YAMI TETRIS

중간 보고서

냠냠쩝쩝 조

<팀원>
20161112546 산업시스템공학과 조성우
20161112582 산업시스템공학과 김유탄
20181112474 산업시스템공학과 복유연

목차

1. 타임 라인

I. 기존 계획

완료된 사항

- ✓ 그림자 오류 개선 / 블록 이동 간편화 / 콤보 기능
- ✓ Level에 따른 속도 변화 / 기존 스킬 제거 / 메뉴 기능 추가
- ✓ AI경쟁 모드 / 미니 모드 / Two hands 모드

진행중

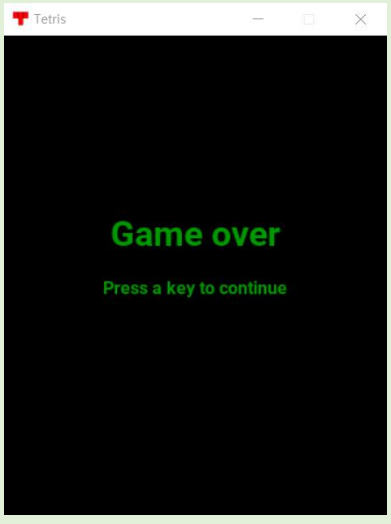
- ✓ 사운드 추가하기 / 디자인 개선

II. 추가 사항

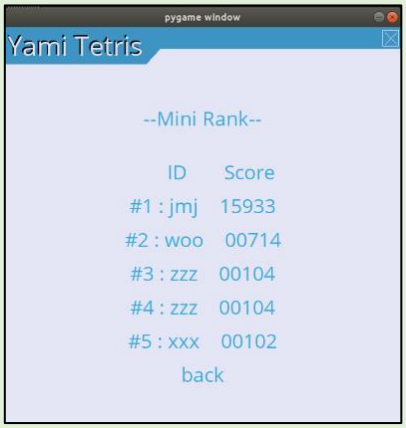
- ✓ 서버를 통한 랭킹 시스템 도입 (완료)
  - 기존의 local text 파일에 저장되어 있는 점수를 불러오는 방식이 아닌 서버에 점수를 기록하고 다른 컴퓨터에서 게임을 진행한 사람들의 점수까지 포함한 랭킹 시스템 구축하기
- ✓ 화면크기 조절 (진행중)
  - 화면의 크기를 조절할 수 없었던 게임을 화면 조절이 가능하도록 바꾸기

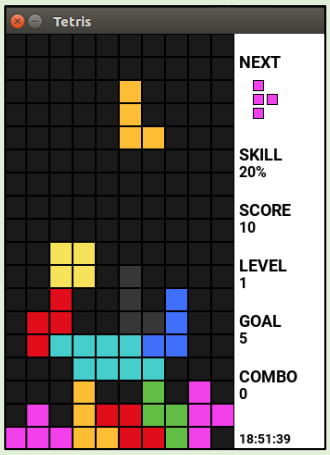
➔ 화면의 크기를 늘리면 그에 맞춰 게임의 화면도 함께 커지도록 하기

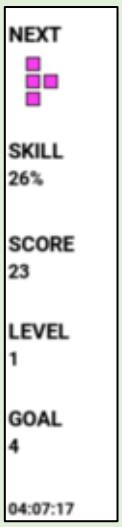
2. 진행 사항

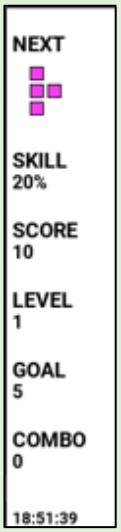
항목	변경 전	변경 후
인터페이스 개선	✓ 메뉴 없이 바로 게임 시작 	✓ 시작 시 게임 선택 및 랭크 보기 개선 - Pygame_menu 모듈을 활용한 메뉴 만들기 - 구성할 모드에 맞춰서 메뉴 구성   
	✓ 게임 종료 후 다시 게임 시작 	✓ 게임 종료 후 아이디 기록 

	✓ 랭크 보기 X	✓ 랭크 확인 메뉴 구현
	✓ 그림자 오류 발생	✓ 그림자 오류 개선
	✓ 콤보 X	✓ 콤보 화면에 구현





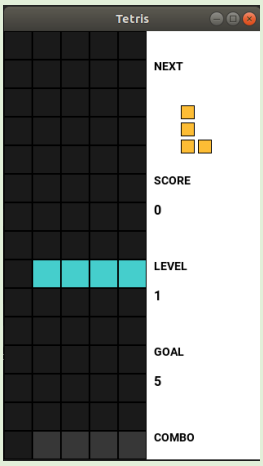




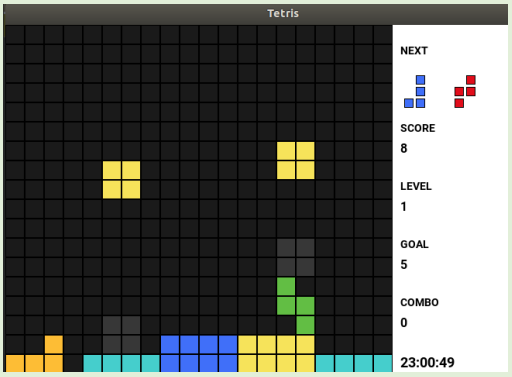
게임모드구현

✓ 기존 테트리스 모드

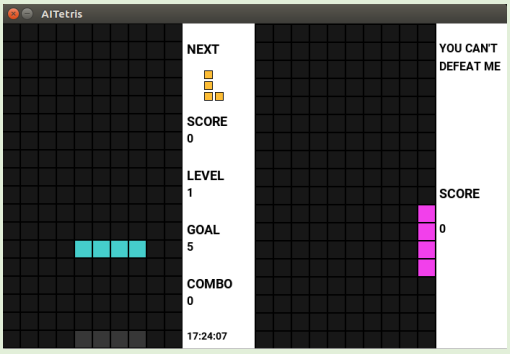
✓ 미니 게임 모드
- 가로 5칸 세로 15칸의 미니모드



✓ 투핸즈 모드
- 양손으로 플레이 가능한 모드



✓ AI 모드
- 유전알고리즘을 활용해 학습된 weight을 통해서 자동으로 진행되는 테트리스 구현
- User의 level이 올라가면, user와 AI의 속도 증가



기능 구현

✓ 콤보 기능 X

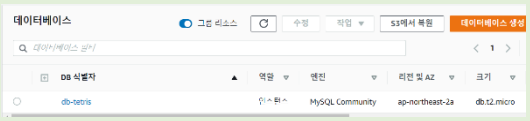
✓ 한줄 삭제 후 10초 간 콤보 추가 없을 시 콤보 초기화 기능 추가

```
def combo_null_start(self):
    for i in range(self.combo_max):
        if self.combo==i:
            self.timer_list[i]=threading.Timer(10, self.combo_null)
            self.timer_list[i].start()
            for j in range(self.combo_max):
                if i != j:
                    self.timer_list[j].cancel()
```

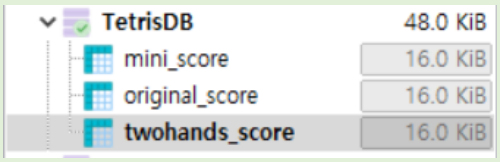
✓ 랭크 데이터 X

✓ 랭크 데이터 만들기

- Amazon RDS에서 MySQL데이터 서버 만들기



- 파이썬 내에서 데이터 서버 불러오기 및 데이터 서버 업데이트가 가능 하도록 구현



✓ 블록 연속 이동 X

✓ 블록 이동 간편화

- 연속 키 입력으로 구현

```
pygame.key.set_repeat(self.delay, self.interval)
```

✓ 그림자 오류 개선

```
def draw_shadow(self, array2d, dx, dy):  # 그림자 오류 디버깅     #########
    for y, row in enumerate(array2d):
        y += dy
        if y >= 0 and y < self.height:
            for x, block in enumerate(row):
                x += dx
                if block:
                    tmp = 1
                    while self.can_move_piece(0, tmp):
                        tmp += 1
                    x_s, y_s = self.pos_to_pixel(x, y + tmp - 1)
                    pygame.draw.rect(self.screen, Deep_gray,
                                     (x_s, y_s, self.block_size, self.block_size))
                    pygame.draw.rect(self.screen, BLACK,
                                     (x_s, y_s, self.block_size, self.block_size), 1)
```

✓ 난이도별 속도 조절

```
    # 추가 - 레벨별 스피드 조절
def level_speed(self):
    if self.level <= 9:
        pygame.time.set_timer(pygame.USEREVENT, (self.user_start_speed - (self.user_per_speed * self.level))
        pygame.time.set_timer(pygame.USEREVENT + 1, (self.AI_start_speed - (self.AI_speed * self.level))
    elif self.level >= 10:
        pygame.time.set_timer(pygame.USEREVENT, (self.user_start_speed - (self.user_per_speed * self.level))
        pygame.time.set_timer(pygame.USEREVENT + 1, (self.AI_start_speed - (self.AI_speed * self.level))
```

✓ 기존 스킬 제거

3. 중간 평가

I. 문제점

- ✓ 게임 재시작 시에 게임이 자동으로 종료되는 현상이 가끔 발견됨
- ✓ 게임 창 크기 조절 시에 정수인 변수만 사용 가능한 코드들이 존재하여 문제 발생
- ✓ 전체적인 디자인이 사용자의 흥미를 돋구기에 부족함
- ✓ Github에서 각자 fork한 저장소에서 작업을 진행하고, remote의 본래 저장소에 합치는 과정에서 생기는 conflict를 해결하는 능력이 미숙함
- ✓ Github의 본래 저장소와 fork한 저장소의 동기화 하는 과정에서 또한 conflict가 자주 발생
- ✓ 비슷한 코드를 활용해서 모드를 만드는 과정에서 중복되는 코드가 많음
- ✓ 사운드 저작권 문제로 리눅스 재생 시 오류가 발생

II. 추후 개선 사항

- ✓ 게임 진행 시 생기는 전체적인 오류 제거
- ✓ 화면 크기 조절
- ✓ 전체적인 인터페이스 개선
- ✓ 각 상황에 맞는 사운드 개선 및 리눅스 오류 개선
- ✓ 중복되는 코드 정리
- ✓ 코드 내에 들어가는 숫자 정의 해주기