스프라이트 (Sprite)

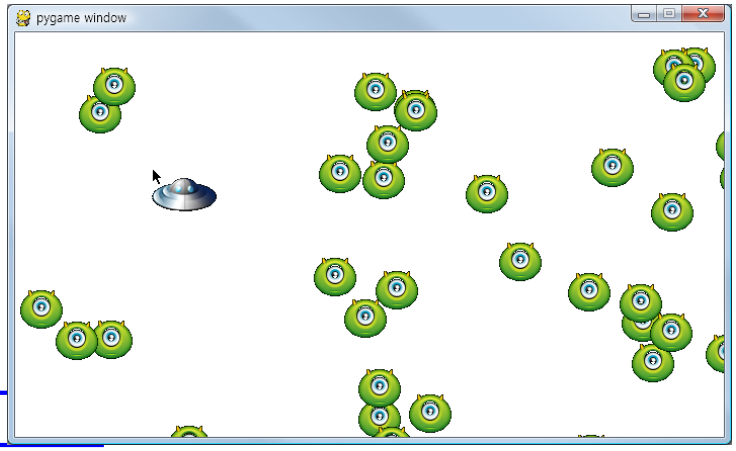
순천향대학교 컴퓨터공학과
이 상 정

학습 내용

- 스프라이트 소개
- 스프라이트 클래스
- 그룹 클래스
- 스프라이트 충돌
- 블록 수집 게임 예
- 게임 레벨 증가 및 점수 표시
- 스프라이트 이동

스프라이트 소개

- 스프라이트 (sprite)
  - 큰 그래픽 장면의 부분으로 사용되는 단일 2차원 이미지
    => 쪽화면
  - 게임의 장면에서 서로 상호작용(충돌 등) 하는 물체
    => 캐릭터, 아바타
  - 파이게임에서는 일반적으로 클래스로 구현된 객체



스프라이트 클래스

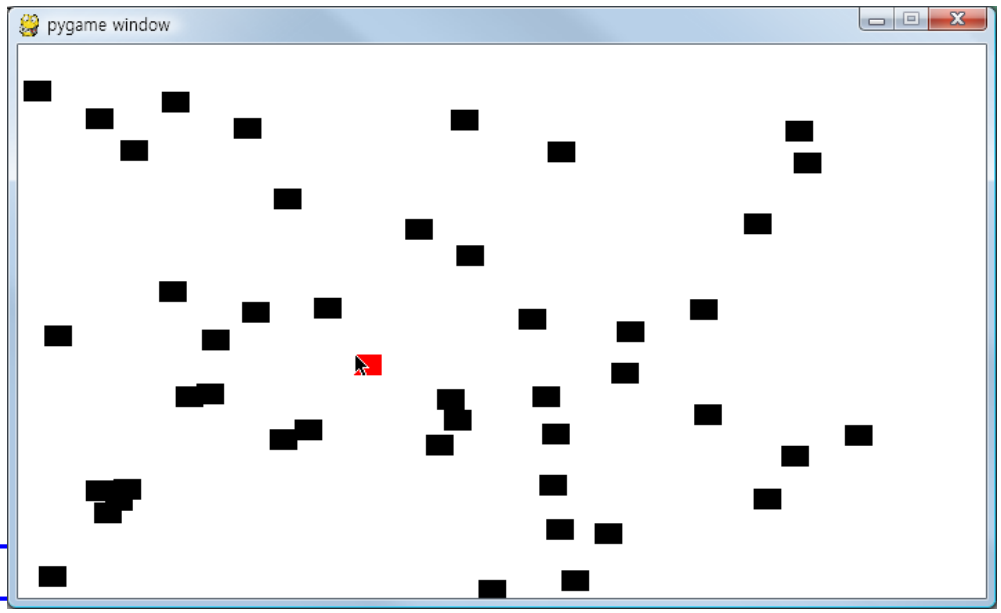
- 스프라이트 기본 클래스
  - pygame.sprite.Sprite(*groups): return Sprite
  - 속성
    - Sprite.image
      - 스프라이트 이미지 (Surface)
    - Sprite.rect
      - 스프라이트 위치 ((x,y), width, height)
  - 모든 게임 객체(물체)들의 기본 클래스(base class)
  - Group 클래스와 함께 동작
  - 생성자는 그룹 또는 그룹의 리스트를 인수로 전달

그룹 클래스

- 그룹 클래스
  - pygame.sprite.Group(*sprites): return Group
  - 다수의 스프라이트 객체들을 다루는 컨테이너(container) 클래스
  - 메서드
    - Group.add(*sprites): return None
      - 그룹에 스프라이트를 추가
    - Group.remove(*sprites): return None
      - 그룹에서 스프라이트를 제거
    - Group.draw(Surface): return None
      - Surface에 포함된 스프라이트를 그리기
    - Group.update(*args): return None
      - 그룹 내 모든 스프라이트의 update() 메서드 호출

블록 수집 게임 예

- 마우스로 움직이는 빨간 블록이 검은 블록을 수집하는 예
  - 검은 블록 스프라이트, 빨간 블록 스프라이트(플레이어)
  - 충돌 후 검은 블록 없어지고 점수 올라감



Block 클래스 정의

- Sprite 클래스를 상속받아 블록을 정의하는 클래스
  - 블록의 색과 크기를 생성자 인수로 전달

```
# 블록을 정의하는 클래스
# Sprite 부모 클래스를 상속
class Block(pygame.sprite.Sprite):
        # 블록의 색과 크기를 생성자 인수로 전달
        def __init__(self, color, width, height):
                # 부모 클래스 생성자 호출
                pygame.sprite.Sprite.__init__(self)
                # 블록의 이미지와 색 지정
                self.image = pygame.Surface([width, height])
                self.image.fill(color)
                # 이미지 크기의 rect 객체 지정
                self.rect = self.image.get_rect()
```

```
# 검은 블록 생성
 block = Block(black, 20, 15)

# 빨간 블록 생성
player = Block(red, 20, 15)
```


다양한 모양의 스프라이트 생성 예

```
# 타원의 이미지와 색 지정
self.image = pygame.Surface([width, height])
self.image.fill(white)
self.image.set_colorkey(white)
pygame.draw.ellipse(self.image, color,[0,0,width,height])
```

```
# 검은 타원 생성
 block = Block(black, 20, 15)

# 빨간 타원 생성
player = Block(red, 20, 15)
```

```
def __init__(self, filename):
        # 부모 클래스 생성자 호출
        pygame.sprite.Sprite.__init__(self)
        # 블록의 이미지와 색 지정
        self.image = pygame.image.load(filename).convert()
        self.image.set_colorkey(black)
        # 이미지 크기의 rect 객체 지정
        self.rect = self.image.get_rect()
```

```
# 외계인 블록 생성
block = Block("alien.png")
```


```
# UFO 블록 생성
player = Block("ufo.png")
```


스프라이트 그룹 생성

- 전체 스프라이트 객체를 갖는 스프라이트 그룹 생성
  - 검은 블록의 스프라이트 그룹
  - 플레이어를 포함한 모든 스프라이트 그룹

```
# 스프라이트 그룹 생성, 검은 블록의 스프라이트 그룹
block_list = pygame.sprite.Group()

# 스프라이트 그룹 생성, 플레이어(빨간 블록)를 포함한 모든 스프라이트 그룹
all_sprites_list = pygame.sprite.Group()
```


그룹에 스프라이트 추가

- 각 그룹에 스프라이트 추가
  - 그룹에 50개의 블록 객체 추가
  - 위치는 랜덤으로 지정

```
for i in range(50):
    # 검은 블록 생성
    block = Block(black, 20, 15)
    # 블록들의 위치를 랜덤 생성
    block.rect.x = random.randrange(screen_width)
    block.rect.y = random.randrange(screen_height)
    # 스프라이트 그룹에 블록 객체 추가
    block_list.add(block)
    all_sprites_list.add(block)

# 플레이어(빨간 블록) 생성
player = Block(red, 20, 15)
all_sprites_list.add(player)
```

스프라이트 충돌 (1)

- 한 스프라이트와 그룹 내 스프라이트 간의 충돌 조사
  - pygame.sprite.spritecollide(sprite, group, dokill, collided = None): return Sprite_list
  - sprite와 group 내의 스프라이트 간의 충돌 조사
  - dokill 인수가 True이면 충돌된 스프라이트가 그룹에서 제거
  - collided 인수는 두 스프라이트의 충돌 여부를 계산하는 callback 함수
    - 생략되면 스프라이트 영역이 rect 값을 가지고, 이를 충돌 계산에 적용
  - 충돌한 스프라이트들의 리스트 리턴

스프라이트 충돌 (2)

```
# 플레이어 블록과 충돌하는 블록 조사
blocks_hit_list = pygame.sprite.spritecollide(player, block_list, True)

# 충돌된 블록 수 조사하여 점수 조정
if len(blocks_hit_list) > 0:
    score +=len(blocks_hit_list)
    print( score )

# 모든 스프라이트 그리기
all_sprites_list.draw(screen)
```

블록 수집 게임 코드 (1)

```
import pygame
import random

black = ( 0, 0, 0)
white = ( 255, 255, 255)
red = ( 255, 0, 0)

# 블록을 정의하는 클래스
# Sprite 부모 클래스를 상속
class Block(pygame.sprite.Sprite):
        # 블록의 색과 크기를 생성자 인수로 전달
        def __init__(self, color, width, height):
            # 부모 클래스 생성자 호출
            pygame.sprite.Sprite.__init__(self)
            # 블록의 이미지와 색 지정
            self.image = pygame.Surface([width, height])
            self.image.fill(color)
            # 이미지 크기의 rect 객체 지정
            self.rect = self.image.get_rect()

pygame.init()
```

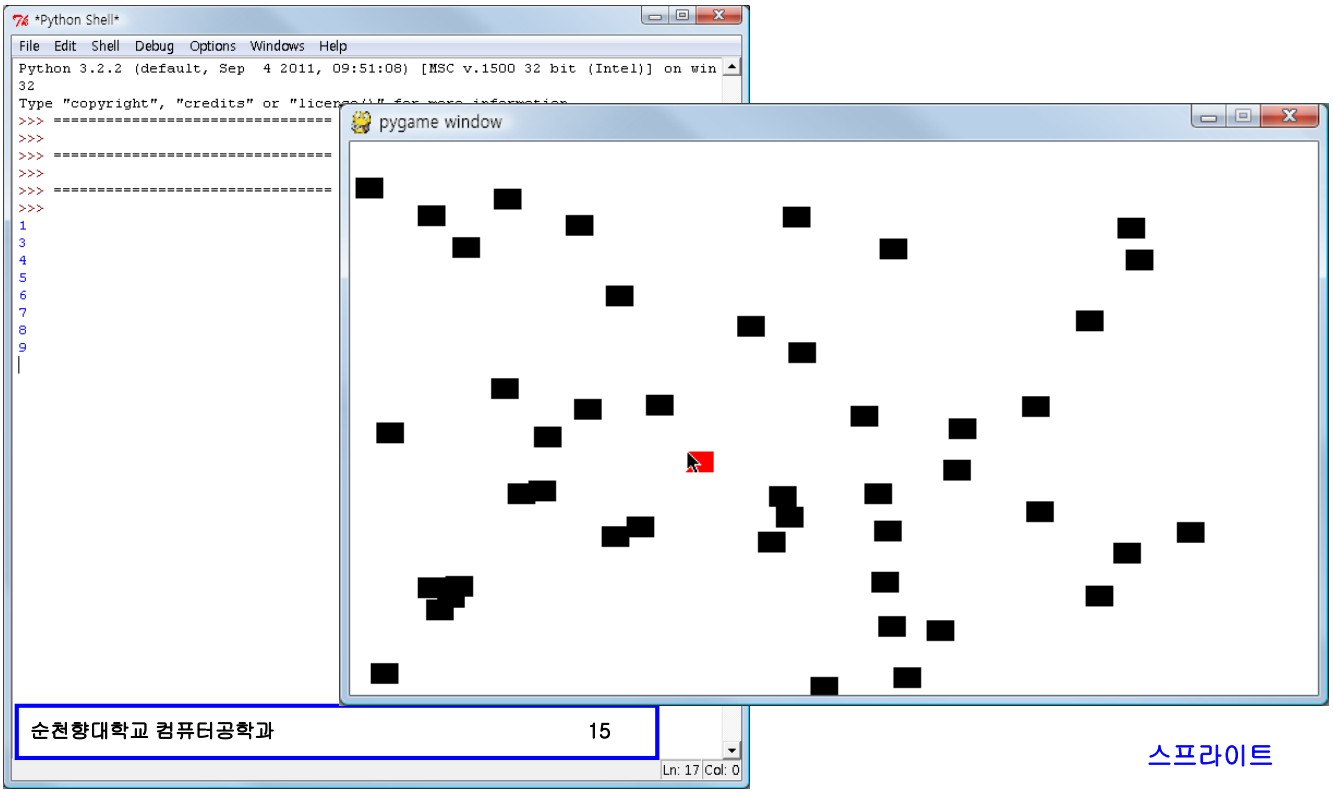
```
# 윈도우 설정
screen_width=700
screen_height=400
screen=pygame.display.set_mode([screen_width,
                                screen_height])

# 스프라이트 그룹 생성, 검은 블록의 스프라이트  그룹
block_list = pygame.sprite.Group()
# 스프라이트 그룹 생성, 플레이어(빨간 블록)를 포함한
# 모든 스프라이트 그룹
all_sprites_list = pygame.sprite.Group()

for i in range(50):
    # 검은 블록 생성
    block = Block(black, 20, 15)
    # 블록들의 위치를 랜덤 생성
    block.rect.x = random.randrange(screen_width)
    block.rect.y = random.randrange(screen_height)
    # 스프라이트 그룹에 블록 객체 추가
    block_list.add(block)
    all_sprites_list.add(block)
```


블록 수집 게임 코드 (2)

```
 # 플레이어 블록(빨간 블록) 생성
player = Block(red, 20, 15)
all_sprites_list.add(player)

done=False
clock=pygame.time.Clock()
score = 0

while done==False:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            done=True

    screen.fill(white)

    # 현재 마우스 위치 읽어서 플레이어 블록 객체의
    # 위치로 지정
    pos = pygame.mouse.get_pos()
    player.rect.x=pos[0]
    player.rect.y=pos[1]
```

```
    # 플레이어 블록과 충돌하는 블록 조사
    blocks_hit_list =
      pygame.sprite.spritecollide (player, block_list, True)

    # 충돌된 블록 수 조사하여 점수 조정
    if len(blocks_hit_list) > 0:
        score +=len(blocks_hit_list)
        print( score )

    # 모든 스프라이트 그리기
    all_sprites_list.draw(screen)

    clock.tick(20)
    pygame.display.flip()

pygame.quit()
```

시험주행



블록 수집 게임 – 게임 레벨 증가

- 모든 블록을 수집했으면 레벨 상승
  - 레벨에 따라 새로운 블록 추가

```
# 플레이어 블록과 충돌하는 블록 조사
blocks_hit_list = pygame.sprite.spritecollide(player, block_list, True)

# 모든 블록을 수집했으면 레벨 상승
if len(block_list) == 0:
    level += 1
    # 레벨에 따라 새로운 블록 추가
    for i in range(level * 10):
        # 검은 블록 생성
        block = Block(black, 20, 15)
        # 블록들의 위치를 랜덤 생성
        block.rect.x = random.randrange(screen_width)
        block.rect.y = random.randrange(screen_height)
        # 스프라이트 그룹에 블록 객체 추가
        block_list.add(block)
        all_sprites_list.add(block)
```

블록 수집 게임 – 게임 점수 표시

- 윈도우에 점수와 레벨 표시

```
# 텍스트 폰트 생성
font = pygame.font.Font("nanumgothicBold.ttf", 25)

# 충돌된 블록 수 조사하여 점수 조정
if len(blocks_hit_list) > 0:
    score +=len(blocks_hit_list)

# 점수 및 레벨 텍스트 표시
text=font.render("점수: "+str(score), True, black)
screen.blit(text, [10, 10])
text=font.render("레벨: "+str(level), True, black)
screen.blit(text, [10, 40])
```


블록 수집 코드 – 레벨 증가 및 점수 표시 (1)

```
import pygame
import random

black = ( 0, 0, 0)
white = ( 255, 255, 255)
red = ( 255, 0, 0)

# 블록을 정의하는 클래스
# Sprite 부모 클래스를 상속
class Block(pygame.sprite.Sprite):
        # 블록의 색과 크기를 생성자 인수로 전달
        def __init__(self, color, width, height):
            # 부모 클래스 생성자 호출
            pygame.sprite.Sprite.__init__(self)
            # 블록의 이미지와 색 지정
            self.image = pygame.Surface([width, height])
            self.image.fill(color)
            # 이미지 크기의 rect 객체 지정
            self.rect = self.image.get_rect()

pygame.init()
```

```
# 윈도우 설정
screen_width=700
screen_height=400
screen=pygame.display.set_mode([screen_width,
                                screen_height])

# 스프라이트 그룹 생성, 검은 블록의 스프라이트  그룹
block_list = pygame.sprite.Group()
# 스프라이트 그룹 생성, 플레이어(빨간 블록)를 포함한
# 모든 스프라이트 그룹
all_sprites_list = pygame.sprite.Group()

for i in range(50):
    # 검은 블록 생성
    block = Block(black, 20, 15)
    # 블록들의 위치를 랜덤 생성
    block.rect.x = random.randrange(screen_width)
    block.rect.y = random.randrange(screen_height)
    # 스프라이트 그룹에 블록 객체 추가
    block_list.add(block)
    all_sprites_list.add(block)
```

블록 수집 코드 – 레벨 증가 및 점수 표시 (2)

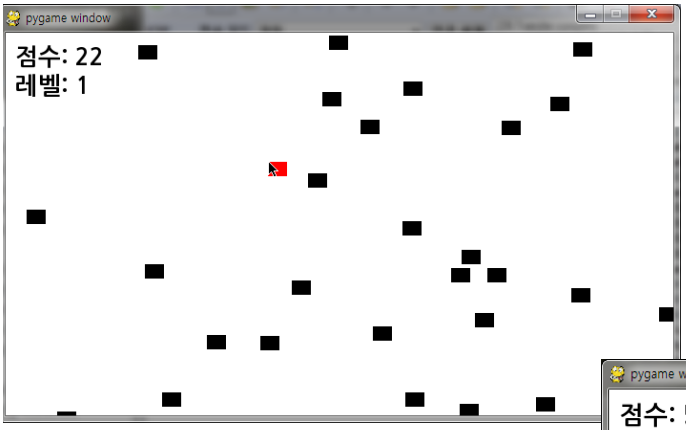
```
 # 플레이어 블록(빨간 블록) 생성
player = Block(red, 20, 15)
all_sprites_list.add(player)

done=False
clock=pygame.time.Clock()

# 텍스트 폰트 생성
font = pygame.font.Font("nanumgothicBold.ttf",
                                                25)
score = 0
level = 1

while done==False:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            done=True

    screen.fill(white)
    # 현재 마우스 위치 읽어서 플레이어 블록 객체의
    #위치로 지정
    pos = pygame.mouse.get_pos()
    player.rect.x=pos[0]
    player.rect.y=pos[1]

    # 플레이어 블록과 충돌하는 블록 조사
    blocks_hit_list = pygame.sprite.spritecollide(player,
                                        block_list, True)

    # 충돌된 블록 수 조사하여 점수 조정
    if len(blocks_hit_list) > 0:
        score +=len(blocks_hit_list)

    # 모든 블록을 수집했으면 레벨 상승
    if len(block_list) == 0:
        level += 1
        # 레벨에 따라 새로운 블록 추가
        for i in range(level * 10):
            # 검은 블록 생성
            block = Block(black, 20, 15)
            # 블록들의 위치를 랜덤 생성
            block.rect.x = random.randrange(screen_width)
```

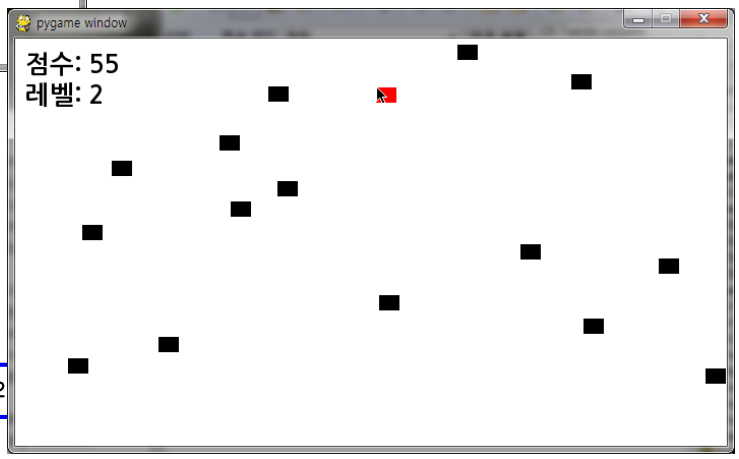
GUI 설계기법

블록 수집 코드 – 레벨 증가 및 점수 표시 (2)

```
            block.rect.y = random.randrange(screen_height)
            # 스프라이트 그룹에 블록 객체 추가
            block_list.add(block)
            all_sprites_list.add(block)

    # 모든 스프라이트 그리기
    all_sprites_list.draw(screen)

    # 점수 및 레벨 텍스트 표시
    text=font.render("점수: "+str(score), True, black)
    screen.blit(text, [10, 10])
    text=font.render("레벨: "+str(level), True, black)
    screen.blit(text, [10, 40])

    clock.tick(20)
    pygame.display.flip()

pygame.quit()
```

시험주행





스프라이트 이동

- 스프라이트가 위에서 아래로 이동
  - Block 클래스에 update() 메서드 추가
    - 각 프레임에서 블록의 y 좌표 증가
      - 증가 속도는 레벨에 따라 커짐
    - 블록이 아래 끝에 도달하면 y 위치 값을 위로 조정

```
class Block(pygame.sprite.Sprite):
        ……
        # y 좌표 값 리셋
        def reset_pos(self):
            self.rect.y = random.randrange(-100,-10)
            self.rect.x = random.randrange(0,screen_width)

        def update(self, change_y):
            # 블록을 아래로 이동
            self.rect.y += change_y
            if self.rect.y > screen_height:
                self.reset_pos()
```

블록 수집 코드 – 블록 이동 (1)

```
import pygame
import random

black = ( 0, 0, 0)
white = ( 255, 255, 255)
red = ( 255, 0, 0)

# 블록을 정의하는 클래스
# Sprite 부모 클래스를 상속
class Block(pygame.sprite.Sprite):
        # 블록의 색과 크기를 생성자 인수로 전달
       def __init__(self, color, width, height):
          # 부모 클래스 생성자 호출
          pygame.sprite.Sprite.__init__(self)
          # 블록의 이미지와 색 지정
          self.image = pygame.Surface([width,
    height])
          self.image.fill(color)
          # 이미지 크기의 rect 객체 지정
          self.rect = self.image.get_rect()
```



```
    # y 좌표 값 리셋
    def reset_pos(self):
        self.rect.y = random.randrange(-100,-10)
        self.rect.x = random.randrange(0,screen_width)

    def update(self, change_y):
        # 블록을 아래로 이동
        self.rect.y += change_y
        if self.rect.y > screen_height:
            self.reset_pos()

pygame.init()

# 윈도우 설정
screen_width=700
screen_height=400
screen=pygame.display.set_mode([screen_width,
                                screen_height])
# 스프라이트 그룹 생성
block_list = pygame.sprite.Group()
all_sprites_list = pygame.sprite.Group()
```


블록 수집 코드 – 블록 이동 (2)

```
for i in range(50):
    # 검은 블록 생성
    block = Block(black, 20, 15)
    # 블록들의 위치를 랜덤 생성
    block.rect.x = random.randrange(screen_width)
    block.rect.y = random.randrange(screen_height)
    # 스프라이트 그룹에 블록 객체 추가
    block_list.add(block)
    all_sprites_list.add(block)

# 플레이어 블록(빨간 블록) 생성
player = Block(red, 20, 15)
all_sprites_list.add(player)

done=False
clock=pygame.time.Clock()
# 텍스트 폰트 생성
font = pygame.font.Font("nanumgothicBold.ttf", 25)

score = 0
level = 1
```

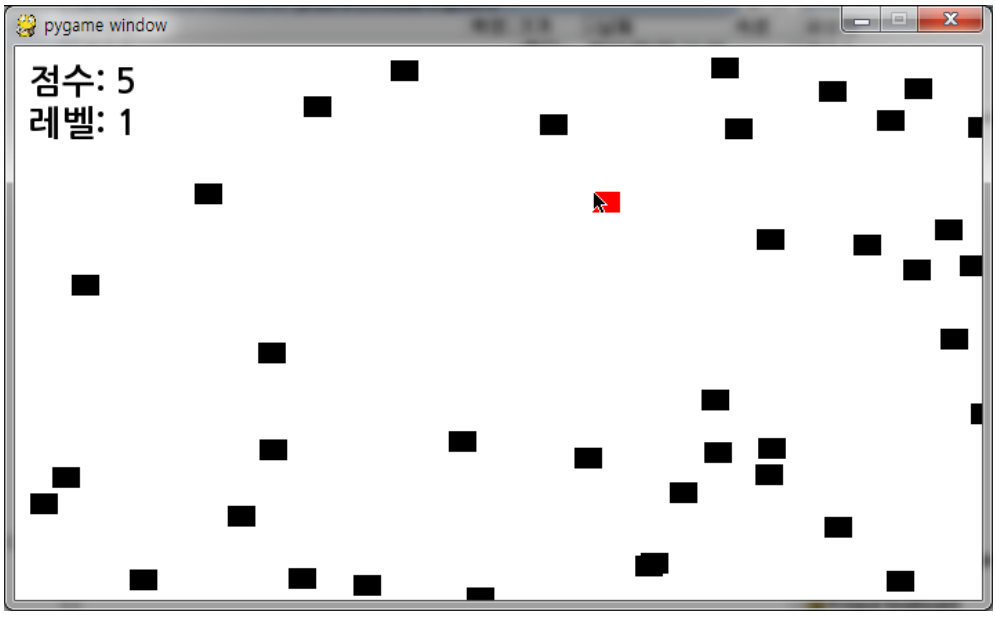
```
while done==False:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            done=True

    screen.fill(white)
    # 현재 마우스 위치를 플레이어 블록의 위치로 지정
    pos = pygame.mouse.get_pos()
    player.rect.x=pos[0]
    player.rect.y=pos[1]

    # 플레이어 블록과 충돌하는 블록 조사
    blocks_hit_list = pygame.sprite.spritecollide(player,
                                    block_list, True)

    # 충돌된 블록 수 조사하여 점수 조정
    if len(blocks_hit_list) > 0:
        score +=len(blocks_hit_list)
    # 모든 블록을 수집했으면 레벨 상승
    if len(block_list) == 0:
        level += 1
```

블록 수집 코드 – 블록 이동 (3)

```
    # 레벨에 따라 새로운 블록 추가
    for i in range(level * 10):
        # 검은 블록 생성
        block = Block(black, 20, 15)
        # 블록들의 위치를 랜덤 생성
        block.rect.x = random.randrange(screen_width)
        block.rect.y = random.randrange(screen_height)
        # 스프라이트 그룹에 블록 객체 추가
        block_list.add(block)
        all_sprites_list.add(block)

# 모든 스프라이트 그리기
all_sprites_list.draw(screen)

# 점수 및 레벨 텍스트 표시
text=font.render("점수: "+str(score), True, black)
screen.blit(text, [10, 10])
text=font.render("레벨: "+str(level), True, black)
screen.blit(text, [10, 40])
```

```
    # 그룹 내 스프라이트 update() 메서드 호출
    # 이동 속도는 레벨 값으로 지정
    block_list.update(level)

    clock.tick(20)
    pygame.display.flip()

pygame.quit()
```


시험주행



과제

1. *앞에서 소개된 블록 수집 프로그램을 작성하고 실행*

2. 앞에서 배운 내용을 활용한 임의의 프로그램 작성
   - 프로그램 설명
   - 프로그램 소스
   - 실행 결과