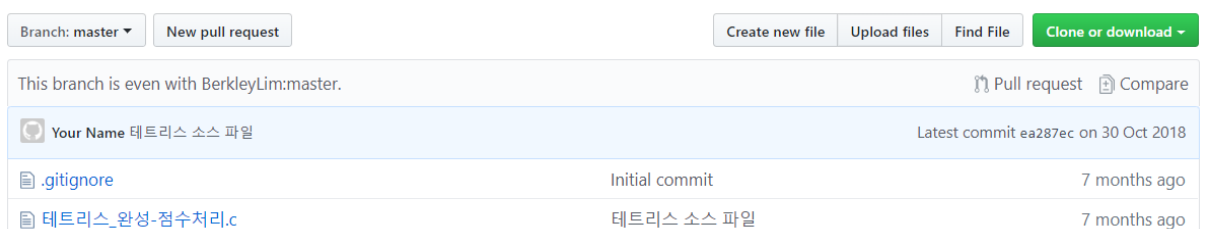
[그림 29] 머리 주석 작성

● 여러 기능을 포함하는 작업 최소화

해당 오픈소스는 여러 기능을 포함하는 함수와 코드가 있다. 하나의 작업이 여러 기능을 포함한다면 유지 보수가 매우 어려워지고 가독성을 떨어트리는 원인이 되므로, 기능 별로 작업을 분할하였다. 예를 들어, 오픈소스의 moveBlock 함수는 게임의 승패 여부를 확인하여 게임을 진행하거나 종료하는 기능과 블록을 움직이는 기능 두 가지의 기능을 갖고 있다. 따라서 moveBlock 함수를 기능 별로 나누어 두 개의 함수로 최소화하였다. 블록을 움직이는 기능의 함수를 더 작게 분할하려고 여러 가지 시도를 해보았지만 이전 블록의 잔상이 남는 오류가 발생하여 더 이상 최소화하지 못하였다.

● 분할 컴파일

해당 오픈소스는 하나의 소스 파일에 모든 코드를 포함하고 있어서 소스 파일이 너무 길고 복잡해져 가독성을 떨어트린다. 따라서 새로운 헤더 파일을 생성하여 소스 파일의 내용을 헤더 파일에 정의하였다. Visual Studio를 통해서는 여러 개의 헤더 파일과 소스 파일을 생성하여 변수와 함수를 기능별로 나누어 완벽하게 분할 컴파일을 하였다. 하지만 Git을 이용하여 분할 컴파일을 한 후 merge 하니까 충돌이 발생하였다. 따라서 해당 프로젝트에서는 기존 소스 파일에 정의되어 있던 블록 배열을 block.h 파일에 선언하는 작업과 cursor와 관련된 기능을 수행하는 cursor.h 파일과 cursor.c 파일을 생성하고 보드에 출력되는 변수와 관련된 기능을 수행하는 print.h 파일과 print.c 파일을 생성하여 분할한 것 외에는 수행하지 못하였다.



[그림 30] 기존 오픈소스 파일

Branch: master ▼ New pull request		Create new file	Upload files	Find File	Clone or download ▼
hhyein 보충 Latest commit 32f3541 17 minutes ago					
.gitignore	add gitignore	4 days ago			
LICENSE	Create LICENSE	4 days ago			
README.md	Update README.md	10 hours ago			
block.h	보충	17 minutes ago			
cursor.c	보충	17 minutes ago			
cursor.h	보충	17 minutes ago			
scorelevel.c	보충	17 minutes ago			
scorelevel.h	보충	17 minutes ago			
테트리스_완성-점수처리.c	보충	17 minutes ago			

[그림 31] 현재 오픈소스 파일

● 오류 수정

해당 오픈소스는 랜덤한 모양의 블록을 생성하지 못하고 같은 모양의 블록만 생성하는 오류와 블록 바로 밑에 보드가 있을 경우 보드가 출력되지 않는 오류를 가지고 있다.

먼저 첫 번째 오류는 수학적 계산을 오류이기 때문에, 올바른 계산을 하는 코드로 바꾸어 랜덤한 모양의 블록을 생성하도록 오류를 수정하였다.

<pre>n = (rand() % RAND) * 4; n = rand() % RAND; n = rand() % 7; n = n * 4; n = 6;</pre>	<pre>block = (rand() % 7) * 4;</pre>
[그림 32] 수정 전 블록 모양 오류	[그림 33] 수정 후 블록 모양 오류

[표 6] 블록 모양 오류 수정 전 후

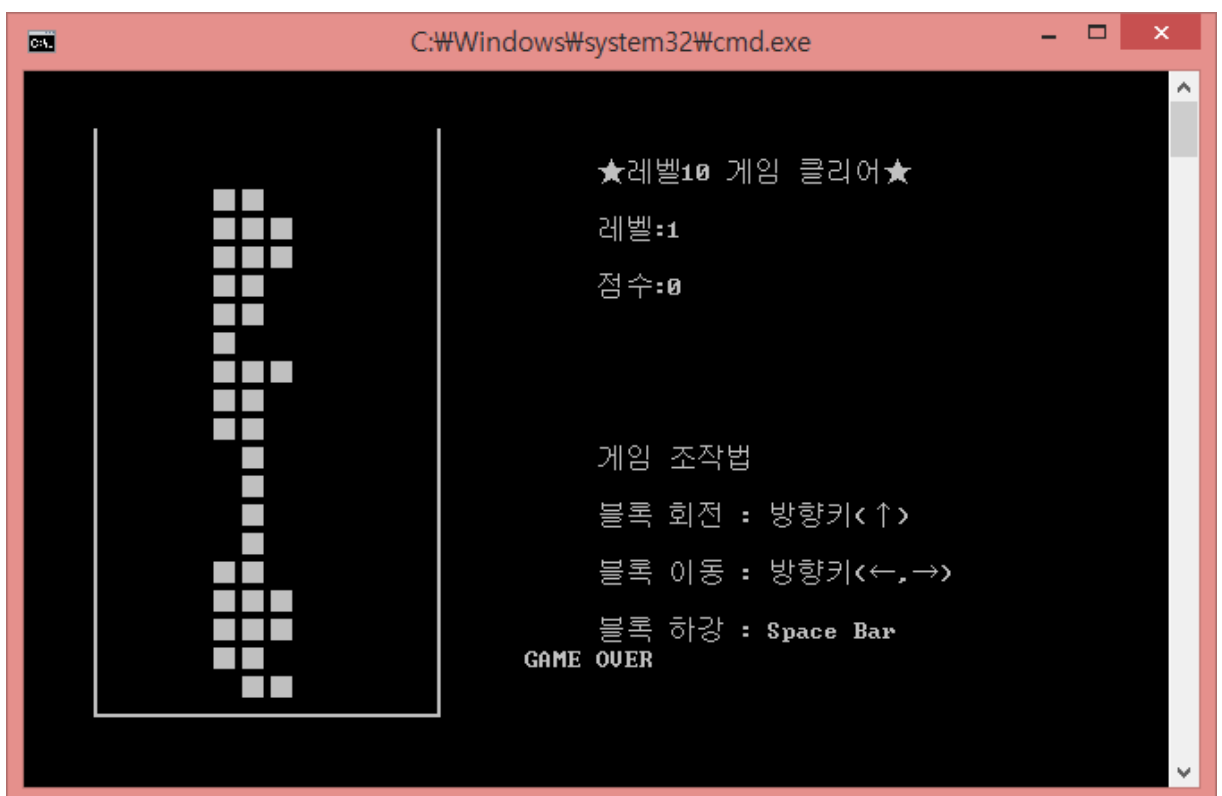
두 번째 오류는 Visual Studio 환경 설정 때문에 발생한 오류이다. Window 10이 업데이트를 하면서 콘솔의 기능이 개선되었고, 개선된 콘솔을 사용하기 위해서는 속성에서 레거시 콘솔 사용의 체크 박스를 해제하고 재실행 해야 한다. 즉, 원작자는 레거시 콘솔 사용의 체크 박스를 해제하였지만 팀원들은 레거시 콘솔 사용의 체크 박스를 해제하지 않아서 보드가 출력되지 않은 것이다. 따라서 팀원끼리 이 문제를 오류라고 해야 하는지 아닌 지에 대해 Issues을 통해 토론하였는데, 오류가 아니라고 판단하였지만 프로그램 머리 주석에 제한 사항으로 해당 문제에 대한 내용을 작성을 하자고 결론을 내렸다.



[그림 34] 레거시 콘솔 사용의 체크 박스를 해제한 후 실행한 화면

- 기능 추가

기존 오픈소스는 테트리스 게임 조작 방법에 대한 설명을 제공하지 않는다. 따라서 콘솔 창에 테트리스 게임 조작 방법에 대한 설명을 출력하는 기능을 추가하여 사용자가 테트리스 게임을 하는데 도움을 줄 수 있도록 한다.



[그림 35] 기능 추가 후 실행한 화면

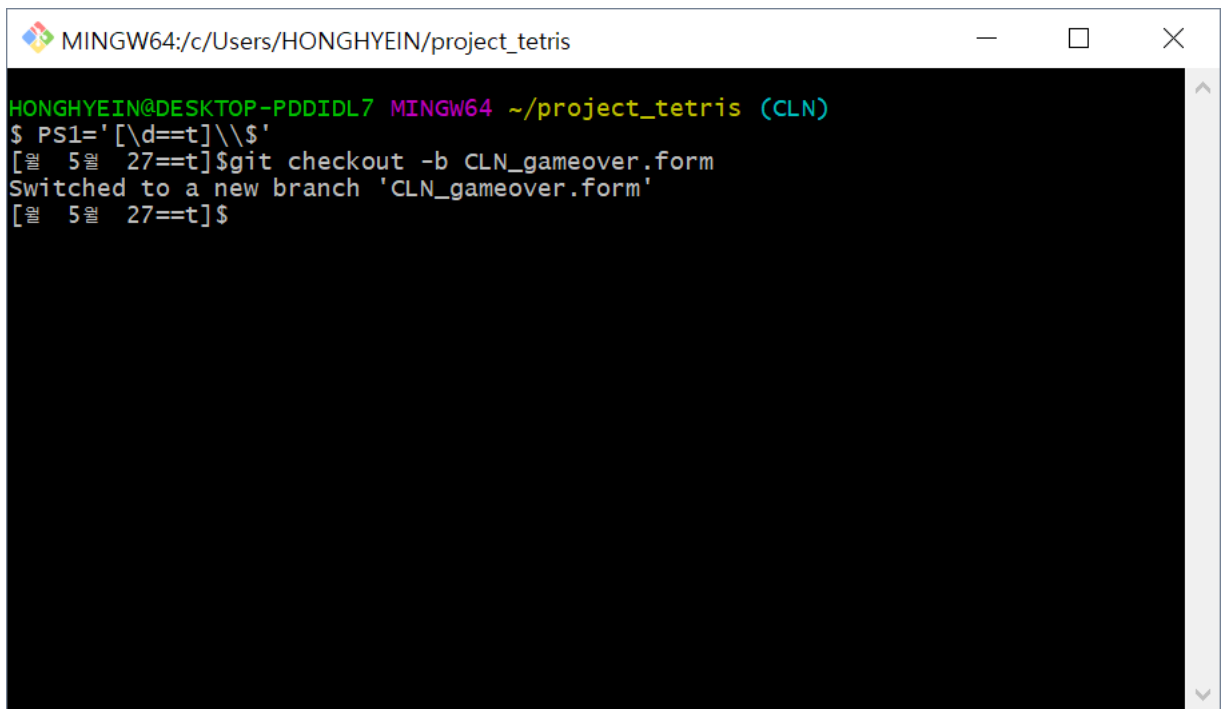
IV. 프로젝트 진행 과정 및 시행 착오

가) 프로젝트 진행 과정

해당 프로젝트는 [표 2]에서 볼 수 있다시피 5개의 버전의 순서대로 프로젝트를 진행하였다. 먼저 오픈소스의 가독성을 높이기 위해 소스 코드를 수정하였다. 그 후에 불필요한 주석을 삭제하거나 필요한 주석을 작성하여 가독성 목적으로의 오픈소스 수정을 끝마쳤다. 그 후에 해당 오픈소스가 가지고 있는 오류를 수정하고, 여러 개의 기능을 가지고 있는 함수와 소스 파일을 분할 컴파일 하여 유지보수가 편할 수 있도록 오픈소스를 수정하였다. 원래의 계획으로는 4개의 버전만에 프로젝트가 끝나야 했지만, 해당 버전에서 수정하지 못하고 나중에 발견한 문제점이나 보충하고 싶은 부분이 생겨서 추가적으로 버전을 만들어 오픈소스를 더욱 발전시켰다.

각 단계마다 수정 과정은 다음과 같다. 우선 자신의 로컬 branch에서 오픈소스를 수정한 후 add와 commit을 한다. 그 이후에 해당 branch를 원격 저장소로 push한 후 pull requests을 작성한다. 만약 팀원의 허락을 받았다면 해당 branch를 상위 branch에 merge한다. 자세한 과정은 아래에 구체적인 예시를 들어 설명하였다.

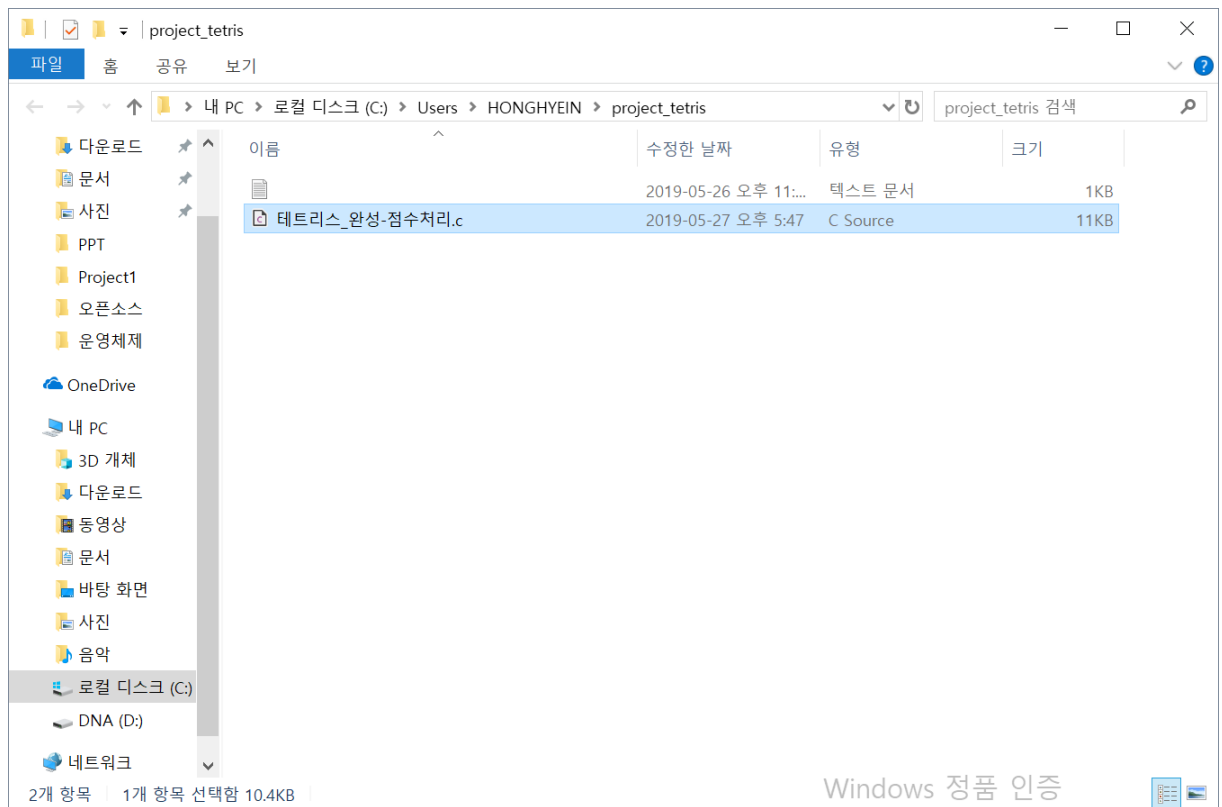
1. 항상 자신의 작업을 시작하기 전에 원격 저장소에 있는 branch를 fetch한 후 수동 병합을 해서 동기화한다.
2. CLN branch의 하위 branch인 CLN_gameover.form branch를 생성한 후 해당 branch로 이동한다.



```
MINGW64:c:/Users/HONGHYEIN/project_tetris
HONGHYEIN@DESKTOP-PDDIDL7 MINGW64 ~/project_tetris (CLN)
$ PS1='\d==t\\$'
[월 5월 27==t]$git checkout -b CLN_gameover.form
Switched to a new branch 'CLN_gameover.form'
[월 5월 27==t]$
```

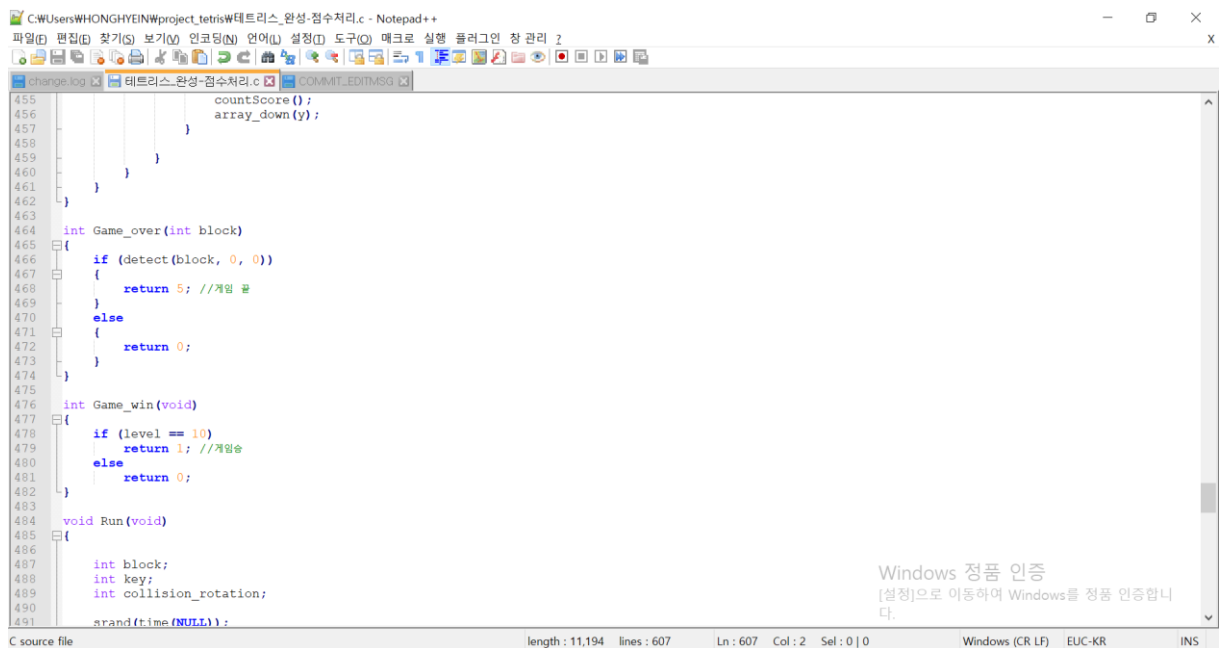
[그림 36] 프로젝트 진행 과정 1

3. 프로젝트의 디렉토리에서 수정할 파일을 우 클릭하고 Edit with Notepad++을 클릭한다.



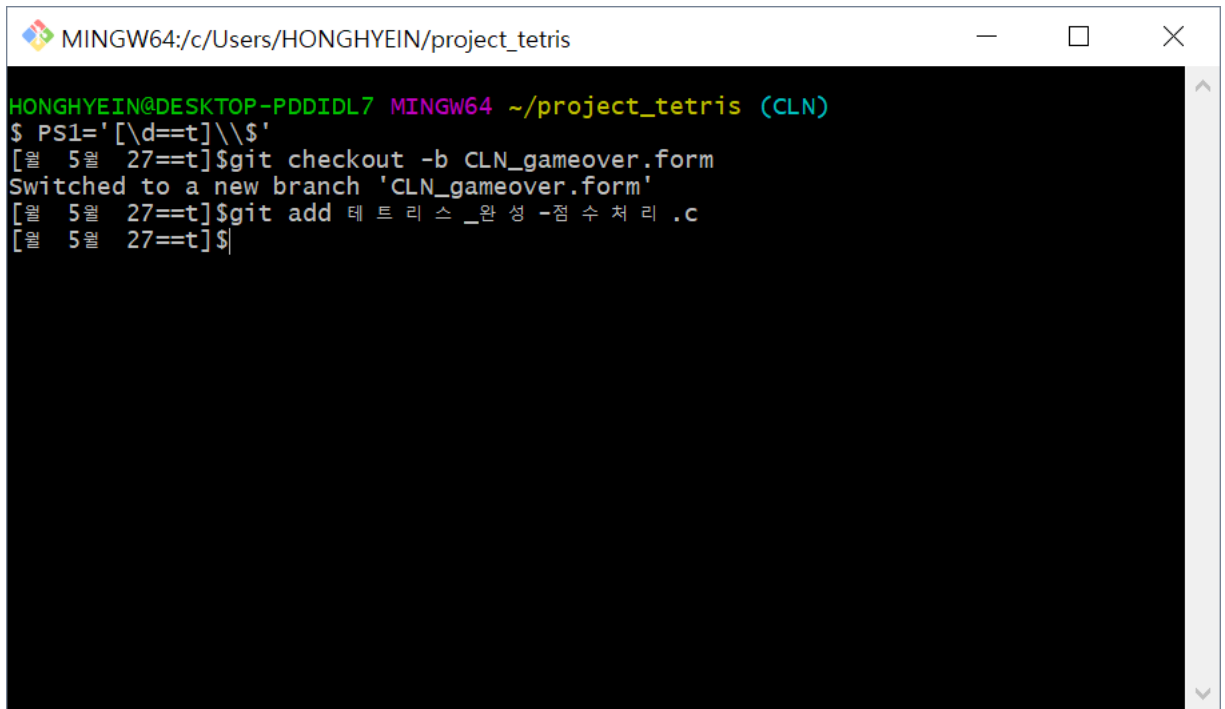
[그림 37] 프로젝트 진행 과정 2

4. Visual Studio에서 파일을 수정한 후 Notepad++에 붙여 넣어 저장 버튼을 누른다.



[그림 38] 프로젝트 진행 과정 3

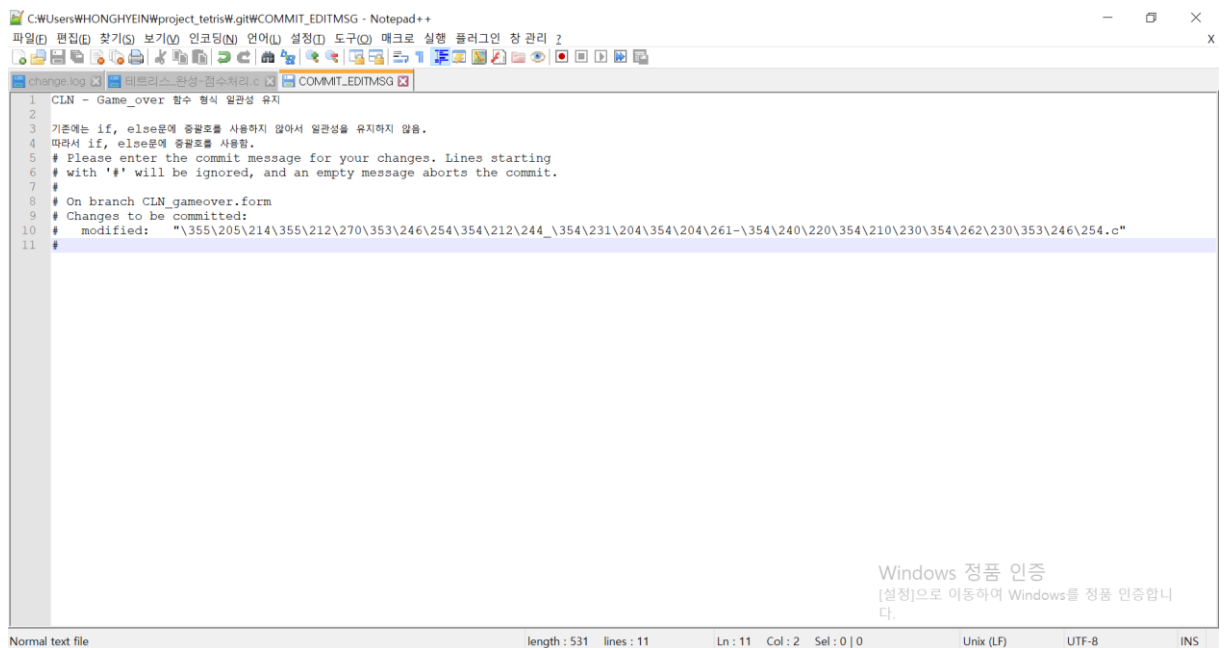
5. [git add 파일] 명령어를 실행하여 수정한 파일을 add 한다.



```
MINGW64/c/Users/HONGHYEIN/project_tetris
HONGHYEIN@DESKTOP-PDDIDL7 MINGW64 ~/project_tetris (CLN)
$ PS1='[\d==t]\\$'
[월 5월 27==t]$git checkout -b CLN_gameover.form
Switched to a new branch 'CLN_gameover.form'
[월 5월 27==t]$git add 테트리스_완성_점수처리.c
[월 5월 27==t]$
```

[그림 39] 프로젝트 진행 과정 4

6. [git commit] 명령어를 실행하여 commit 메시지를 작성한 후 저장하여 수정한 파일을 commit 한다.



```
C:\Users\HONGHYEIN\project_tetris\git\COMMIT_EDITMSG - Notepad++
파일(F) 편집(E) 찾기(S) 보기(V) 인코딩(N) 언어(L) 설정(O) 도구(D) 매크로 실행 플러그인 창 관리 ?
change log 레트리스_완성_점수처리.c COMMIT_EDITMSG
1 CLN - Game_over 함수 형식 일관성 유지
2
3 기존에는 if, else문에 중괄호를 사용하지 않아서 일관성을 유지하지 않음.
4 따라서 if, else문에 중괄호를 사용함.
5 # Please enter the commit message for your changes. Lines starting
6 # with '#' will be ignored, and an empty message aborts the commit.
7 #
8 # On branch CLN_gameover.form
9 # Changes to be committed:
10 #   modified:   "\355\205\214\355\212\270\353\246\254\354\212\244_\354\231\204\354\204\261-\354\240\220\354\210\230\354\262\230\353\246\254.c"
11 #
```

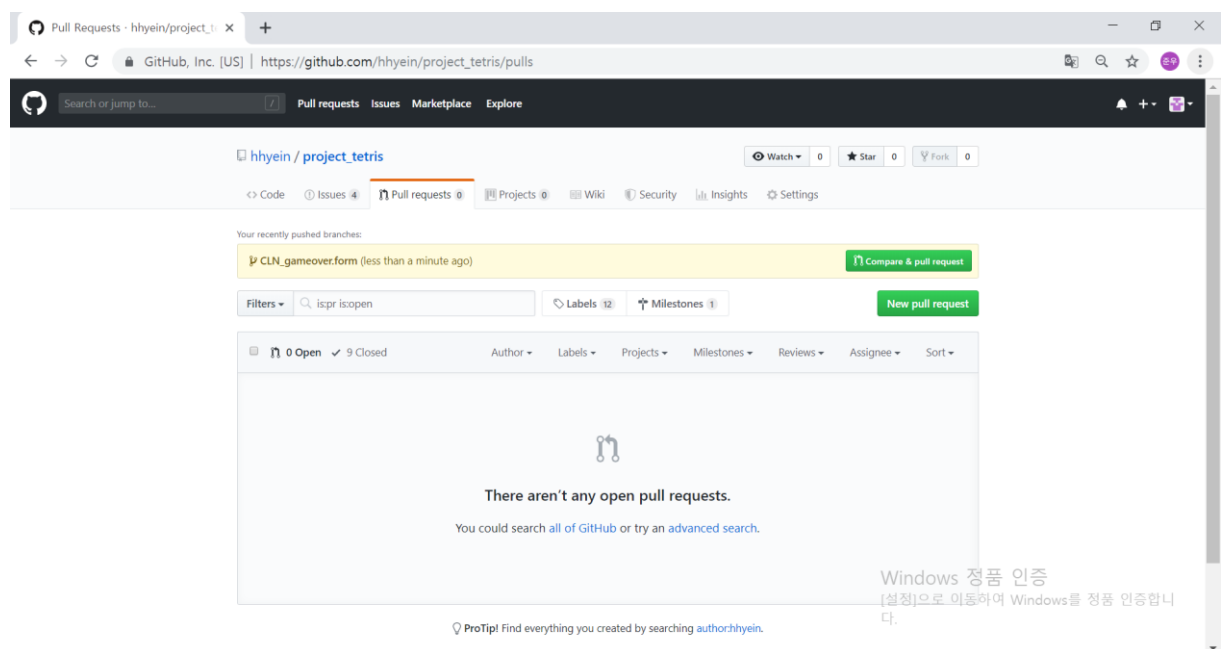
[그림 40] 프로젝트 진행 과정 5

7. [git push origin branch] 명령어를 실행하여 해당 branch를 원격 저장소로 push 한다. 여기서 항상 주의해야 할 사항은, push 하기 전에 원격 저장소에 있는 branch를 fetch한 후 수동 병합을 해야 한다는 것이다. 만약 fetch와 수동 병합을 하지 않으면 파일을 수정하고 있는 도중에 팀원이 다른 부분을 수정하여 충돌이 생길 수 있다. 따라서 충돌을 방지하기 위해 꼭 해야 한다.


```
MINGW64:/c/Users/HONGHYEIN/project_tetris
HONGHYEIN@DESKTOP-PDDIDL7 MINGW64 ~/project_tetris (CLN)
$ PS1='[\d==t]\\$'
[월 5월 27==t]$git checkout -b CLN_gameover.form
Switched to a new branch 'CLN_gameover.form'
[월 5월 27==t]$git add 테트리스_완성_점수처리.c
[월 5월 27==t]$git commit
[CLN_gameover.form dde5c38] CLN - Game_over 함수 형식 일관성 유지
1 file changed, 4 insertions(+)
[월 5월 27==t]$git push origin CLN_gameover.form
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 463 bytes | 463.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
remote:
remote: Create a pull request for 'CLN_gameover.form' on GitHub by visiting:
remote:   https://github.com/hhyein/project_tetris/pull/new/CLN_gameover.form
remote:
To https://github.com/hhyein/project_tetris.git
 * [new branch]      CLN_gameover.form -> CLN_gameover.form
[월 5월 27==t]$
```

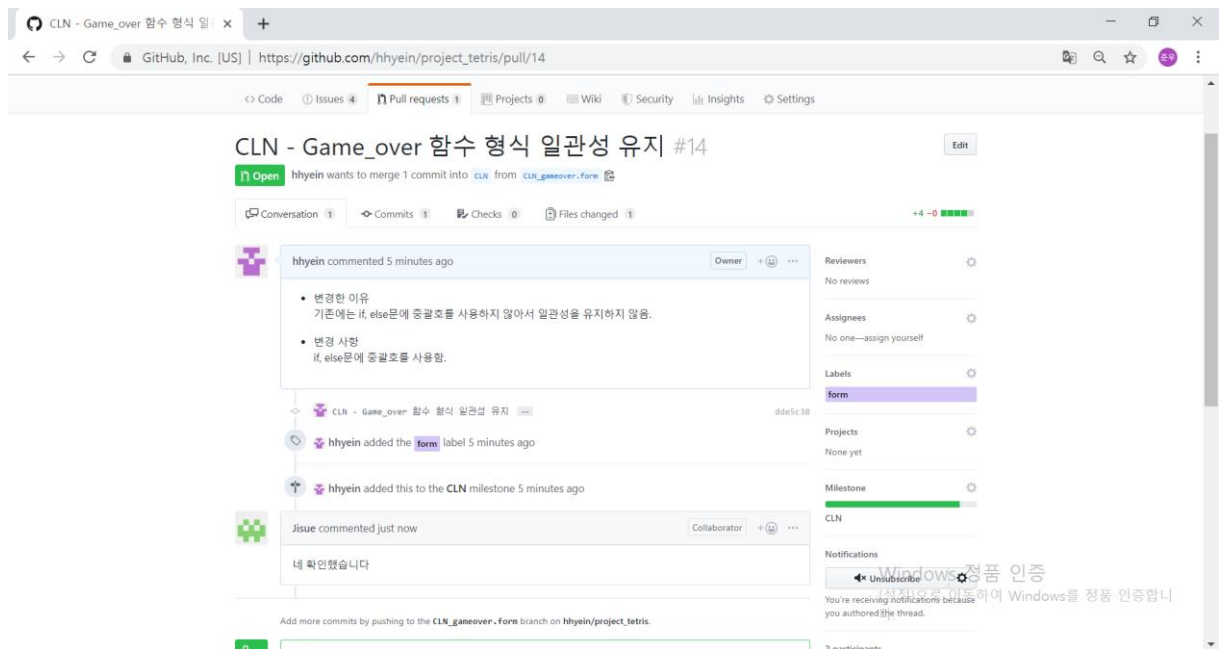
[그림 41] 프로젝트 진행 과정 6

8. 해당 branch에 대한 Pull requests를 작성한다.



[그림 42] 프로젝트 진행 과정 7

9. 팀원이 Pull requests를 허락한다는 comment를 작성한 것을 확인한다.



[그림 43] 프로젝트 진행 과정 8

10. 상위 branch로 이동한 후 [git merge branch] 명령어를 실행하여 해당 branch를 merge 하고, merge 된 상위 branch를 원격 저장소로 push 한다.

```

MINGW64/c/Users/HONGHYEIN/project_tetris
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 463 bytes | 463.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
remote:
remote: Create a pull request for 'CLN_gameover.form' on GitHub by visiting:
remote:   https://github.com/hhyein/project_tetris/pull/new/CLN_gameover.form
remote:
To https://github.com/hhyein/project_tetris.git
* [new branch]      CLN_gameover.form -> CLN_gameover.form
[월 5월 27==t]$git checkout CLN
Switched to branch 'CLN'
[월 5월 27==t]$git merge CLN_gameover.form
Updating 7a8dd68..dde5c38
Fast-forward
...04\354\204\261-\354\240\220\354\210\230\354\262\230\353\246\254.c" | 4 +++++
1 file changed, 4 insertions(+)
[월 5월 27==t]$git push origin CLN
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/hhyein/project_tetris.git
7a8dd68..dde5c38  CLN -> CLN
[월 5월 27==t]$

```

[그림 44] 프로젝트 진행 과정 9

나) 시행 착오

해당 프로젝트는 잘못된 merge로 인해 repository를 2번 삭제하고 다시 만들어 진행하였다. 첫 번째 실수는 Github에서 merge 한 것이다. 프로젝트를 진행하기 초반에 한 팀원이 다른 팀원의 동의 없이 Pull request에 있는 merge 버튼을 눌렀다. 이로 인해 해당 수정 사항에 대한 내용을 확인하지 못한 채 상위 branch에 해당 branch가 merge 되어버렸고, 결국 기존의 repository를 삭제하고 다시 만들게 되었다. 이

일을 통해 팀원들은 함부로 merge을 하면 안된다는 점을 깨닫고 다시는 이러한 일이 생기지 않도록 주의하였다. 하지만 얼마 지나지 않아 두 번째 실수가 발생하였다. 해당 프로젝트는 master branch 밑에 CLN과 EOR branch가 있다. 먼저 CLN branch을 고친 후에 master에 merge 하고, EOR branch을 변경한 후에 master branch에 merge 하였다. 여기까지는 어떠한 문제점도 발생하지 않았다. 하지만 그 후에 CLN branch을 다시 수정하고 master branch에 merge하자 충돌이 발생하였다. 그 때 당시에 팀원들은 CLN과 EOR은 같은 master branch의 밑으로 뻗은 branch이기 때문에 master branch가 변경되면 하위 branch의 내용도 당연히 master와 동일하게 바뀌어 있는 줄 알았다. 이러한 문제가 발생한 당시에는 왜 충돌이 발생하는지 깨닫지 못해서 repository을 삭제하고 또 다시 생성하였다. 하지만 프로젝트를 진행하다 보니 그 때 당시에 왜 그러한 문제점이 생기게 되었는지 알게 되어 매우 아쉽다는 생각이 든다.

또한 Visual Studio을 통해서는 여러 개의 헤더 파일과 소스 파일을 생성하여 변수와 함수를 기능별로 나누어 완벽하게 분할 컴파일을 하였는데, Git을 이용하여 분할 컴파일을 한 후 merge 하니까 충돌이 발생하는지 아직까지도 이해하지 못하였다. 추후에 오픈소스 프로젝트가 끝나더라도 왜 이러한 문제가 발생하였는지 분석할 예정이다.

마지막으로 Github의 Projects 기능과 관련된 어려움이 있었다. Projects 기능을 사용하기 위해 Projects란 어떤 기능을 제공해주는 도구인지 찾아보았다. 하지만 Projects에 대한 자료가 거의 없어서 어려움을 느꼈다. 따라서 아무 것도 모르는 상황이지만 일단 Projects을 생성하고 일일이 클릭해보면서 어떻게 사용해야 하는지 경험을 통해 배우게 되었다.

V. 역할 분담

오픈소스 코드 수정은 각자 오픈소스를 분석한 부분에 대해 수정하였다. 하지만 팀원 모두가 오픈소스를 구체적으로 파악하고 있어야 하므로 자신이 수정하지 않는 부분에 대해서도 숙지하였다. 또한 비교적 쉬운 역할 수행을 맡은 사람이나 자신의 역할을 마친 사람은 다른 팀원을 도와 프로젝트의 진행이 더디지 않도록 조절하였다. 자세한 내용은 아래의 [표 7], [표 8], [표 9]에서 확인할 수 있다.

기능	역할 분담
README, LICENSE 파일 생성	홍혜인
Release	김지수, 홍혜인
Issues, Pull requests	김지수, 이승민, 홍혜인
Labels, Milestones, Projects	김지수, 이승민, 홍혜인
Wiki	홍혜인

[표 7] Github 활용 역할 분담

기능	역할 분담
branch	김지수, 이승민, 홍혜인
commit 메시지	김지수, 이승민, 홍혜인
tag	김지수, 홍혜인

[표 8] Git 활용 역할 분담

기능	역할 분담
가독성	김지수, 이승민, 홍혜인
오류 수정	김지수, 이승민, 홍혜인
함수 최소화	김지수, 홍혜인
분할 컴파일	김지수, 홍혜인
기능 추가	홍혜인

[표 9] 오픈소스 수정 역할 분담

VI. 참고문헌

테트리스

<https://ko.wikipedia.org/wiki/%ED%85%8C%ED%8A%B8%EB%A6%AC%EC%8A%A4>

Visual Studio Community 2017 소프트웨어 사용권 계약서

<https://visualstudio.microsoft.com/ko/license-terms/mlt553321/>

오픈소스 repository

<https://github.com/BerkleyLim/Tetris>

레거시 콘솔 체크 박스

[https://docs.microsoft.com/en-us/previous-versions/orphan-topics/ws.11/mt427362\(v=ws.11\)](https://docs.microsoft.com/en-us/previous-versions/orphan-topics/ws.11/mt427362(v=ws.11))