오픈소스SW 과제중심수업 보고서

경제학부

2019012633 염재원

GitHub repository 주소 : https://github.com/jaebong87/osw.git

1. 각 함수들의 역할

```
# Wormy (a Nibbles clone)
# By Al Sweigart al@inventwithpython.com
# http://inventwithpython.com/pygame
# Released under a "Simplified BSD" license

import random, pygame, sys
from pygame.locals import *

FPS = 15
WINDOWWIDTH = 640
WINDOWHEIGHT = 480
CELLSIZE = 20
assert WINDOWWIDTH % CELLSIZE == 0, "Window width must be a multiple of cell size."
assert WINDOWHEIGHT % CELLSIZE == 0, "Window height must be a multiple of cell size."
CELLWIDTH = int(WINDOWWIDTH / CELLSIZE)
CELLHEIGHT = int(WINDOWHEIGHT / CELLSIZE)
```

프로그램 시작부분에서 게임에 필요한 상수들을 설정.

셀의 넓이와 높이를 CELLSIZE에 저장.

assert 문을 써서 셀이 윈도우 안에 남는 영역 없이 들어가는지 확인. assert문으로 정수개의 셀이 윈도우에 들어가는지 확인해야함

```
#             R    G    B
WHITE     = (255, 255, 255)
BLACK     = (  0,   0,   0)
RED       = (255,   0,   0)
GREEN     = (  0, 255,   0)
DARKGREEN = (  0, 155,   0)
YELLOW    = (255, 255,   0)
DARKGRAY  = ( 40,  40,  40)

BGCOLOR = BLACK
TEXTCOLOR = WHITE

UP = 'up'
DOWN = 'down'
LEFT = 'left'
RIGHT = 'right'

HEAD = 0 # syntactic sugar: index of the worm's head
```

과제 중 게임 시작 화면 문구 파트를 노란색 계열로 바꾸라는 조건이 있었음 그래서 YELLOW를 추가하고 R,B,G를 노란색 색상에 맞게 설정값을 집어넣음

더 많은 상수는 19 ~ 32 행에 설정됨. HEAD 상수는 뒤에서 설명하고자함.

```
def main():
    global FPSCLOCK, DISPLAYSURF, BASICFONT, BIGFONT

    pygame.init()
    FPSCLOCK = pygame.time.Clock()
    DISPLAYSURF = pygame.display.set_mode((WINDOWWIDTH, WINDOWHEIGHT))
    BASICFONT = pygame.font.Font('D2Coding.ttc', 18)# 폰트변경
    BIGFONT = pygame.font.Font('D2Coding.ttc', 100)#폰트 변경
    pygame.display.set_caption('2019012633_yeomjaewon')##이름 변경

    showStartScreen()

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

Wormy 게임 프로그램에서는 코드의 주요 부분을 runGame() 이라는 함수에 넣음.
게임 상태창 변경위해 pygame.display.set_caption('2019012633_yeomjaewon') 로 정하였으며 폰트 역시 제공된 폰트 'D2Coding.tcc'로 모두 수정함
게임 시작 시 제공된 3개의 음악 중 1개가 랜덤으로 실행하라는 과제를 수행하기 위해 기존의 Tetromino 참고함. Tetromino의 경우 if/else 문이었기에 노래 3개를 넣어야 하므로 if, elif문을 사용함
음악 3곡이 랜덤으로 나와야 하므로 radint함수를 사용하여 3개의 노래가 랜덤으로 재생할 수 있게 수정함
게임을 시작하면 게임 시작 화면을 한 번만 보여주고 그 외의 게임 재실행시에는 게임 시작 화면이 뜨면 안됨 그래서 게임 루프부분을 runGame()함수로 뺌
게임 시작 화면을 보여주고 runGame()을 호출해 게임을 시작, 벌레가 벽에 부딪히거나 자기 자신에게 부딪힐 경우 게임 종료, runGame()은 main()으로 반환

```
def runGame():
    # Set a random start point.
    startx = random.randint(5, CELLWIDTH - 6)
    starty = random.randint(5, CELLHEIGHT - 6)
    wormCoords = [{'x': startx,     'y': starty},
                  {'x': startx - 1, 'y': starty},
                  {'x': startx - 2, 'y': starty}]
    direction = RIGHT

    # Start the apple in a random place.
    apple = getRandomLocation()
```

게임이 시작되면 벌레는 무작위로 정해진 위치에서 시작, 근데 게임판의 가장자리랑 너무 가까우면 안됨. 시작하는 무작위 위치의 좌표를 startx와 starty에 저장.
벌레의 몸은 dictionary값의 리스트로 저장됨. 벌레의 몸 하나하나마다 딕셔너리 값을 할당
딕셔너리는 몸 부분의 X,Y좌표에 대한 'x','y'의 키 값을 가짐. 벌레머리부분 좌표는 startx와 starty로 몸마디 다른부분 2개는 머리의 왼쪽에 놓이게 됨

```
while True: # main game loop
    for event in pygame.event.get(): # event handling loop
        if event.type == QUIT:
            terminate()
        elif event.type == KEYDOWN:
            if (event.key == K_p):
                # Pausing the game
                DISPLAYSURF.fill(BGCOLOR)
                pygame.mixer.music.stop()
                showTextScreen('Paused') # pause until a key press
                pygame.mixer.music.play(-1, 0.0)
            elif (event.key == K_LEFT or event.key == K_a) and direction != RIGHT:
                direction = LEFT
            elif (event.key == K_RIGHT or event.key == K_d) and direction != LEFT:
                direction = RIGHT
            elif (event.key == K_UP or event.key == K_w) and direction != DOWN:
                direction = UP
            elif (event.key == K_DOWN or event.key == K_s) and direction != UP:
                direction = DOWN
            elif event.key == K_ESCAPE:
                terminate()
```

while True에서 시작 부분, 그 밑줄은 이벤트 처리 시작부분임. QUIT이벤트가 발생하면 terminate()를 호출함.

만약 KEYDOWN 이벤트가 발생했으면 이 키가 어떠한 키인지 확인함. 화살표 키인지, WASD키인지 P키인지 확인. 만약 P키를 눌렀을 경우 배경이 바뀌고 음악이 멈추면서 showTextScreen 함수를 정의하고 사용하여 (후술 예정) 멈췄을 때 화면을 보여줌.

하지만 여기에 벌레가 자기에게 충돌하는지 검사해 보는 과정 필요. 예를 들어 벌레가 왼쪽으로 이동하고 있는데 플레이어가 오른쪽 화살표 키를 누르면 벌레는 갑자기 오른쪽으로 방향을 바꿔야 하고 그럼 자기 자신과 충돌함, 따라서 키를 누를 때는 direction 변수를 같이 검사해야 함. 플레이어가 실수로 자기자신과 충돌하는 키를 누르면 그 키는 무시함.

```
    # check if the worm has hit itself or the edge
    if wormCoords[HEAD]['x'] == -1 or wormCoords[HEAD]['x'] == CELLWIDTH
     or wormCoords[HEAD]['y'] == -1 or wormCoords[HEAD]['y'] == CELLHEIGHT:
        return # game over
    for wormBody in wormCoords[1:]:
        if wormBody['x'] == wormCoords[HEAD]['x'] and wormBody['y'] == wormCoords[HEAD]['y']:
            return # game over
```

벌레의 머리부분이 게임판의 테두리에 닿거나 머리부분이 몸의 다른 부분에 닿으면 충돌한것으로 되어 게임 종료됨.

게임이 종료되면 그냥 runGame()에서 main()으로 돌아가면 됨. main으로 돌아가면 runGame() 다음의 showGameOverScreen()함수를 호출하고 "Game Over"텍스트가 보임

마지막에는 return으로 처리함

```
    # check if worm has eaten an apply
    if wormCoords[HEAD]['x'] == apple['x'] and wormCoords[HEAD]['y'] == apple['y']:
        # don't remove worm's tail segment
        apple = getRandomLocation() # set a new apple somewhere
    else:
        del wormCoords[-1] # remove worm's tail segment
```

벌레의 머리 부분과 사과와도 위와 비슷한 충돌감지를 수행, 벌레의 머리 부분과 사과의 좌표가 일치하면 사과의 위치를 다시 무작위로 설정함. 사과의 새 위치는 getRandomLocation()함수로 결정함. 머리부분과 사과가 만나지 않았으면 몸의 마지막 마디를 지움.

```
# move the worm by adding a segment in the direction it is moving
if direction == UP:
    newHead = {'x': wormCoords[HEAD]['x'], 'y': wormCoords[HEAD]['y'] - 1}
elif direction == DOWN:
    newHead = {'x': wormCoords[HEAD]['x'], 'y': wormCoords[HEAD]['y'] + 1}
elif direction == LEFT:
    newHead = {'x': wormCoords[HEAD]['x'] - 1, 'y': wormCoords[HEAD]['y']}
elif direction == RIGHT:
    newHead = {'x': wormCoords[HEAD]['x'] + 1, 'y': wormCoords[HEAD]['y']}
wormCoords.insert(0, newHead)
```

벌레를 움직이기 위해 wormCoords리스트의 시작 부분에 새로운 마디를 하나씩 더해감. 리스트의 앞부분에 붙인 마디는 새로운 머리가 됨. 새로운 머리의 좌표는 이전 머리의 바로 옆 좌표가 됨. X좌표에 1을 더하거나 뺄것인지 혹은 Y좌표에 1을 더하거나 빼서 새로운 좌표를 만들 것인지 진행방향에 달려있음.
새 머리 부분은 후에 wormCoords 리스트에 insert() 리스트 메소드로 추가함.

```
    DISPLAYSURF.fill(BGCOLOR)
    drawGrid()
    drawWorm(wormCoords)
    drawApple(apple)
    drawScore(len(wormCoords) - 3)
    pygame.display.update()
     FPSCLOCK.tick(FPS)
```

배경색으로 전체 디스플레이 Surface를 한 번 채운다음 격자, 벌레, 사과, 점수를 디스플레이 Surface에 그리고 그다음 pygame.display.update()를 호출해 디스플레이 surface를 실제 화면상에 반영함.

```
def drawPressKeyMsg():
    pressKeySurf = BASICFONT.render('Press a key to play.', True, DARKGRAY)
    pressKeyRect = pressKeySurf.get_rect()
    pressKeyRect.topleft = (WINDOWWIDTH - 200, WINDOWHEIGHT - 30)
    DISPLAYSURF.blit(pressKeySurf, pressKeyRect)
```

시작 화면 애니메이션이 나올때나 게임 종료화면이 보일 때 오른쪽 아래구석에 항상 'Press a key to play' 라는 글씨가 보임, showStartScreen()과 showGameOverScreen()에서 그냥 이 함수 호출하면 됨

```
def checkForKeyPress():
    if len(pygame.event.get(QUIT)) > 0:
        terminate()

    keyUpEvents = pygame.event.get(KEYUP)
    if len(keyUpEvents) == 0:
        return None
    if keyUpEvents[0].key == K_ESCAPE:
        terminate()
    return keyUpEvents[0].key
```

이 함수는 우선 이벤트 큐를 검사해서 QUIT이벤트가 발생했는지 봄. pygame.event.get()는 이벤트 큐에 들어있는 모든 QUIT이벤트를 반환함. 만약 QUIT 이벤트가 않았으면 pygame.event.get()은 빈 리스트 []를 반환함.

pygame.event.get() 이 빈 리스트를 반환했으면 len()은 값이 0임. 이 값이 1이상의 수이면 QUIT이벤트가 발생한것이므로 terminate()함수를 호출하고 프로그램 종료함. QUIT이벤트가 발생한적 없으면 pygame.event.get()는 이벤트 큐에서 KEYUP이벤트 가져옴. 만약 Esc 키를 누르면 이역시 프로그램 종료. Esc 키도 누른 적이 없으면 pygame.event.get()에서 반환한 첫 번째 키 이벤트 객체를 반환.

```
def showStartScreen():
    titleFont = pygame.font.Font('D2Coding.ttc', 100) ##폰트 바꾸기
    titleSurf1 = titleFont.render('OSW Game', True, BLACK, YELLOW) ### 색변화, OSW g
    titleSurf2 = titleFont.render('OSW Game', True, YELLOW) ### 색변화, OSW game으로

    degrees1 = 0
    degrees2 = 0
    while True:
        DISPLAYSURF.fill(BGCOLOR)
```

게임 시작전 어떤 게임인지 알려주는 화면, 과제로 폰트바꾸기, 색 변환하기, 타이틀 바꾸기 주어져 이 부분에서 수정을 해 과제를 수행함

워미의 시작화면에 'OSW Game'이라는 글씨를 쓸 Surface객체 2개가 필요함, render을 호출했고 시작화면의 글씨를 크게 설정하기 위해 Font 생성함수에 Font 객체를 100으로 설정함. 첫번째 'OSW Game'은 검은색 바탕에 노란 글씨이고 두번째 'OSW Game'는 투명한 바탕에 노란글씨로 만듦.
While문은 넣어 애니메이션이 실행하는동안 이 글씨 2개가 회전하며서 디스플레이Surface 객체에 그려짐

```
        rotatedSurf1 = pygame.transform.rotate(titleSurf1, degrees1)
        rotatedRect1 = rotatedSurf1.get_rect()
        rotatedRect1.center = (WINDOWWIDTH / 2, WINDOWHEIGHT / 2)
        DISPLAYSURF.blit(rotatedSurf1, rotatedRect1)

        rotatedSurf2 = pygame.transform.rotate(titleSurf2, degrees2)
        rotatedRect2 = rotatedSurf2.get_rect()
        rotatedRect2.center = (WINDOWWIDTH / 2, WINDOWHEIGHT / 2)
        DISPLAYSURF.blit(rotatedSurf2, rotatedRect2)

        drawPressKeyMsg()

        if checkForKeyPress():
            pygame.event.get() # clear event queue
            return
        pygame.display.update()
        FPSCLOCK.tick(FPS)
```

showStartscreen()함수는 'OSW Game'적혀있는 Surface 객체상의 이미지를 회전함. 함수의 첫 번째 파라미터는 회전할 이미지의 복사본을 만들 surface객체임. 두번째 파라미터는 Surface 객체를 얼마나 회전시킬지 정하는 각도임. pygame.transform.rotate()함수는 넘겨준 Surface 객체를 변경하지 않고, 회전한 이미지를 그린 새로운 Surface객체를 생성해서 반환해줌.
회전시킨 2개의 'OSW Game' Surface 객체는 각 프레임의 디스플레이 Surface 객체상에 복사되어서 애니메이션 루프를 돌게됨

```
        degrees1 += 3 # rotate by 3 degrees each frame
        degrees2 += 7 # rotate by 7 degrees each frame
```

텍스트 Surface객체를 얼마나 회전시킬지 degree1과 degree2 변수에 저장함
각각 3도 7도씩 증가시킴

```
def terminate():
    pygame.quit()
    sys.exit()
```

terminate()함수는 pygame.quit()과 sys.exit()를 호출해 게임을 제대로 종료함

```
def getRandomLocation():
    return {'x': random.randint(0, CELLWIDTH - 1), 'y': random.randint(0, CELLHEIGHT - 1)}
```

사과를 놓을 새로운위치가 필요할 때 호출함. X,Y좌표를 무작위로 선택한 다음 키 값이 'x'와 'y'인 딕셔너리를 반환

```
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, BASICFONT, TEXTCOLOR)
    titleRect.center = (int(WINDOWWIDTH / 2), int(WINDOWHEIGHT / 2))
    DISPLAYSURF.blit(titleSurf, titleRect)

    # Draw the text
    titleSurf, titleRect = makeTextObjs(text, BASICFONT, 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 a key to play.', BASICFONT, TEXTCOLOR)
    pressKeyRect.center = (int(WINDOWWIDTH / 2), int(WINDOWHEIGHT / 2) + 100)
    DISPLAYSURF.blit(pressKeySurf, pressKeyRect)

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

앞선 과제를 수행하기 위해 일시정지 기능을 넣기 위해 정의한 함수 showTextScreen

Tetrino에서 참고하였으며 시연영상과 동일하게 하기 위해 BASICFONT로 수정함. 또한 TEXTCOLOR을 WHITE로 설정하여 해당 함수를 정의함.

```
def showGameOverScreen():
    gameOverFont = pygame.font.Font('D2Coding.ttc', 150)
    gameSurf = gameOverFont.render('Game', True, WHITE)
    overSurf = gameOverFont.render('Over', True, WHITE)
    gameRect = gameSurf.get_rect()
    overRect = overSurf.get_rect()
    gameRect.midtop = (WINDOWWIDTH / 2, 10)
    overRect.midtop = (WINDOWWIDTH / 2, gameRect.height + 10 + 25)

    DISPLAYSURF.blit(gameSurf, gameRect)
    DISPLAYSURF.blit(overSurf, overRect)
    drawPressKeyMsg()
    pygame.display.update()
```

게임 종료화면은 게임 시작 화면과 비슷한데 애니메이션은 없음. 'Game'과 'Over'를 각각 2개의 Surface 객체에 그려서 화면상에 나타나도록 함

```
    pygame.time.wait(500)
    checkForKeyPress() # clear out any key presses in the event queue

    while True:
        if checkForKeyPress():
            pygame.event.get() # clear event queue
            return
```

플레이어가 키를 누를 때까지 'Game Over'의 글씨가 화면에 계속 보임. 키를 너무 빨리 눌러 화면을 못보는 일이 없도록 pygame.time.wait()를 호출해서 0.5초 정도 기다림
checkForKeyPress()호출해서 showGameOverScreen()함수 호출 뒤 누른 키가 있으면 모두 큐에서 지움,

```
def drawScore(score):
    scoreSurf = BASICFONT.render('Score: %s' % (score), True, WHITE)
    scoreRect = scoreSurf.get_rect()
    scoreRect.topleft = (WINDOWWIDTH - 120, 10)
    DISPLAYSURF.blit(scoreSurf, scoreRect)
```

score파라미터로 받은 점수를 텍스트로 렌더링해서 디스플레이 객체상에 그림

```
def drawWorm(wormCoords):
    for coord in wormCoords:
        x = coord['x'] * CELLSIZE
        y = coord['y'] * CELLSIZE
        wormSegmentRect = pygame.Rect(x, y, CELLSIZE, CELLSIZE)
        pygame.draw.rect(DISPLAYSURF, DARKGREEN, wormSegmentRect)
        wormInnerSegmentRect = pygame.Rect(x + 4, y + 4, CELLSIZE - 8, CELLSIZE - 8)
        pygame.draw.rect(DISPLAYSURF, GREEN, wormInnerSegmentRect)
```

벌레 몸통에 해당하는 각 마디를 초록상자로 그림
각 마디는 WormCoords파라미터로 받았고 WormCoords는 'x'키와 'y'키를 가지고 있는 딕셔너리 리스트임. for문은 WormCoords의 각 딕셔너리 값에 대해 반복문을 수행함

```
def drawApple(coord):
    x = coord['x'] * CELLSIZE
    y = coord['y'] * CELLSIZE
    appleRect = pygame.Rect(x, y, CELLSIZE, CELLSIZE)
    pygame.draw.rect(DISPLAYSURF, RED, appleRect)
```

셀 하나를 사과의 빨간색으로 채움

```
def drawGrid():
    for x in range(0, WINDOWWIDTH, CELLSIZE): # draw vertical lines
        pygame.draw.line(DISPLAYSURF, DARKGRAY, (x, 0), (x, WINDOWHEIGHT))
    for y in range(0, WINDOWHEIGHT, CELLSIZE): # draw horizontal lines
        pygame.draw.line(DISPLAYSURF, DARKGRAY, (0, y), (WINDOWWIDTH, y))
```

셀을 구분하는 격자를 그리기 위해 pygame.draw.line()으로 가로선과 세로선을 모두 그림

```
if __name__ == '__main__':
    main()
```

모든 함수, 상수 그리고 전역변수를 정의하고 만든 다음 게임 시작 위해 main()함수 호출함

2. 함수의 호출 순서 또는 호출 조건