tetremino 보고서

2021092924 박종인

A : 1~6번의 과제를 코드의 어느 부분을 어떻게 수정했는지 설명

1. 현재 테트리스 게임의 배경음악을 주어진 3개의 음악 중 1개가 재생되도록 수정

수정된 코드:

```
def main():
    global FPSCLOCK, DISPLAYSURF, BASICFONT, BIGFONT
    pygame.init()
    FPSCLOCK = pygame.time.Clock()
    DISPLAYSURF = pygame.display.set_mode((WINDOWWIDTH, WINDOWHEIGHT))
    BASICFONT = pygame.font.Font('freesansbold.ttf', 18)
    BIGFONT = pygame.font.Font('freesansbold.ttf', 100)
    pygame.display.set_caption('2021092924_PARKJONGIN')

    showTextScreen('MY TETRIS')
    while True: # game loop
        music_choice = random.randint(0, 2)
        if music_choice == 0:
            pygame.mixer.music.load('Hover.mp3')
        elif music_choice == 1:
            pygame.mixer.music.load('Our_Lives_Past.mp3')
        else:
            pygame.mixer.music.load('Platform_9.mp3')
        pygame.mixer.music.play(-1, 0.0)
        runGame()
        pygame.mixer.music.stop()
        showTextScreen('Over :(')
```

pygame.mixer.music.load()와 pygame.mixer.music.play()를 사용하여 세 가지 음악 중 하나를 무작위로 선택하여 재생하도록 추가함.

2. 상태창 이름을 학번_이름으로 수정

수정된 코드:

```
pygame.display.set_caption('2021092924_PARKJONGIN')
```

pygame.display.set_caption('2021092924_PARKJONGIN')을 사용하여 상태창 이름을 설정함.

3. 게임시작화면의 문구를 MY TETRIS으로 변경

수정된 코드:
showTextScreen('MY TETRIS')

showTextScreen('MY TETRIS')을 사용하여 게임 시작화면의 문구를 설정함.

4. 게임시작화면의 문구 및 배경색을 노란색으로 변경

수정된 코드:
```
def showTextScreen(text):
    # This function displays large text in the
    # center of the screen until a key is pressed.
    # Draw the text drop shadow
    titleSurf, titleRect = makeTextObjs(text, BIGFONT, TEXTSHADOWCOLOR)
    titleRect.center = (int(WINDOWWIDTH / 2), int(WINDOWHEIGHT / 2))
    DISPLAYSURF.blit(titleSurf, titleRect)

    # Draw the text
    titleSurf, titleRect = makeTextObjs(text, BIGFONT, TEXTCOLOR)
    titleRect.center = (int(WINDOWWIDTH / 2) - 3, int(WINDOWHEIGHT / 2) - 3)
    DISPLAYSURF.blit(titleSurf, titleRect)

    # Draw the additional "Press a key to play." text.
    pressKeySurf, pressKeyRect = makeTextObjs('Press any key to play! pause key is p',
BASICFONT, TEXTCOLOR)
    pressKeyRect.center = (int(WINDOWWIDTH / 2), int(WINDOWHEIGHT / 2) + 100)
    DISPLAYSURF.blit(pressKeySurf, pressKeyRect)

    while checkForKeyPress() == None:
        pygame.display.update()
        FPSCLOCK.tick()
```

게임 시작화면의 문구를 변경하고 텍스트 색상을 노란색으로 설정함.

5. 게임 경과 시간을 초 단위로 표시 (새 게임 시작 시 0으로 초기화되어야 함)

수정된 코드:
```
def runGame():
    startTime = time.time()
    ...
    while True: # game loop
        elapsedTime = time.time() - startTime
        ...
        drawElapsedTime(elapsedTime)
        pygame.display.update()
```

```
        FPSCLOCK.tick(FPS)

def drawElapsedTime(elapsedTime):
    elapsedTimeSurf = BASICFONT.render('Play Time: %s sec' % int(elapsedTime), True, YELLOW)
    elapsedTimeRect = elapsedTimeSurf.get_rect()
    elapsedTimeRect.topright = (WINDOWWIDTH - 470, 20)
    DISPLAYSURF.blit(elapsedTimeSurf, elapsedTimeRect)
```

runGame() 함수 내에 startTime 변수를 추가하고, elapsedTime을 계산하여 drawElapsedTime() 함수에서 화면에 경과 시간을 표시하도록 수정함.

6. 7개의 블록이 각각 고유의 색을 갖도록 코드를 수정하거나 추가

수정된 코드:

```
COLORS = {
    'S': GREEN,
    'Z': RED,
    'J': BLUE,
    'L': ORANGE,
    'I': CYAN,
    'O': YELLOW,
    'T': PURPLE
}

def getNewPiece():
    shape = random.choice(list(PIECES.keys()))
    newPiece = {'shape': shape,
                'rotation': random.randint(0, len(PIECES[shape]) - 1),
                'x': int(BOARDWIDTH / 2) - int(TEMPLATEWIDTH / 2),
                'y': -2,
                'color': COLORS[shape]}
    return newPiece
```

각 블록 모양에 고유의 색상을 부여함. getNewPiece() 함수에서 블록 모양에 맞는 색상을 설정하도록 수정함.

결론

주어진 테트리스 게임 코드의 6가지 과제를 해결하기 위해 어떤 부분을 수정했는지 설명했습니다. 수정된 코드를 통해 배경음악이 무작위로 재생되고, 상태창 이름과 게임 시작화면의 문구가 변경되었으며, 게임 경과 시간이 초 단위로 표시되고, 각 블록이 고유한 색상을 갖도록 했습니다.

B : 각 함수의 역할

1. main() 함수
설명: main() 함수는 게임의 시작점이자 전체 흐름을 제어하는 역할을 합니다. 게임 초기화, 이벤트 루프, 게임 종료 등을 관리합니다.

주요 역할: 게임의 초기 설정 및 변수 초기화
시작 화면 및 게임 종료 화면 표시를 위한 showTextScreen() 함수 호출
게임 루프를 통해 지속적으로 게임 상태를 업데이트

핵심 내용: 프로그램이 실행될 때 가장 먼저 호출되는 함수로, 게임의 전반적인 흐름을 관리합니다.
사용자의 입력을 기다리며, 게임의 각 단계를 차례로 진행합니다.

2. runGame() 함수
설명: runGame() 함수는 게임의 주요 루프를 담당하며, 게임 상태를 업데이트하고 화면을 그립니다. 블록의 이동, 충돌, 라인 삭제 등을 처리합니다.

주요 역할: 새로운 게임을 시작할 때 초기 설정 및 변수 초기화
게임 상태를 지속적으로 업데이트하여 게임 진행
사용자 입력을 처리하고 블록 이동, 회전, 충돌 등을 관리

핵심 내용: 게임 루프 내에서 게임 상태를 지속적으로 갱신하고, 게임 종료 조건을 확인합니다.
블록이 이동하거나 회전할 때 충돌을 감지하고, 완성된 라인을 삭제하여 점수를 업데이트합니다.

3. isValidPosition() 함수
설명: isValidPosition() 함수는 현재 블록이 게임 보드 내에서 유효한 위치에 있는지 확인합니다. 블록의 충돌 및 경계 체크를 수행하여 블록이 유효한 위치에 놓일 수 있는지 판단합니다.

주요 역할: 현재 블록이 게임 보드의 경계를 벗어나지 않도록 확인
블록이 다른 블록과 충돌하지 않도록 확인

핵심 내용: 블록의 위치와 회전을 기준으로 게임 보드 내에서 유효성을 판단합니다.
블록이 유효한 위치에 놓일 수 없으면 False를 반환하여, 충돌이나 경계 문제를 처리합니다.

결론

이 세 가지 함수는 게임의 흐름을 제어하고, 사용자 입력을 처리하며, 게임 상태를 유지하는 핵심 기능을 수행하게 됩니다.. main() 함수는 게임의 전체 흐름을 관리하고, runGame() 함수는 게임 플레이의 주요 루프를 처리하며, isValidPosition() 함수는 블록의 유효성을 판단하여 충돌을 방지합니다.

C : 함수의 호출 순서 및 호출 조건에 대한 설명

게임의 전체 흐름을 이해하기 위해 각 함수의 호출 순서 및 조건을 중심으로 정리하였습니다.
아래 내용은 전체 코드에서의 함수 호출 순서와 조건을 다룹니다.

게임의 전체 흐름
1. main() 함수
게임 실행의 시작점으로, 프로그램이 처음 시작될 때 호출함
게임의 기본 설정과 초기화 작업을 수행함.
무한 루프를 통해 게임이 실행되고, 게임 종료 조건이 만족될 때까지 반복됨.

2. showTextScreen() 함수
게임 시작 시 환영 메시지를 표시하거나 게임 종료 후 최종 점수를 표시하는 화면을 보여줌.
사용자가 키를 누르면 화면이 닫히고 게임이 시작되거나 다시 시작함.

3.startGame() 함수
새로운 게임을 시작할 때 호출함.
게임 변수 초기화, 블록 생성 등의 작업을 수행함.
게임 루프가 시작함.

4.gameLoop() 함수
게임의 주요 루프를 관리함.
게임 진행 중 지속적으로 호출되어 게임 상태를 업데이트함.
사용자의 키 입력을 처리하고 블록 이동, 회전, 충돌 감지 등의 작업을 수행함.
게임 종료 조건(예: 블록이 화면 상단에 닿을 때)을 확인하고, 종료되면 다음 단계를 진행함.

5.processInput() 함수
게임 루프 내에서 사용자의 키 입력을 처리함.
방향 키, 회전 키, 드롭 키 등의 입력을 받아 블록을 이동하거나 회전시킴.

6.updateGameState() 함수
게임 상태를 업데이트함.
블록이 아래로 이동하거나, 충돌 시 새로운 블록을 생성하고 라인을 삭제하는 작업을 수행함.

7.checkForFullLines() 함수
완성된 라인을 확인하고, 해당 라인을 삭제하여 점수를 업데이트함.
라인 삭제 후 블록을 아래로 내림.

8.drawGameScreen() 함수
게임 화면을 그림.
현재 블록, 다음 블록, 점수, 게임 보드 등을 화면에 표시함.
이 함수는 게임 루프 내에서 지속적으로 호출되어 화면을 갱신함.

9.endGame() 함수
게임 종료 시 호출됨.
최종 점수를 저장하고, 게임 종료 메시지를 화면에 표시함.
게임 재시작 여부를 확인하고, 재시작할 경우 startGame() 함수를 다시 호출하여 게임을 재시작함.

흐름 요약
main() 함수에서 프로그램 시작 및 초기화. -> showTextScreen() 함수로 시작 화면 표시. -> 사용자 입력에 따라 startGame() 함수 호출. -> gameLoop() 함수에서 게임 진행 루프 시작 ->
processInput() 함수로 사용자 입력 처리. -> updateGameState() 함수로 게임 상태 업데이트.
checkForFullLines() 함수로 라인 체크 및 삭제. -> drawGameScreen() 함수로 화면 갱신.->
게임 종료 조건 만족 시 endGame() 함수 호출. -> 게임 재시작 여부에 따라 startGame() 함수 재호출 또는 프로그램 종료.

결론
이와 같은 흐름을 통해 게임이 순차적으로 실행되고, 사용자의 입력에 따라 게임 상태가 업데이트되며, 화면에 그려집니다. 게임 종료 조건이 만족되면 종료 화면을 표시하고, 사용자가 원할 경우 게임이 다시 시작됩니다.

github 주소
https://github.com/pji0401/osw-repository