OSSProj 주간보고서

이번 주 주요 내용	• 피드백을 반영한 수정 • 하드모드 & 속도 조절 기능 구현

교과목명	오픈소스 소프트웨어 프로젝트			**담당교수**	김동호 교수님
과제명	오픈소스를 활용한 테트리스 기능 개발			**팀명**	DoitDoit
회의일자	11/10 11/13	**시간**	15:00 - 17:00	**장소**	아리수, Webex

내용

1. 수정 요구사항 개선

* 수정 요구 사항이었던 변수들을 다른 사람들이 이해하고 수정하기 쉽도록 별도의 파이썬 파일을 만들어 분류 작업 진행 중

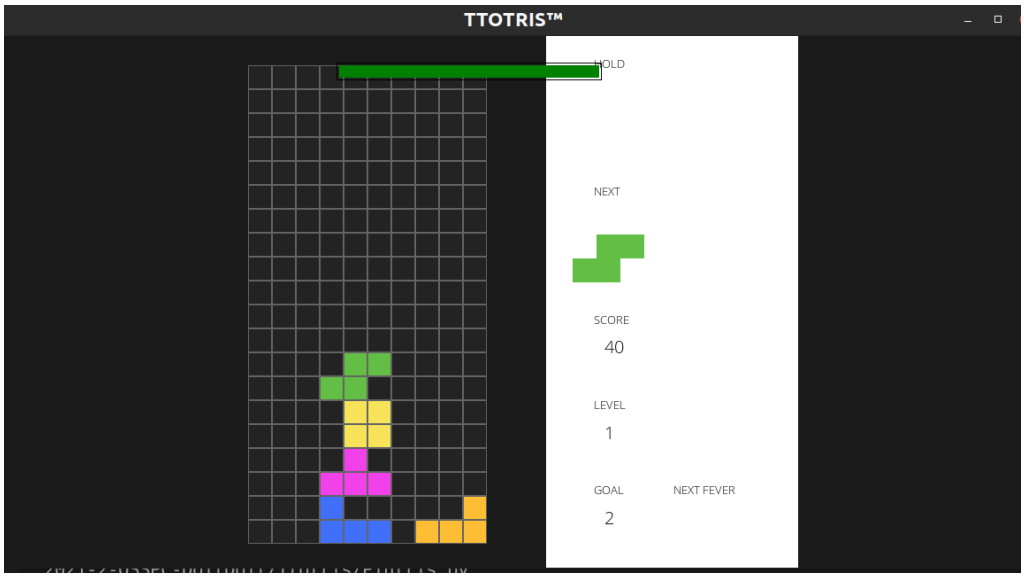
Pintris.py M | init_values.py M | ui.py

Ttotris > init_values.py > ...

```
1  import pygame
2  from pygame import font
3
4  class values:
5      feverAddingTime = [0,1,2,3,4] ## 피버 조건 달성시 추가되는 time
6      feverTimeAddScore = [0,1500,5000,15000,30000] ## 추가적인 피버타임을 얻기위한 점수 구
7      feverBlockGoal = 3 ## 몇개의 블럭을 격파해야 피버가 주어지는지를 정하는 변수
8
```

2. 구현 완료된 기능 검토 & 수정

1) 구현 완료된 피버타임 수정



- 기존: 피버타임 발생 시, 폭탄 이미지가 지속적으로 나타남 → 게임에 방해가 되므로 삭제

- 수정: 피버타임이 얼마나 남았는지 유저가 직관적으로 알 수 있도록 타이머 bar 추가
- 추가 고려 사항: 타이머 bar를 적절한 위치로 배치할 계획

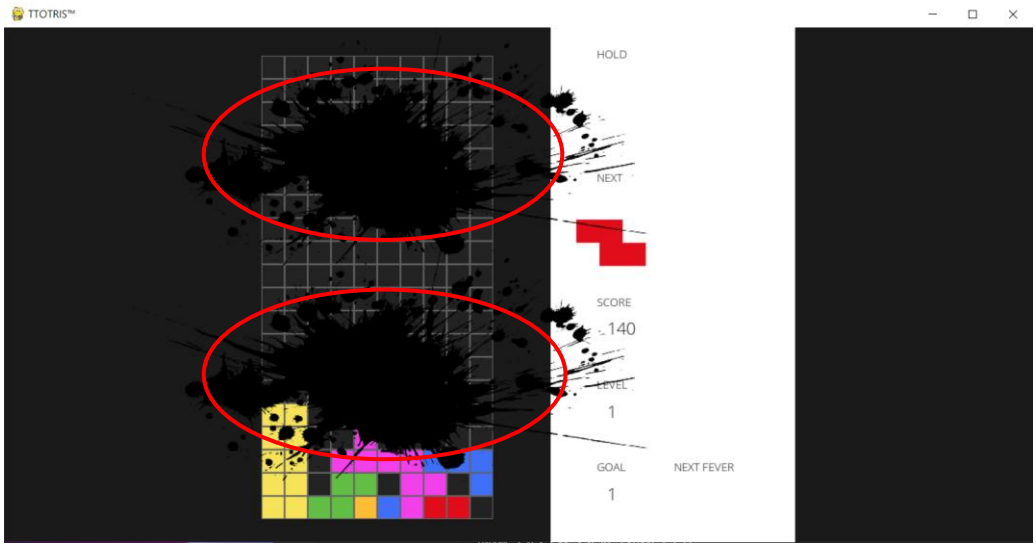
2) 구현 완료된 aws 기능 수정

```
if DIFFICULTY_NAMES[current_selected] == "NORMAL":
    cursor = tetris.cursor()
    sql = "SELECT COUNT(id) FROM NORMAL"
    cursor.execute(sql)
    num = cursor.fetchone()
    for i in range(int(num[0])):
        if i > 2:
            continue
        query = "SELECT * FROM NORMAL ORDER BY score DESC"
        cursor.execute(query)
        datas = cursor.fetchmany(size =int(num[0]))
        ScoreBoard = font2.render(''.join(str(i+1)+'st  '+str(datas[i][0])+'    '+str(datas[i][1]))
        screen.blit(ScoreBoard, ScoreBoard.get_rect(center=(SCREEN_WIDTH / 11, ((SCREEN_HEIGHT * 0
```

- 기존: aws의 모든 기록을 보여주는 문제점 발견
- 수정: count함수를 추가하여 최대 3등까지의 기록을 보여주도록 수정

3. 하드 모드 구현

1) 지난주 회의록에서 예상했던 화면과 동일하게 구현 완료
2) 게임을 진행해본 결과, 장애물이 하나인 것보다 두개일 때 시야 방해에 효과적



: 특정 조건에서, 시야를 방해하는 장애물 등장 화면

3) 추가 고려 사항
  - 장애물이 등장하는 조건 정하기
  - 현재 시야를 방해하는 장애물이 깜빡이도록 구현되어 있는데, 깜빡이는 간격을 더 길게 늘이거나 깜빡이지 않고 시야만 가릴 수 있도록 하는 방안 고려
  - 장애물 기능 이외에 게임 난이도를 높일 수 있는 기능을 추가할 필요가 있을 경우 추가하는 방안 고려

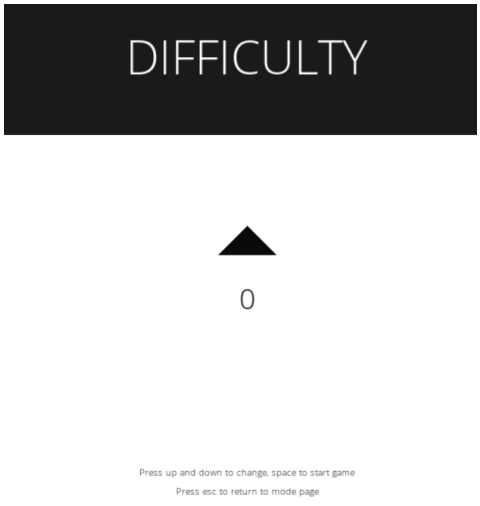
4. 모드별 게임 시작 전 속도 조절 페이지 구현

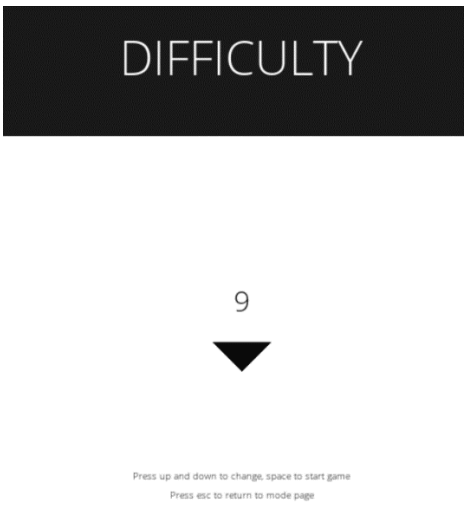
1) 모드를 선택하고 게임이 시작되기 전 속도 조절 페이지 구현 완료
2) 위, 아래 방향키를 통해 0~9까지의 범위에서 초기 속도(difficulty)를 조절하여 선택 가능

3) 초기 속도 선택 후, 스페이스바를 누르면 코드 변수 speed_change에 따라 게임 초기 속도가 결정되고 게임이 시작됨

: 속도 조절 페이지와 기능 구현 완료 화면

4) 아래 사진에서 보이는 것처럼, 최소 숫자인 0에서는 속도를 증가하는 위쪽 화살표만, 최대 숫자인 9에서는 속도를 낮출 수 있는 아래쪽 화살표만 나타남 → 유저가 최소, 최대 숫자임을 인지할 수 있도록 구현





5) 발견된 이슈

- 게임이 종료된 후 메인 페이지로 돌아가 게임을 새로 시작할 때, difficulty가 새로 0에서 시작하는 것이 아닌 이전에 설정했던 difficulty로 저장되어 있는 오류 발견 → 게임 루프에서 K_RETURN마다 초기 변수들을 다시 초기화해줄 때 difficulty 변수 또한 초기화해주도록 함으로써 오류 수정 완료

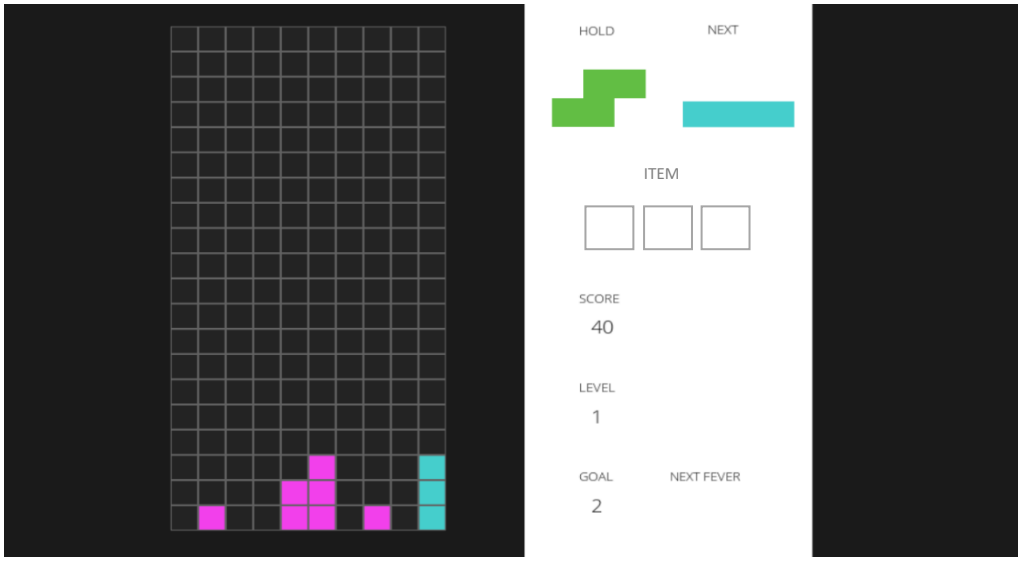
- 같은 맥락으로, Difficulty page에서 esc 키를 통해 이전화면(Mode page)로 돌아가도 difficulty가 저장되어 있음 → Mode page에 돌아가면 difficulty=0이 되도록 초기화하여 오류 수정 완료

5. 협업 규칙

- 이전까지의 git 사용 방안: 프로젝트 초반에는 타임라인에 맞추어 aws 연동과 피버타임 수정에 대해서만 구현하고 나머지 기능들은 구상하는 단계였기 때문에 충돌 발생 문제가 없었지만 여러 기능이 함께 구현되면서 충돌이 발생
- 변경된 git 사용 방안: 실습 시간에 배운 git 사용법대로 open request 후 충돌 발생 시 팀원들과의 논의 후 수정 과정을 거쳐 merge하는 방안 채택
- 팀원들 전체에 merge 권한을 부여함

6. 다음주 진행 예정 사항

1) 구현이 완료된 기능 지속적 검토&수정
2) 하드 모드 구현 완성
3) 아이템 모드 구상



: 아이템 모드 예상 화면) 화면 우측 ITEM 칸에 아이템을 구현할 계획

- 아이템 종류가 조건에 따라 랜덤으로 생성되고 아이템 보드에 아이템(이미지)이 삽입됨
- Z키를 통해 아이템을 사용 가능
- 인벤토리에는 아이템을 최대 3개까지 저장 가능하며, 인벤토리가 가득 찬 경우 아이템을 추가 획득할 수 없도록 구현할 예정