
Python

TETRIS 오소오세요 팀

최종 발표

조장 | 통계학과 2017110514 이일영 |
조원 | 철학과 2014113236 지정원 |
조원 | 통계학과 2017110496 오희정 |

<목차>

1

기존 프로젝트 및 개선 방향

2

기능별 코드 분석

3

문제해결과정

4

결과물 및 향후과제

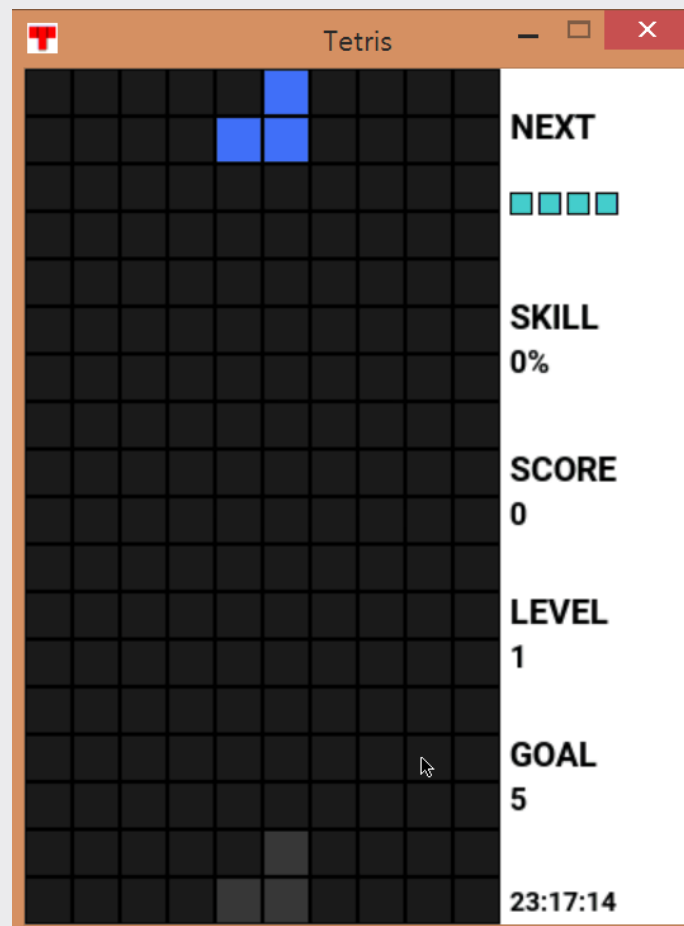


기존 프로젝트 및 개선 방향

▶ alchon/OSD_game
(https://github.com/alchon/OSD_game)

▷ Board.py(331 lines)
▷ Tetris.py(108 lines)
▷ Piece.py(67 lines) 총 506라인

▶ GNU General Public License v3.0



기존 프로젝트

- ▶ 단순한 스킬
블럭 1개 Skill 2점 up, 100점에 스킬 기능
- ▶ 레벨 구조
Goal만큼 라인 없애면 레벨 업, 속도 상승
- ▶ 일반모드
Gameover 하지 않는 이상 계속 플레이



흥미 유발



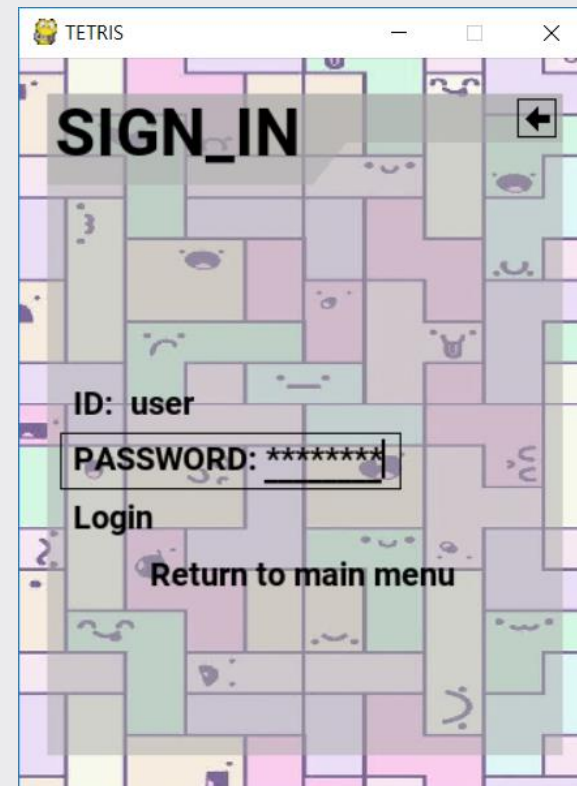
개선 항목

- ▶ 단순한 스킬 삭제
Skill, goal 삭제
- ▶ 흥미유발 요소
아이템과 콤보 점수 기능
- ▶ 일반모드 → 타임어택화
- ▶ UI 구성
계정생성 및 로그인, 메인 메뉴

1

기존 프로젝트 및 개선 방향

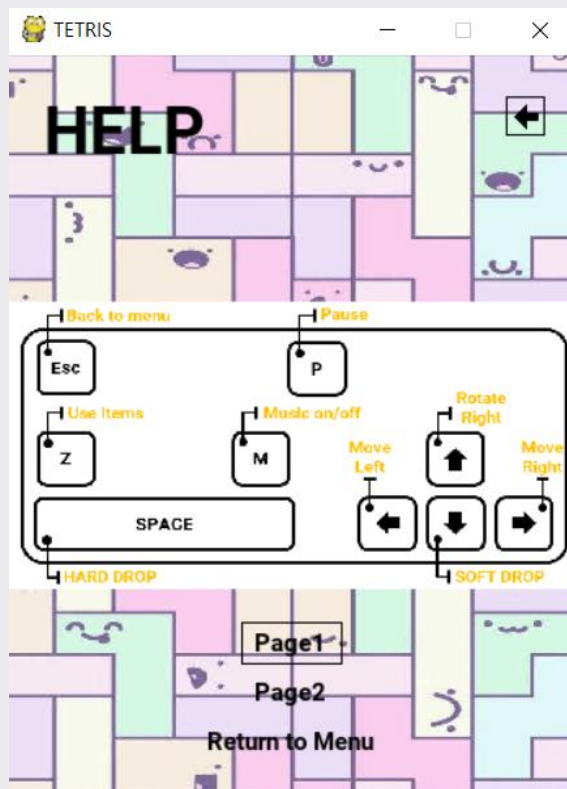
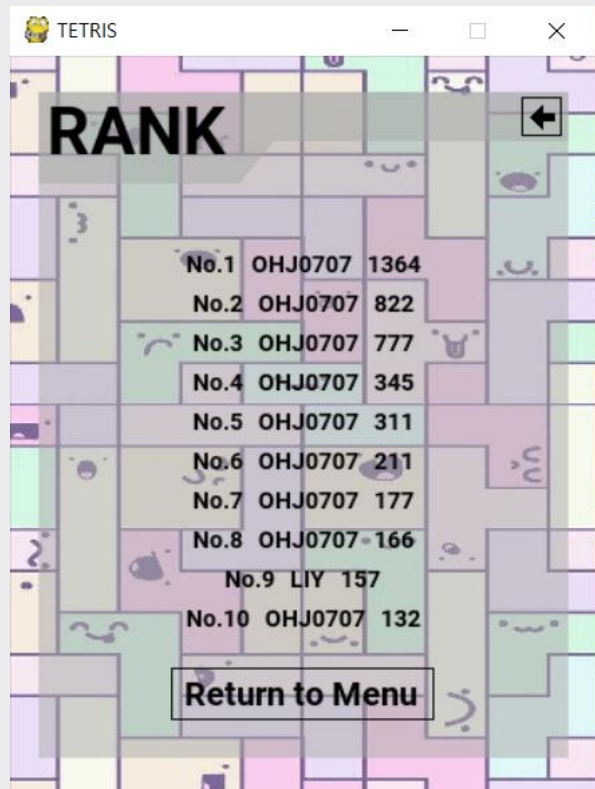
- 메뉴 구성



1

기존 프로젝트 및 개선 방향

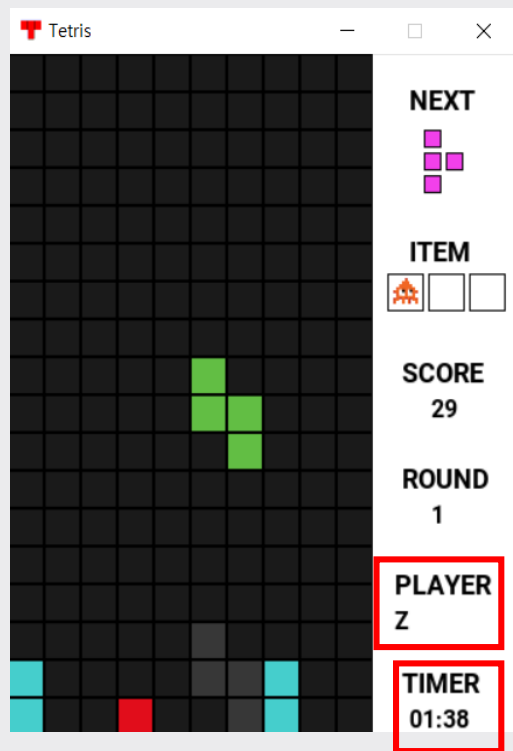
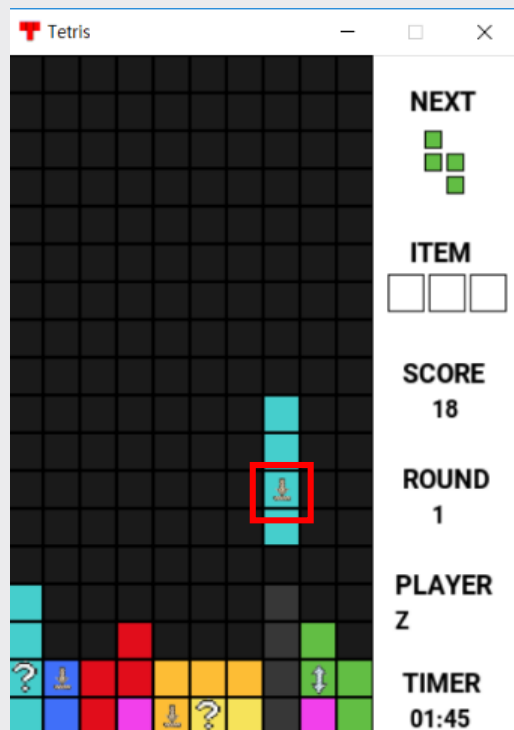
- 메뉴 구성



1

기존 프로젝트 및 개선 방향

- 아이템&타임어택화



pygameMenu 라이브러리 소개

오픈소스 주소 :

<https://github.com/ppizarror/pygame-menu>

Library structure

Module	Description
<i>pygameMenu.config</i>	Default configuration of Menus
<i>pygameMenu.controls</i>	Control definition, constants, etc.
<i>pygameMenu.events</i>	Events definition, constants, etc.
<i>pygameMenu.font</i>	Menu font management
<i>pygameMenu.locals</i>	Menu constants
<i>pygameMenu.Menu</i>	Menu class
<i>pygameMenu.sound</i>	Sound management
<i>pygameMenu.TextMenu</i>	TextMenu class
<i>pygameMenu.version</i>	Version of the library

Parameters are the following:

Param	Description	Type	Default
surface	Pygame surface object	Pygame Surface	-
window_width	Window width size (px)	int	-
window_height	Window height size (px)	int	-
font	Font file dir	str	-
title	Title of the menu (main title)	str	-
back_box	Draw a back-box button on header	bool	True
bgfun	Background drawing function (only if menupause app)	function	None
color_selected	Color of selected item	tuple	MENU_SELECTEDCOLOR
dopause	Pause game	bool	True
draw_region_x	Drawing position of element inside menu (x-axis) as percentage	int	MENU_DRAW_X
draw_region_y	Drawing position of element inside menu (y-axis) as percentage	int	MENU_DRAW_Y
draw_select	Draw a rectangle around selected item (bool)	bool	MENU_SELECTED_DRAW
enabled	Menu is enabled by default or not	bool	True

```
pygameMenu.Menu(surface, window_width, window_height, font, title, *args) # -> Menu object
```

```
pygameMenu.TextMenu(surface, window_width, window_height, font, title, *args) # -> TextMenu object
```

```
def fun():
    pass
```

```
help_menu = pygameMenu.TextMenu(surface, window...)
help_menu.add_option('Simple button', fun, align=pygameMenu.locals.ALIGN_LEFT)
help_menu.add_option('Return to Menu', pygameMenu.events.MENU_BACK)
```


2

기능별 코드 분석

- 메인 메뉴에 sign_up / sign_in 추가

```
# signin_menu
signin_menu = pygameMenu.Menu(surface,
                                bgfun=main_background,
                                color_selected=COLOR_WHITE,
                                font=pygameMenu.font.FONT_BEBAS,
                                font_color=COLOR_BLACK,
```

```
# Add text inputs with different configurations
signin_menu.add_text_input('ID: ',
                            default=' ',
                            onreturn=check_name_test,
                            textinput_id='ID')
signin_menu.add_text_input('PASSWORD: ',
                            password=True,
                            maxchar=8,
                            textinput_id='PASSWORD',
                            input_underline='_')
```

라이브러리의
add_text_input,
add_option 사용

```
def data2_func():
    data = signin_menu.get_input_data() # UI상에 입력된 정보 받아서 data에 저장
    input_id = (data['ID'].strip()).upper()
    input_pw = (data['PASSWORD'].strip()).upper()
    f = open("assets/account.txt", "r") # 가입한 계정이 저장되는 위치
    r = f.read()
    l = r.split()
    f.close()
    id_idx = 0
    login_con = 0
    for id_idx in range(len(l)): # 아이디 인덱스가 리스트 길이보다 짧으면
        if input_id != l[id_idx]: # 리스트 아이디와 아이디인덱스 value 비교
            id_idx += 3
        elif input_id == l[id_idx]: # 아이디 있으면 비밀번호 입력
            if login_con == 0:
                if input_pw == l[id_idx + 1]:
                    login_con = 1
                    print("signed in")
                    Tetris().run(input_id)
                    break
            else:
                print("비밀번호가 틀렸습니다")
                signin_menu.full_reset() ## 아직 안돼서 그냥 이전 메뉴로 돌아가게
                break
```

```
signin_menu.add_option('Login', data2_func) # Call function
signin_menu.add_option('Return to main menu', pygameMenu.events.BACK, align=pygameMenu.locals.ALIGN_CENTER)
```



2

기능별 코드 분석

- 메인 메뉴에 sign_up / sign_in 추가

파일 입출력 을 이용해

txt 파일로 계정 및 로그인 관리

```
def data_func():

    print('signup data:')
    f = open("assets/account.txt", "a")
    data = signup_menu.get_input_data()
```

```
if (data['ID'].strip()).upper() in open("assets/account.txt").read():
    print("중복되는 아이디가 있습니다.")
elif (len(data['ID'].strip().split()) > 1):
    print("띄어쓰기는 허용되지 않습니다.")
else:
    new_user_id = (data['ID'].strip()).upper()
    new_user_pw = (data['PASSWORD'].strip()).upper()
    new_user_sc = "0" # score 점수 0으로 초기화
    f.write(new_user_id + " " + new_user_pw + " " + new_user_sc + "\n")
f.close()
```



```
def data2_func():

    data = signin_menu.get_input_data() # UI상에 입력된 정보 받아서 data에 저장
    input_id = (data['ID'].strip()).upper()
    input_pw = (data['PASSWORD'].strip()).upper()
    f = open("assets/account.txt", "r") # 가입한 계정이 저장되는 위치
    r = f.read()
    l = r.split()
    f.close()
```

```
id_idx = 0
login_con = 0
for id_idx in range(len(l)): # 아이디 인덱스가 리스트 길이보다 짧으면
    if input_id != l[id_idx]: # 리스트 아이디와 아이디인덱스 value 비교
        id_idx += 3
    elif input_id == l[id_idx]: # 아이디 있으면 비밀번호 입력
        if login_con == 0:
            if input_pw == l[id_idx + 1]:
                login_con = 1
                print("signed in")
                Tetris().run(input_id)
                break
            else:
                print("비밀번호가 틀렸습니다")
                signin_menu.full_reset()
                break
```

2

기능별 코드 분석

- 메인 메뉴에 Top10 랭킹 띄우기

기존 파일에
(input_id, score) 튜플 추가

`pickle.dump()`

highscores.txt 파일

`pickle.load()`

ranks = []에 불러옴
랭크 메뉴에 텍스트 띄움

```
def HighScore(self, input_id):
    high_scores = []
    with open('assets/highscores.txt', 'rb') as f:
        high_scores = pickle.load(f)

    high_scores.append((str(input_id), self.board.score))
    high_scores = sorted(high_scores, key=itemgetter(1), reverse=True)[:10]
    with open('assets/highscores.txt', 'wb') as f:
        pickle.dump(high_scores, f)
```

```
ranks=[]
with open('assets/highscores.txt', 'rb') as f:
    r = pickle.load(f)
    for i, (user_id, score) in enumerate(r):
        ranks.append('No.' + str(i+1) + ' ' + str(user_id) + ' ' + str(score))

rank_menu = pygameMenu.TextMenu(surface,
```

```
for line in ranks:
    rank_menu.add_line(line) # Add line
rank_menu.add_option('Return to Menu', pygameMenu.events.BACK,
```

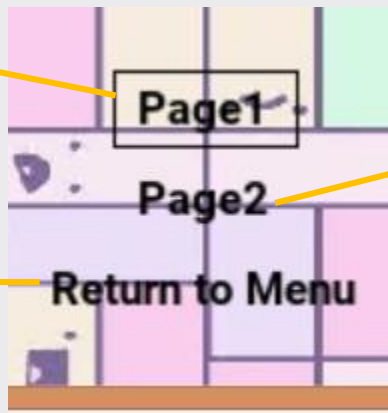
2

기능별 코드 분석

- 메인 메뉴 help 이미지 띄우기

```
def show_help():
    global help
    help=1
```

```
def close_help():
    global help
    help=0
    help_menu.full_reset()
```



```
def show_help2():
    global help
    help=2
```

help 전역변수의 값에 따라

surface에 띄울 이미지들을 다르게 함

help = 0 → 기본 배경

help = 1 → 그 위에 설명문1

help = 2 → 그 위에 설명문2

```
def main_background():
    global surface
    global help

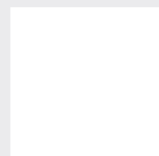
    if help==0:
        surface.fill((0, 0, 0))
        surface.blit(home_bg, (0,0))
    elif help==1:
        surface.blit(home_bg, (0,0))
        surface.blit(help_bg, (0,150))
    elif help==2:
        surface.blit(home_bg, (0,0))
        surface.blit(help2_bg, (30,80))
```

2

기능별 코드 분석

- 아이템 피스 생성

```
class Piece: #아이템 피스를 생성
    0 = (((0,0,0,0,0), (0,0,0,0,0), (0,0,1,1,0), (0,0,1,1,0), (0,0,0,0,0)),) * 4 #
    02 = (((0,0,0,0,0), (0,0,0,0,0), (0,0,8,1,0), (0,0,1,1,0), (0,0,0,0,0)),) * 4 #
    03 = (((0,0,0,0,0), (0,0,0,0,0), (0,0,15,1,0), (0,0,1,1,0), (0,0,0,0,0)),) * 4
    04 = (((0,0,0,0,0), (0,0,0,0,0), (0,0,22,1,0), (0,0,1,1,0), (0,0,0,0,0)),) * 4
```



1~7



8~14

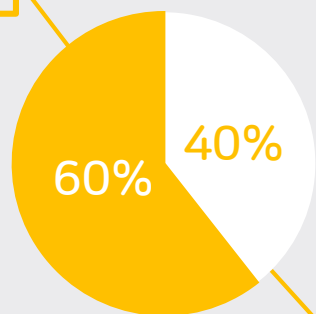


15~21



22~28

일반 피스



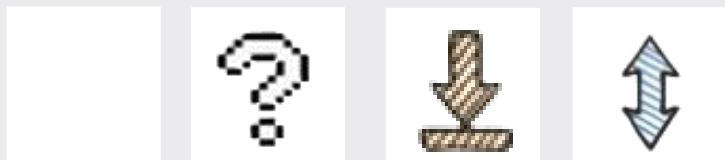
아이템 피스

```
def __init__(self, piece_name=None):
    self.rotation = 0
    if piece_name:
        self.piece_name = piece_name
    else:
        a = random.randint(1,5)
        if a < 3:
            self.piece_name = random.choice(list(Piece.PIECES2.keys()))
            self.array2d = Piece.PIECES2[self.piece_name][self.rotation]
        else:
            self.piece_name = random.choice(list(Piece.PIECES.keys()))
            self.array2d = Piece.PIECES[self.piece_name][self.rotation]
```

2

기능별 코드 분석

- 보드에 아이템 피스 그리기

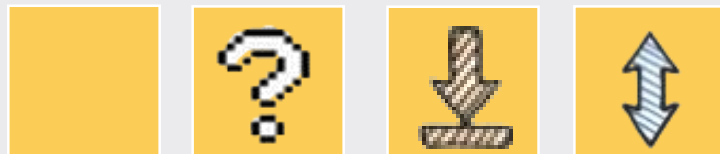


block : 1~7 8~14 15~21 22~28

def col_num(self, block)

num : 1 8 15 22

def draw_blocks(self, ..)



```
def col_num(self, block):    #블록
    if block<8 and block:
        return 1
    elif block>7 and block<15:
        return 8
    elif block>14 and block<22:
        return 15
    elif block>21 :
        return 22
```

```
def draw_blocks(self, array2d, color=WHITE, dx=0, dy=0):    #조건
    for y, row in enumerate(array2d):
        y += dy
        if y >= 2 and y < self.height:
            for x, block in enumerate(row):
                if block:
                    x += dx
                    x_pix, y_pix = self.pos_to_pixel(x, y)
                    num = self.col_num(block)
                    pygame.draw.rect(self.screen, self.piece.T_OO
                                     (x_pix, y_pix, self.b
                    pygame.draw.rect(self.screen, BLACK,
                                     (x_pix, y_pix, self.b

                    if num==8:
                        self.screen.blit(question,(x_pix,y_pix))
                    elif num==15:
                        self.screen.blit(delete,(x_pix,y_pix))
                    elif num==22:
                        self.screen.blit(updown,(x_pix,y_pix))
```

3

기능별 코드 분석

- draw_blocks 그림자 오류 디버깅

def draw_blocks

그림자 그리기

블록 그리기

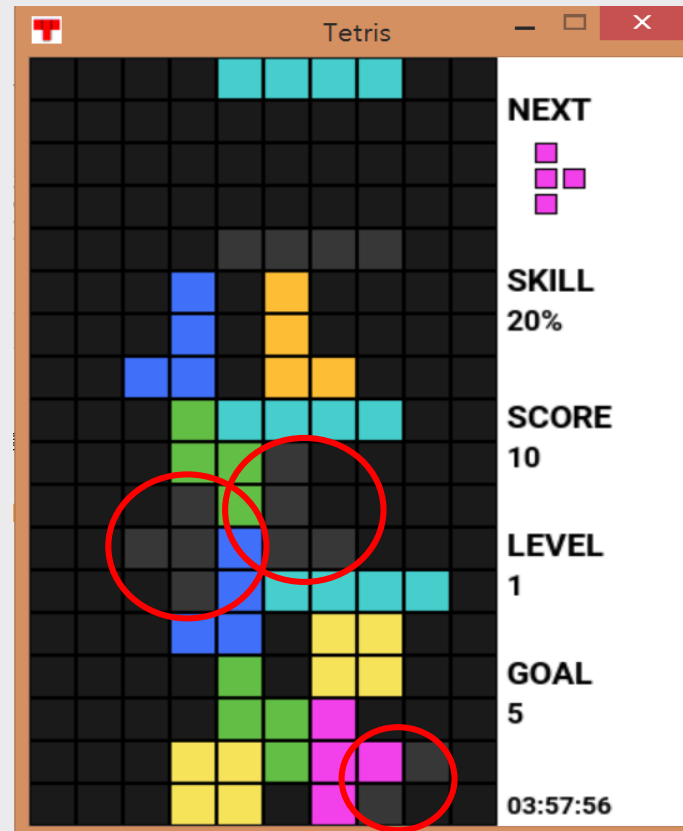


def draw_blocks

블록 그리기

def draw_shadow



그림자 그리기



2

기능별 코드 분석

- 보드에 인벤토리 생성, 추가

item_list = [ ,  ,  , ]

초기 상태 inven= []

랜덤

get_item() : inven= [아이템 이미지]

use_item() 이후 inven= []

각자 기능

show_item() : 실시간으로 인벤토리의
아이템을 화면에 띄움

```
def get_item(self):      #인벤토리에 아이템 생성
    if len(inven)<3:
        inven.append(item_list[random.randrange(0,4)])
    return inven
```

```
def use_item(self):      #인벤토리의 아이템 사용
    if len(inven)>0:
        item=inven[0]
        inven.pop(0)
        if item==item_list[0]:
            self.slow()
            t=threading.Timer(3,self.back_to_ori
            t.start()
```

```
def show_item(self):      #인벤토리의 아이템들을 보여줌
    if len(inven)>=1:
        self.screen.blit(inven[0] ,(260,145))
    if len(inven)>=2:
        self.screen.blit(inven[1] ,(288,145))
    if len(inven)==3:
        self.screen.blit(inven[2] ,(316,145))
```


Board.py

```
def back_to_origin(self): # 원래 속도로 돌아옴
    if self.round<=9:
        pygame.time.set_timer(pygame.USEREVENT,(500 - 50 * (self.round-1)))
    else :
        pygame.time.set_timer(pygame.USEREVENT,100)
```

```
def slow(self): # 달팽이 아이템 기능
    pygame.time.set_timer(pygame.USEREVENT,1000)
```

```
def fast(self): # 번개 아이템 기능
    pygame.time.set_timer(pygame.USEREVENT,100)
```

```
def change(self): # 피스 바꾸는 아이템 기능
    self.next_piece=Piece()
```

```
def squid_ink(self,a,b): # 오징어 먹물 아이템 기능
    ink_sound=pygame.mixer.Sound('assets/sounds/squid_ink.wav')
    ink_sound.play()
    for i in range(3000):
        ink=self.screen.blit(squid_ink,(a, b))
        if self.minutes==0 and self.seconds==0 :
            break
```



slow() : 피스 속도 감소



fast() : 피스 속도 증가

back_to_origin() : 피스 속도 원래대로



change() : 피스 바꾸기



squid_ink() : 오징어 먹물



use_item() 에서 **threading** 모듈 활용

2

기능별 코드 분석

- 아이템 기능(즉시 적용)

Board.py

```
def delete_under(self): # 맨 밑줄 아이템  
    self.delete_line(19)  
  
def delete_vertical(self,x): # 세로 아이템  
    for i in range(len(self.board)):  
        self.board[i][x]=0
```



delete_under() : 맨 밑줄 제거



delete_vertical() : 세로로 제거



delete_lines()에서 아이템 블록 있을 때 수행

```
delete_line()  
nextpiece()  
pygame.time.set_timer()  
...
```



기존 프로젝트의 함수 최대한 활용

2

기능별 코드 분석

- 다음 라운드

Board.py

```
self.board = [] # 보드 초기화
del inven[:]
self.round+=1

if self.round<=9:
    pygame.time.set_timer(pygame.USEREVENT, (500 - 50 * (self.round-1)))
else :
    pygame.time.set_timer(pygame.USEREVENT, 100)

for _ in range(self.height):
    self.board.append([0]*self.width)

pygame.display.update()

running = True
while running:
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            sys.exit()
        elif event.type == KEYUP and event.key == K_a:
            running = False
```

화면 이미지

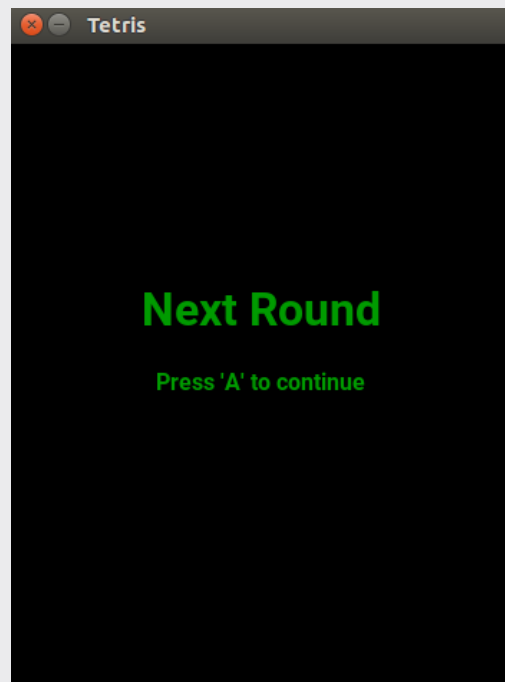
이전 라운드 초기화

보드, 인벤토리, 시간

속도 증가

a키 누르면 다음 라운드

▷ 다음 라운드 화면



2

기능별 코드 분석

- All clear

Board.py

```
del inven[:] #인벤토리와 속도 리셋되도록 설정
pygame.time.set_timer(pygame.USEREVENT, 500)
pygame.display.update()

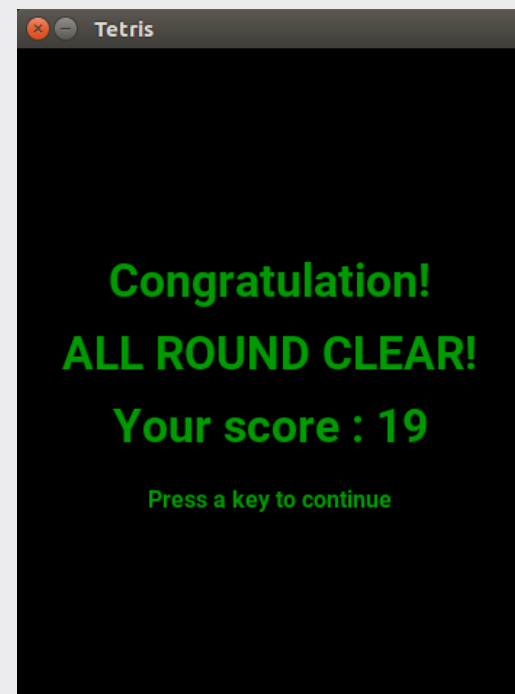
running = True
while running:
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            sys.exit()
        elif event.type == KEYDOWN:
            running = False
```

화면 이미지

이전 라운드 초기화

속도, 인벤토리, 시간

▷ all clear 화면



Board.py

```
def timer(self):
    self.total_seconds=self.start_time-(self.frame_count//30)
    if self.total_seconds<0:
        self.total_seconds=0

    self.minutes=self.total_seconds//60
    self.seconds=self.total_seconds%60

    output='{0:02}:{1:02}'.format(self.minutes,self.seconds)
    time_value=pygame.font.Font('assets/Roboto-Bold.ttf', 16).render(output, True, BLACK)
    self.screen.blit(time_value,(275,430))
    self.frame_count+=1
```

- ▶ 플레이 시간 2분
- ▶ Tetris.py의 run()의 while문
- ▶ minutes & seconds=0
--→ 다음 라운드 or 게임 종료



Tetris.py의 run()

```
self.board.timer()
```

```
if self.board.minutes==0 and self.board.seconds==0:
    if self.board.round!=10:
        self.board.next_round()
        self.board.init()
    else:
        self.screen.fill(BLACK)
        pygame.mixer.music.stop()
        self.board.all_clear()
        self.HighScore(input_id)
        self.check_reset = True
        self.board.init_board()
```



- ▶ 오징어 먹물 이미지
오징어 먹물 이미지 뜨면 피스 정지
- ▶ 맨 밑 줄 아이템
아이템 실행 시, 보드의 블록이 전부 지워짐
- ▶ 세로 아이템
아이템 실행 시, 동시에 채워진 아래쪽 줄 안 지워짐
- ▶ help 이미지 띄우기
지정된 메인 배경을 다른 배경으로 바꾸기 어려움



- ▶ 먹물 이미지 띄우는 함수에 따로 Thread를 부여
- ▶ 맨 밑 줄 아이템 블록이 있으면 바로 함수를 실행하는 것이 아니라 라인을 먼저 없앤 뒤에 수행하도록 함
- ▶ 라인을 먼저 지운 뒤에 세로 아이템을 사용하도록 loop의 바깥쪽으로 뺌
- ▶ 전역변수를 이용해 메인 배경에 추가로 띄움

▶ CSID-DGU/2019-2-OSSPC-OSO_OSEYO-5
(https://github.com/CSID-DGU/2019-2-OSSPC-OSO_OSEYO-5)

- ▷ Board.py(331 lines) -> (516 lines)
- ▷ Tetris.py(108 lines) -> (416 lines)
- ▷ Piece.py(67 lines) -> (170 lines)

총 1102라인





- ▶ 비행기 방해 아이템 구현 x
- ▶ 계정 생성 or 로그인 시, 계정이 저장되거나 오류가 났을 때 메시지 못 띄움
- ▶ 오징어 먹물 이미지 flickering
- ▶ 멀티 플레이

Python

감사합니다!

Q&A

오 소 오 세 요 팀
