1. Snake Game

Code] (주석이나 설명 포함)

**1.Original Ver**

# -\*- coding: utf8 -\*-

from \_\_future\_\_ import print\_function

import random

import os

import re

from tabnanny import check

import time

import msvcrt

class Snake:

    def \_\_init\_\_(self, n):

        self.length = n

        self.head = []

        self.tail = []

class SnakeGame:

    direction = {"LEFT":-2, "DOWN":-1, "NON\_DIR":0, "UP":1, "RIGHT":2}

    sprite = {"EMPTY":0, "BODY":1, "HEAD":2, "FOOD":3}

    element = {"SPRITE":0, "DIRECTION":1}

    def \_\_init\_\_(self, w, h, length, delay):

        self.W = w

        self.H = h

        self.initLen = length

        self.snake = Snake(length)

        self.delay = delay

        self.board = [[[0]\*2 for x in range(self.W)] for y in range(self.H)]

        #self.board[a][b][c]

        #세로 / 가로

        self.snake.head = [self.H//2, self.snake.length-1]

        self.snake.tail = [self.H//2, 0]

        for i in range(0, self.snake.length):

            self.board[self.H//2][i][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

            self.board[self.H//2][i][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["RIGHT"]

        self.board[self.H//2][self.snake.length-1][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

        self.board[self.H//2][self.snake.length-1][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["RIGHT"]

        x = random.randint(0, self.W-1)

        y = random.randint(0, self.H-1)

        while self.board[y][x][SnakeGame.element["SPRITE"]] != SnakeGame.sprite["EMPTY"]:

            x = random.randint(0, self.W-1)

            y = random.randint(0, self.H-1)

        self.board[y][x][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["FOOD"]

    def DrawScene(self):

        os.system('cls||clear')

        for x in range(0, self.W+2):

            print("=", end="")

        print("")

        for y in range(0, self.H):

            print("|", end="")

            for x in range(0, self.W):

                if self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["BODY"]:

                    print("+", end="")

                elif  self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["HEAD"]:

                    print("@", end="")

                elif  self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]:

                    print("\*", end="")

                else:

                    print(" ", end="")

            print("|")

        for x in range(0, self.W+2):

            print("=", end="")

        print("")

    @staticmethod

    def GetDirection():

        rtn = SnakeGame.direction["NON\_DIR"]

        msvcrt.getch()

        ch = msvcrt.getch().decode()

        if ch == chr(72):

            print("UP")

            rtn = SnakeGame.direction["UP"]

        elif ch == chr(75):

            print("LEFT")

            rtn = SnakeGame.direction["LEFT"]

        elif ch == chr(77):

            print("RIGHT")

            rtn = SnakeGame.direction["RIGHT"]

        elif ch == chr(80):

            print("DOWN")

            rtn = SnakeGame.direction["DOWN"]

        return rtn

    def did\_eat(self):

        x = random.randint(0, self.W-1)

        y = random.randint(0, self.H-1)

        while (self.board[y][x][SnakeGame.element["SPRITE"]] != SnakeGame.sprite["EMPTY"]):

            x = random.randint(0, self.W-1)

            y = random.randint(0, self.H-1)

        self.board[y][x][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["FOOD"]

    def didnt\_eat(self):

        self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["EMPTY"]

        direction = self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["DIRECTION"]]

        self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["NON\_DIR"]

        if (direction == 1) :

            self.snake.tail[0] -= 1

        elif (direction == -1) :

            self.snake.tail[0] += 1

        elif (direction == 2) :

            self.snake.tail[1] += 1

        elif (direction == -2) :

            self.snake.tail[1] -= 1

    def check\_crash(self):

        if (self.snake.head[0] < 0 or self.snake.head[0] >= self.H or self.snake.head[1] < 0 or self.snake.head[1] >= self.W):

            print("Game Over")

            exit()

        if (self.snake.head[0] < self.H and self.snake.head[1] < self.W):

            if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["BODY"]):

                print("Game Over")

                exit()

    def GameLoop(self):

        self.DrawScene()

        ret = SnakeGame.direction["RIGHT"]

        current = SnakeGame.direction["RIGHT"]

        while True:

            start = time.time()

            while ((time.time() - start) <= self.delay/10000):

                if msvcrt.kbhit():

                    current = SnakeGame.GetDirection()

                if ((ret == current) or (ret == (current \* -1))):

                    current = ret

                self.check\_crash()

                #헤드 변환 지정

                if (current == 1):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["UP"]

                    self.snake.head[0] -= 1

                    self.check\_crash()

                    # 음식을 먹었는지 여부 확인

                    # 1. 음식을 먹었으면 길이가 길어짐

                    # 2. 음식을 먹지 않으면 전 단계의 꼬리의 물체는 빈 공간으로 채워야한다.

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                        self.did\_eat()

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["UP"]

                elif (current == -1):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["DOWN"]

                    self.snake.head[0] += 1

                    self.check\_crash()

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                        self.did\_eat()

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["DOWN"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                elif (current == 2):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["RIGHT"]

                    self.snake.head[1] += 1

                    self.check\_crash()

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                        self.did\_eat()

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["RIGHT"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                elif (current == -2):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["LEFT"]

                    self.snake.head[1] -= 1

                    self.check\_crash()

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                        self.did\_eat()

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["LEFT"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                time.sleep(0.1)

            ret = current

            self.DrawScene()

            print("Score: {}".format(self.snake.length - self.initLen))

if \_\_name\_\_ == '\_\_main\_\_' :

    game = SnakeGame(60, 24, 4, 300)

    game.GameLoop()

코드 분해 :

우선적으로 키를 입력 받으면 아래 코드에 걸려서 1. 같은 방향이면 누른 값을 유지 2. 반대(180도)이라도 같은 방향을 유지함.

                if ((ret == current) or (ret == (current \* -1))):

                    current = ret

방향 값을 받아 이 방향에 대해서 적용된 board값을 바꿔준다.

현재 헤드값은 body값으로 바꿔준다. 또한 방향에 대해서 누른 키 값으로 변화를 준다.

그리고 헤드의 값의 위치가 1상승한다. -> 위로 올라가야 함.

이후 충돌 체크를 해준다. (나중에 따로 함수로 빼서 설명함.)

음식처리에 대한 내용은 주석으로 설명됨 (나중에 따로 함수로 빼서 설명함)

움직인 헤드의 위치의 보드 값을 헤드와 업으로 바꿔준다.

if (current == 1):

    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["UP"]

    self.snake.head[0] -= 1

    self.check\_crash()

    # 음식을 먹었는지 여부 확인

    # 1. 음식을 먹었으면 길이가 길어짐

    # 2. 음식을 먹지 않으면 전 단계의 꼬리의 물체는 빈 공간으로 채워야한다.

    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

        self.snake.length += 1

        self.did\_eat()

    else:

        self.didnt\_eat()

self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["UP"]

만약에 먹이를 먹었다면 다시 먹이를 생성

def did\_eat(self):

    x = random.randint(0, self.W-1)

    y = random.randint(0, self.H-1)

    while (self.board[y][x][SnakeGame.element["SPRITE"]] != SnakeGame.sprite["EMPTY"]):

        x = random.randint(0, self.W-1)

        y = random.randint(0, self.H-1)

    self.board[y][x][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["FOOD"]

먹이를 먹지 않았다면 꼬리를 이동해줘야 함. 먹었다면 꼬리를 움직일 필요가 없음.

먹지 않았다면 현재 꼬리의 값은 비었다고 해줬다고 해주며 현재 꼬리의 방향의 값을 direction에 담아둔다.

그리고 현재 꼬리의 방향 값을 비워준다.

담아둔 direction에 정보에 따라 꼬리의 이동을 진행한다.

def didnt\_eat(self):

    self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["EMPTY"]

    direction = self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["DIRECTION"]]

    self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["NON\_DIR"]

    if (direction == 1) :

        self.snake.tail[0] -= 1

    elif (direction == -1) :

        self.snake.tail[0] += 1

    elif (direction == 2) :

        self.snake.tail[1] += 1

    elif (direction == -2) :

        self.snake.tail[1] -=

충돌에 대한 처리로 보드의 밖 테두리와 충돌한다면, 게임이 종료해야 한다. 또한 자신의 바디에 헤드가 닿는 다면, 종료해준다.

def check\_crash(self):

    if (self.snake.head[0] < 0 or self.snake.head[0] >= self.H or self.snake.head[1] < 0 or self.snake.head[1] >= self.W):

        print("Game Over")

        exit()

    if (self.snake.head[0] < self.H and self.snake.head[1] < self.W):

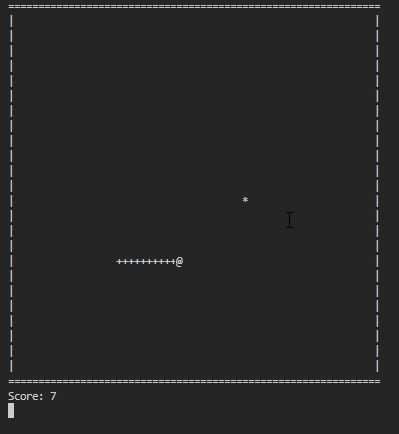
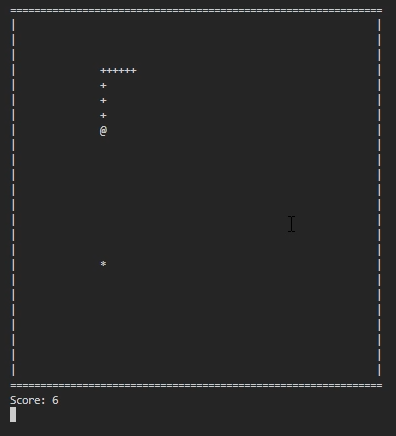
        if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["BODY"]):

            print("Game Over")

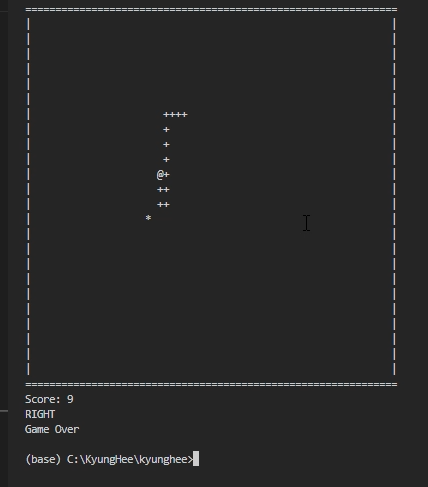
            exit()

Run] Screen shot

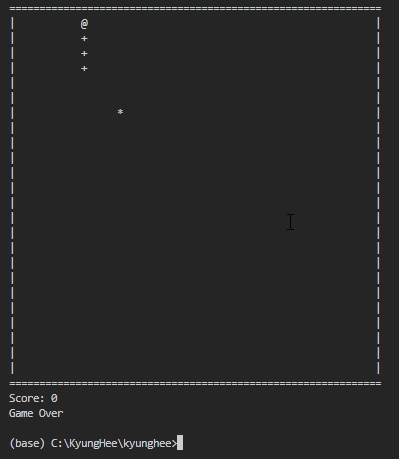
먹이를 먹었을 때 꼬리 생성 확인 및 먹이 랜덤 생성 확인



헤드가 자신의 머리와 충돌 시 Game Over



상하좌우 보드 충돌 체크



텍스트이(가) 표시된 사진

자동 생성된 설명

2. Eat All Ver : 채워져 있는 먹이를 모두 먹는 버전

# -\*- coding: utf8 -\*-

from \_\_future\_\_ import print\_function

from email import header

import random

import os

import re

from tabnanny import check

import time

import msvcrt

class Snake:

    def \_\_init\_\_(self, n):

        self.length = n

        self.head = []

        self.tail = []

class SnakeGame:

    direction = {"LEFT":-2, "DOWN":-1, "NON\_DIR":0, "UP":1, "RIGHT":2}

    sprite = {"EMPTY":0, "BODY":1, "HEAD":2, "FOOD":3}

    element = {"SPRITE":0, "DIRECTION":1}

    def \_\_init\_\_(self, w, h, length, delay, level):

        self.W = w

        self.H = h

        self.initLen = length

        self.snake = Snake(length)

        self.delay = delay

        self.board = [[[0]\*2 for x in range(self.W)] for y in range(self.H)]

        self.level = level

        #self.board[a][b][c]

        #세로 / 가로

        self.snake.head = [self.H//2, 0]

        self.snake.tail = [self.H//2, 0]

        for y in range(0, self.H):

            for x in range(0, self.W):

                if self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["EMPTY"]:

                    self.board[y][x][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["FOOD"]

    def DrawScene(self):

        os.system('cls||clear')

        for x in range(0, self.W+2):

            print("=", end="")

        print("")

        for y in range(0, self.H):

            print("|", end="")

            for x in range(0, self.W):

                if self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["BODY"]:

                    print(" ", end="")

                elif  self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["HEAD"]:

                    print("@", end="")

                elif  self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]:

                    print("\*", end="")

                else:

                    print(" ", end="")

            print("|")

        for x in range(0, self.W+2):

            print("=", end="")

        print("")

    @staticmethod

    def GetDirection():

        rtn = SnakeGame.direction["NON\_DIR"]

        msvcrt.getch()

        ch = msvcrt.getch().decode()

        if ch == chr(72):

            print("UP")

            rtn = SnakeGame.direction["UP"]

        elif ch == chr(75):

            print("LEFT")

            rtn = SnakeGame.direction["LEFT"]

        elif ch == chr(77):

            print("RIGHT")

            rtn = SnakeGame.direction["RIGHT"]

        elif ch == chr(80):

            print("DOWN")

            rtn = SnakeGame.direction["DOWN"]

        return rtn

    def didnt\_eat(self):

        self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["EMPTY"]

        direction = self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["DIRECTION"]]

        self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["NON\_DIR"]

        if (direction == 1) :

            self.snake.tail[0] -= 1

        elif (direction == -1) :

            self.snake.tail[0] += 1

        elif (direction == 2) :

            self.snake.tail[1] += 1

        elif (direction == -2) :

            self.snake.tail[1] -= 1

    def check\_crash(self):

        if (self.snake.head[0] < 0 or self.snake.head[0] >= self.H or self.snake.head[1] < 0 or self.snake.head[1] >= self.W):

            print("Game Over")

            exit()

    def GameLoop(self):

        self.DrawScene()

        ret = SnakeGame.direction["RIGHT"]

        current = SnakeGame.direction["RIGHT"]

        while True:

            start = time.time()

            while ((time.time() - start) <= self.delay/10000):

                if msvcrt.kbhit():

                    current = SnakeGame.GetDirection()

                if ((ret == current) or (ret == (current \* -1))):

                    current = ret

                self.check\_crash()

                #헤드 변환 지정

                if (current == 1):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["UP"]

                    self.snake.head[0] -= 1

                    self.check\_crash()

                    # 음식을 먹었는지 여부 확인

                    # 1. 음식을 먹었으면 길이가 길어짐

                    # 2. 음식을 먹지 않으면 전 단계의 꼬리의 물체는 빈 공간으로 채워야한다.

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["UP"]

                elif (current == -1):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["DOWN"]

                    self.snake.head[0] += 1

                    self.check\_crash()

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["DOWN"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                elif (current == 2):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["RIGHT"]

                    self.snake.head[1] += 1

                    self.check\_crash()

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["RIGHT"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                elif (current == -2):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["LEFT"]

                    self.snake.head[1] -= 1

                    self.check\_crash()

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["LEFT"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                if (level == 1):

                    time.sleep(0.2)

                elif (level == 2):

                    time.sleep(0.08)

                elif (level == 3):

                    time.sleep(0.02)

            if (self.W == 25):

                if (self.snake.length -self.initLen == 25 \* 10 - 1):

                    print("you win!")

                    exit()

            if (self.W == 35):

                if (self.snake.length -self.initLen == 35 \* 14 - 1):

                    print("you win!")

                    exit()

            if (self.W == 45):

                if (self.snake.length -self.initLen == 45 \* 18 - 1):

                    print("you win!")

                    exit()

            ret = current

            self.DrawScene()

            print("Score: {}".format(self.snake.length - self.initLen))

if \_\_name\_\_ == '\_\_main\_\_' :

    print("select size of map : ")

    print("win conditions : eat all food")

    map\_size = int(input("1. 25X10     2. 35X14     3. 45X18\n"))

    print("select speed level: ")

    level = int(input("1. easy    2. medium   3. hard\n"))

    if map\_size == 1:

        game = SnakeGame(25, 10, 4, 300, level)

    elif map\_size == 2:

        game = SnakeGame(35, 14, 4, 300, level)

    elif map\_size == 3:

        game = SnakeGame(45, 18, 4, 300, level)

    game.GameLoop()

코드 분해 :

맵의 전체의 크기를 먼저 입력을 받는다.

이후 헤드의 움직임 속도를 입력받는다.

print("select size of map : ")

print("win conditions : eat all food")

map\_size = int(input("1. 25X10     2. 35X14     3. 45X18\n"))

print("select speed level: ")

level = int(input("1. easy    2. medium   3. hard\n"))

if map\_size == 1:

    game = SnakeGame(25, 10, 4, 300, level)

elif map\_size == 2:

    game = SnakeGame(35, 14, 4, 300, level)

elif map\_size == 3:

    game = SnakeGame(45, 18, 4, 300, level)

처음 맵을 전부다 음식으로 채워 넣는다.

for y in range(0, self.H):

    for x in range(0, self.W):

        if self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["EMPTY"]:

            self.board[y][x][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["FOOD"]

레벨을 정해서 이에 따라 움직이는 속도를 정해준다.

또한 W값에 따라서 먹어야 하는 먹이의 수를 정해두고 이에 맞춰서 게임의 승리 조건을 정함.

if (level == 1):

    time.sleep(0.2)

elif (level == 2):

    time.sleep(0.08)

elif (level == 3):

    time.sleep(0.02)

if (self.W == 25):

if (self.snake.length -self.initLen == 25 \* 10 - 1):

    print("you win!")

    exit()

if (self.W == 35):

if (self.snake.length -self.initLen == 35 \* 14 - 1):

    print("you win!")

    exit()

if (self.W == 45):

if (self.snake.length -self.initLen == 45 \* 18 - 1):

    print("you win!")

    exit()

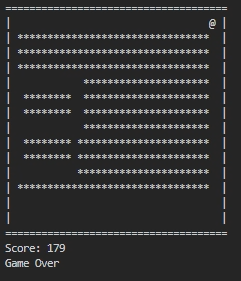
텍스트이(가) 표시된 사진

자동 생성된 설명

플레이 예시

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

3. Block Ver : 먹이를 먹을 때마다 블록인 N이 생성된다. N과 부딛치면 게임이 종료된다.

# -\*- coding: utf8 -\*-

from \_\_future\_\_ import print\_function

from distutils import extension

import random

import os

import re

from tabnanny import check

import time

import msvcrt

class Snake:

    def \_\_init\_\_(self, n):

        self.length = n

        self.head = []

        self.tail = []

class SnakeGame:

    direction = {"LEFT":-2, "DOWN":-1, "NON\_DIR":0, "UP":1, "RIGHT":2}

    sprite = {"EMPTY":0, "BODY":1, "HEAD":2, "FOOD":3, "BLOCK" : 4}

    element = {"SPRITE":0, "DIRECTION":1}

    def \_\_init\_\_(self, w, h, length, delay):

        self.W = w

        self.H = h

        self.initLen = length

        self.snake = Snake(length)

        self.delay = delay

        self.board = [[[0]\*2 for x in range(self.W)] for y in range(self.H)]

        #self.board[a][b][c]

        #세로 / 가로

        self.snake.head = [self.H//2, self.snake.length-1]

        self.snake.tail = [self.H//2, 0]

        for i in range(0, self.snake.length):

            self.board[self.H//2][i][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

            self.board[self.H//2][i][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["RIGHT"]

        self.board[self.H//2][self.snake.length-1][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

        self.board[self.H//2][self.snake.length-1][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["RIGHT"]

        x = random.randint(0, self.W-1)

        y = random.randint(0, self.H-1)

        while self.board[y][x][SnakeGame.element["SPRITE"]] != SnakeGame.sprite["EMPTY"]:

            x = random.randint(0, self.W-1)

            y = random.randint(0, self.H-1)

        self.board[y][x][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["FOOD"]

        x = random.randint(0, self.W-1)

        y = random.randint(0, self.H-1)

        while self.board[y][x][SnakeGame.element["SPRITE"]] != SnakeGame.sprite["EMPTY"]:

            x = random.randint(0, self.W-1)

            y = random.randint(0, self.H-1)

        self.board[y][x][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BLOCK"]

    def DrawScene(self):

        os.system('cls||clear')

        for x in range(0, self.W+2):

            print("=", end="")

        print("")

        for y in range(0, self.H):

            print("|", end="")

            for x in range(0, self.W):

                if self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["BODY"]:

                    print("+", end="")

                elif  self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["HEAD"]:

                    print("@", end="")

                elif  self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]:

                    print("\*", end="")

                elif  self.board[y][x][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["BLOCK"]:

                    print("N", end="")

                else:

                    print(" ", end="")

            print("|")

        for x in range(0, self.W+2):

            print("=", end="")

        print("")

    @staticmethod

    def GetDirection():

        rtn = SnakeGame.direction["NON\_DIR"]

        msvcrt.getch()

        ch = msvcrt.getch().decode()

        if ch == chr(72):

            print("UP")

            rtn = SnakeGame.direction["UP"]

        elif ch == chr(75):

            print("LEFT")

            rtn = SnakeGame.direction["LEFT"]

        elif ch == chr(77):

            print("RIGHT")

            rtn = SnakeGame.direction["RIGHT"]

        elif ch == chr(80):

            print("DOWN")

            rtn = SnakeGame.direction["DOWN"]

        return rtn

    def did\_eat(self):

        x = random.randint(0, self.W-1)

        y = random.randint(0, self.H-1)

        while (self.board[y][x][SnakeGame.element["SPRITE"]] != SnakeGame.sprite["EMPTY"]):

            x = random.randint(0, self.W-1)

            y = random.randint(0, self.H-1)

        self.board[y][x][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["FOOD"]

        x = random.randint(0, self.W-1)

        y = random.randint(0, self.H-1)

        while self.board[y][x][SnakeGame.element["SPRITE"]] != SnakeGame.sprite["EMPTY"]:

            x = random.randint(0, self.W-1)

            y = random.randint(0, self.H-1)

        self.board[y][x][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BLOCK"]

    def didnt\_eat(self):

        self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["EMPTY"]

        direction = self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["DIRECTION"]]

        self.board[self.snake.tail[0]][self.snake.tail[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["NON\_DIR"]

        if (direction == 1) :

            self.snake.tail[0] -= 1

        elif (direction == -1) :

            self.snake.tail[0] += 1

        elif (direction == 2) :

            self.snake.tail[1] += 1

        elif (direction == -2) :

            self.snake.tail[1] -= 1

    def check\_crash(self):

        if (self.snake.head[0] < 0 or self.snake.head[0] >= self.H or self.snake.head[1] < 0 or self.snake.head[1] >= self.W):

            print("Game Over")

            exit()

        if (self.snake.head[0] < self.H and self.snake.head[1] < self.W):

            if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["BODY"]):

                print("Game Over")

                exit()

        if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["BLOCK"]):

            print("Game Over")

            exit()

    def GameLoop(self):

        self.DrawScene()

        ret = SnakeGame.direction["RIGHT"]

        current = SnakeGame.direction["RIGHT"]

        while True:

            start = time.time()

            while ((time.time() - start) <= self.delay/10000):

                if msvcrt.kbhit():

                    current = SnakeGame.GetDirection()

                if ((ret == current) or (ret == (current \* -1))):

                    current = ret

                self.check\_crash()

                #헤드 변환 지정

                if (current == 1):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["UP"]

                    self.snake.head[0] -= 1

                    self.check\_crash()

                    # 음식을 먹었는지 여부 확인

                    # 1. 음식을 먹었으면 길이가 길어짐

                    # 2. 음식을 먹지 않으면 전 단계의 꼬리의 물체는 빈 공간으로 채워야한다.

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                        self.did\_eat()

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["UP"]

                elif (current == -1):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["DOWN"]

                    self.snake.head[0] += 1

                    self.check\_crash()

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                        self.did\_eat()

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["DOWN"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                elif (current == 2):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["RIGHT"]

                    self.snake.head[1] += 1

                    self.check\_crash()

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                        self.did\_eat()

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["RIGHT"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                elif (current == -2):

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["BODY"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["LEFT"]

                    self.snake.head[1] -= 1

                    self.check\_crash()

                    if (self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] == SnakeGame.sprite["FOOD"]):

                        self.snake.length += 1

                        self.did\_eat()

                    else:

                        self.didnt\_eat()

                    #새로 바뀐 헤드 위치에 대해서 보드값 변환

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["DIRECTION"]] = SnakeGame.direction["LEFT"]

                    self.board[self.snake.head[0]][self.snake.head[1]][SnakeGame.element["SPRITE"]] = SnakeGame.sprite["HEAD"]

                time.sleep(0.1)

            ret = current

            self.DrawScene()

            print("Score: {}".format(self.snake.length - self.initLen))

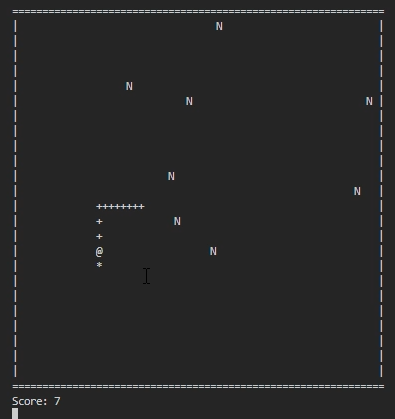
if \_\_name\_\_ == '\_\_main\_\_' :

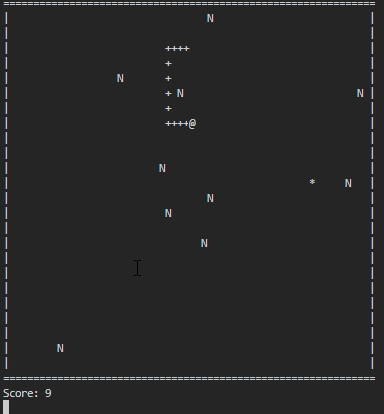
    game = SnakeGame(60, 24, 4, 300)

    game.GameLoop()

기본 코드와 다른 것이 거의 없지만, 랜덤 함수에 의해서 음식과 같이 음식이 사라졌을 때 블록을 생성하고 충돌조건을 두어 만듦

게임 플레이 :





텍스트이(가) 표시된 사진

자동 생성된 설명