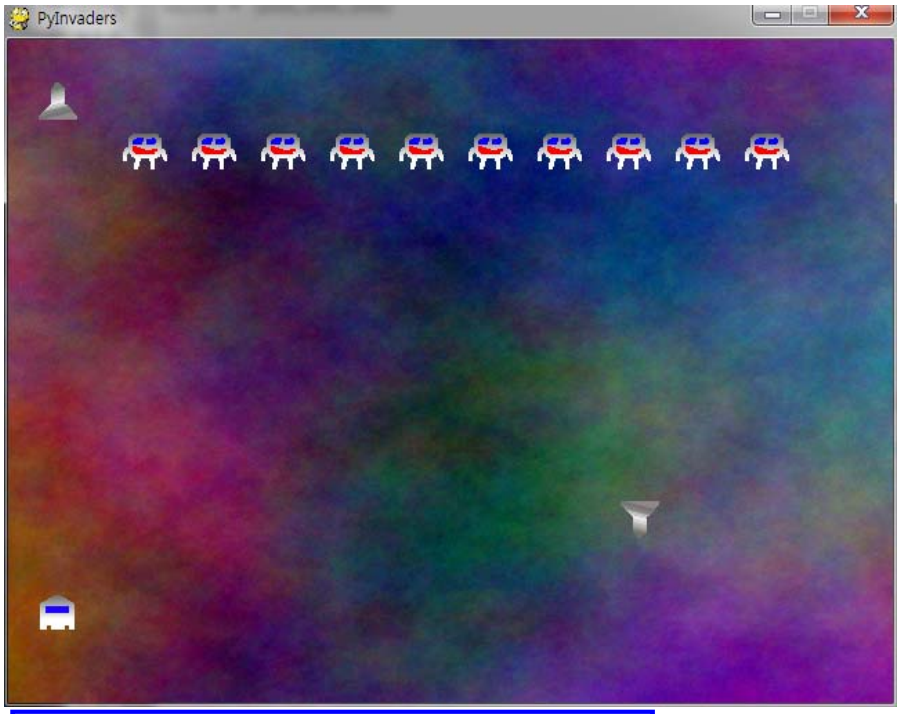
게임 예: PyInvader

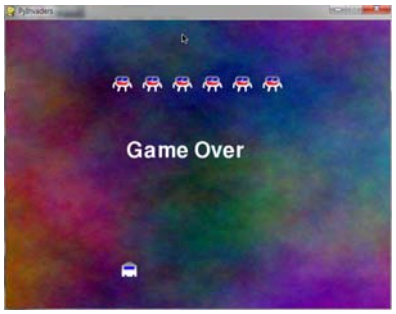
순천향대학교 컴퓨터공학과
이 상 정

학습 내용

- PyInvader게임 사례 소개
  - Space Invader 모방 게임
  - 우주선과 침입자가 서로 미사일 발사
- 스프라이트 작성
  - 파이게임의 스프라이트 사용하지 않고 직접 스프라이트 클래스 정의
  - 충돌 함수 작성

PyInvader





게임 예: PyInvader

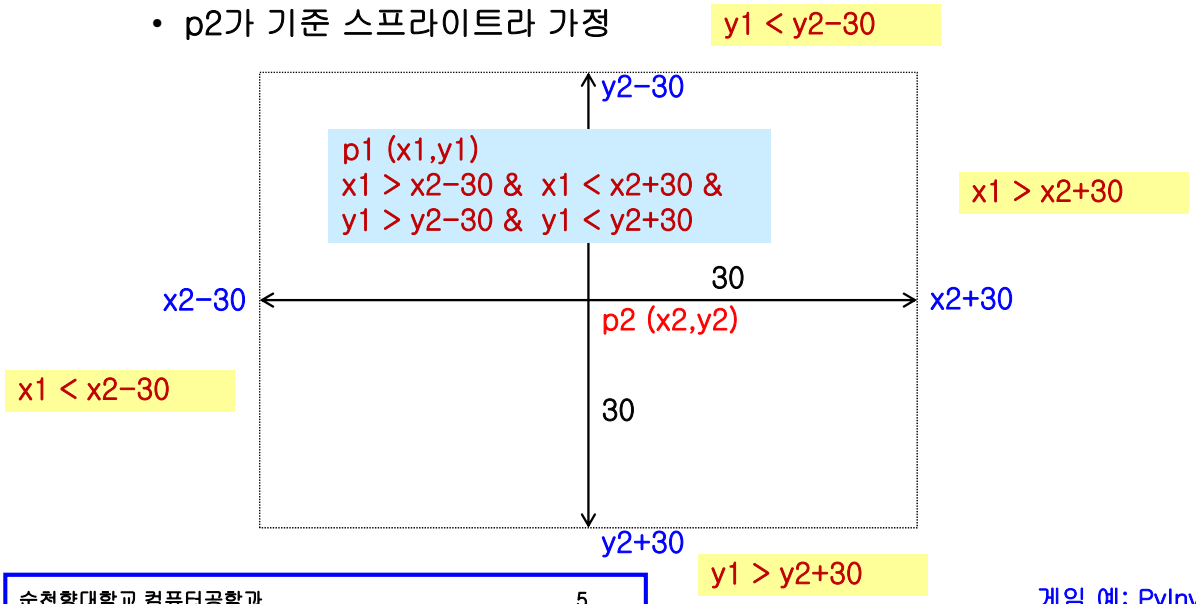
스프라이트 클래스

- 스프라이트 정의 클래스, Sprite
  - 생성자
    - 스프라이트의 이미지, 초기 위치 지정
  - set_position() 메소드
    - 스프라이트 이동 위치 지정
  - render() 메소드
    - 이미지를 스크린에 그리기

```
# 스프라이트 클래스
class Sprite:
        # 생성자, 블록의 위치와 이미지 파일을 인수로 전달
    def __init__(self, xpos, ypos, filename):
            self.x = xpos
            self.y = ypos
            self.bitmap = pygame.image.load(filename)
            self.bitmap.set_colorkey(black)
        # 스프라이트의 위치를 지정
    def set_position(self, xpos, ypos):
            self.x = xpos
            self.y = ypos
    # 이미지를 스크린에 그리기
    def render(self):
            screen.blit(self.bitmap, (self.x, self.y))
```

충돌 감지

- 기준 스프라이트로 부터 +30, -30 픽셀 범위 안에 있으면 충돌(collision)로 인식
  - 예: p1 (x1,y1)과 p2 (x2,y2) 충돌 감지
    - p2가 기준 스프라이트라 가정



충돌 함수

```
# 충돌 감지 함수
# (x1,y1)이 (x2,y2)의 사방에서 +30-,-30 픽셀 범위 안에 들어 오면 충돌 감지
def Collision(x1, y1, x2, y2):
    if (x1 > x2-30) and (x1 < x2+30) and (y1 > y2-30) and (y1 < y2+30):
        return True
    else:
        return False
```

키보드 누름 반복

- 키보드 누르고 있는 동안 우주선의 부드러운 이동(연속적인 이동)을 위해 키보드 누름 반복 설정
  - pygame.key.set_repeat(delay, interval): return None
  - 일정 간격으로 계속 pygame.KEYDOWN 이벤트 발생
  - delay는 처음 키보드 눌렀을 때 pygame.KEYDOWN 이벤트 발생 지연 (ms)
  - interval은 키보드 누르고 있는 동안 pygame.KEYDOWN 이벤트를 보내는 간격 지정 (ms)

  - 예,
    pygame.key.set_repeat(1, 1)

스프라이트 생성

- 우주선,침입자(10개), 우주선/침입자 미사일 스프라이트 생성

```
# 우주선 스프라이트 생성
hero = Sprite(20, 400, 'data/hero.bmp')
# 우주선 미사일 스프라이트 생성
ourmissile = Sprite(0, 480, 'data/heromissile.bmp')

# 침입자 스프라이트 리스트 생성
enemies = []
x = 0
# 50 픽셀 간격으로 한 행에 10개 생성
for count in range(10):
    enemies.append(Sprite(50 * x + 50, 50, 'data/baddie.bmp'))
    x += 1
# 침입자 미사일 스프라이트 생성
enemymissile = Sprite(0, 480, 'data/baddiemissile.bmp')
```

우주선, 미사일 이동 및 그리기

```
        .....
        if event.type == pygame.KEYDOWN:
                # 좌우 화살표 키 누르면 우주선 좌우 이동
                if event.key == pygame.K_RIGHT and hero.x < 590:
                        hero.x += 5
                if event.key == pygame.K_LEFT and hero.x > 10:
                        hero.x -= 5
                # 스페이스 키 누르면 우주선 미사일 초기값 지정
                if event.key == pygame.K_SPACE:
                        ourmissile.x = hero.x
                        ourmissile.y = hero.y
        .....
        # 우주선 미사일 위로 이동
        if ourmissile.y < 479 and ourmissile.y > 0:  # 스크린 범위 내이면
                ourmissile.render()     # 우주선 미사일 그리기
                ourmissile.y -= 5       # 위로 이동

.....
# 우주선 그리기
hero.render()
.....
```


침입자 이동 및 그리기 (1)

```
# 침입자 좌우 이동 속도
enemyspeed = 3
.....
# 침입자 x 좌표값 갱신
for count in range(len(enemies)):
        enemies[count].x += enemyspeed

# 오른쪽 침입자 스크린 오른쪽 끝 도달 시
if enemies[len(enemies)-1].x > 590:
        enemyspeed = -3    # x 좌표 방향 전환
        for count in range(len(enemies)):
                # 모든 침입자 y 좌표 아래 이동
                enemies[count].y += 5
# 왼쪽 침입자 스크린 왼쪽 끝 도달 시
if enemies[0].x < 10:
        enemyspeed = 3      # x 좌표 방향 전환
        for count in range(len(enemies)):
                enemies[count].y += 5    # 모든 침입자 y 좌표 아래 이동
```

침입자 이동 및 그리기 (2)

```
            # 침입자 미사일이 아래 도달하고, 침입자가 남아 있으면
            if enemymissile.y >= 480 and len(enemies) > 0:
                    # 임의의 침입자 위치로 미사일 초기 위치 지정
                    enemymissile.x = enemies[random.randint(0, len(enemies) - 1)].x
                    enemymissile.y = enemies[0].y

            # 침입자 미사일 그리기
            enemymissile.render()

            # 침입자 미사일 아래로 이동
            enemymissile.y += 5

# 침입자 그리기
    for count in range(len(enemies)):
            enemies[count].render()
 .........
```


미사일 충돌

```
            .............
            # 우주선과 침입자 미사일 충돌 시
            if Collision(hero.x, hero.y, enemymissile.x, enemymissile.y):
                    game_over = True      # 게임 종료

            # 우주선 미사일과 침입자 충돌 시
            for count in range(0, len(enemies)):
                    if Collision(ourmissile.x, ourmissile.y, enemies[count].x, enemies[count].y):
                            del enemies[count]      # 침입자 제거
                            break       # for 문 탈출

            # 모든 침입자 제거시
            if len(enemies) == 0:
                    game_over = True     # 게임 종료
             .............
```

PyInvader 코드

```
import pygame
import random

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

# 스프라이트 클래스
class Sprite:
    # 생성자, 블록의 위치와 이미지 파일을 인수로 전달
    def __init__(self, xpos, ypos, filename):
        self.x = xpos
        self.y = ypos
        self.bitmap = pygame.image.load(filename)
        self.bitmap.set_colorkey(black)
    # 스프라이트의 위치를 지정
    def set_position(self, xpos, ypos):
        self.x = xpos
        self.y = ypos
    # 이미지를 스크린에 그리기
    def render(self):
        screen.blit(self.bitmap, (self.x, self.y))

# 충돌 감지 함수, (x1,y1)이 (x2,y2)의 사방에서
# +30-,-30 픽셀 범위 안에 들어 오면 충돌 감지
def Collision(x1, y1, x2, y2):
    if (x1 > x2-30) and (x1 < x2+30) and
            (y1 > y2-30) and (y1 < y2+30):
        return True
    else:
        return False

pygame.init()

# 윈도우 설정
screen = pygame.display.set_mode((640,480))
pygame.display.set_caption('PyInvaders')
# 키보드 누름 반복 설정
pygame.key.set_repeat(1, 1)
# 텍스트 폰트 생성
font = pygame.font.Font('freesansbold.ttf', 36)
# 배경 이미지 적재
backdrop =
    pygame.image.load('data/backdrop.bmp')
```

```
# 우주선 스프라이트 생성
hero = Sprite(20, 400, 'data/hero.bmp')
# 우주선 미사일 스프라이트 생성
ourmissile = Sprite(0, 480, 'data/heromissile.bmp')

# 침입자 스프라이트 생성
enemies = []
x = 0
# 50 픽셀 간격으로 한 행에 10개 생성
for count in range(10):
    enemies.append(Sprite(50 * x + 50, 50,
                          'data/baddie.bmp'))
    x += 1

# 침입자 미사일 스프라이트 생성
enemymissile = Sprite(0, 480,
                      'data/baddiemissile.bmp')

# 침입자 좌우 이동 속도
enemyspeed = 3

clock =pygame.time.Clock()

quit = False
game_over = False
```

```
while quit != True:
    # 스크린에 배경 이미지 그리기
    screen.blit(backdrop, (0, 0))

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            quit = True
        if event.type == pygame.KEYDOWN:
            # 좌우 화살표 키 누르면 우주선 좌우 이동
            if event.key == pygame.K_RIGHT and
                                    hero.x < 590:
                hero.x += 5
            if event.key == pygame.K_LEFT and
                                    hero.x > 10:
                hero.x -= 5
            # 스페이스 키 누르면 우주선 미사일 초기값
            # 지정
            if event.key == pygame.K_SPACE:
                ourmissile.x = hero.x
                ourmissile.y = hero.y

    if not game_over:  # 게임 종료가 아니면
        # 침입자 x 좌표값 갱신
        for count in range(len(enemies)):
            enemies[count].x += enemyspeed
```

```
    # 오른쪽 침입자 스크린 오른쪽 끝 도달 시
    if enemies[len(enemies)-1].x > 590:
        enemyspeed = -3    # x 좌표 방향 전환
        for count in range(len(enemies)):
            enemies[count].y += 5
                # 모든 침입자 y 좌표 아래 이동
    # 왼쪽 침입자 스크린 왼쪽끝 도달 시
    if enemies[0].x < 10:
        enemyspeed = 3      # x 좌표 방향 전환
        for count in range(len(enemies)):
            enemies[count].y += 5
                # 모든 침입자 y 좌표 아래 이동

    # 우주선 미사일 위로 이동
    if ourmissile.y < 479 and ourmissile.y > 0:
                            # 스크린 범위 내이면
        ourmissile.render() # 우주선 미사일 그리기
        ourmissile.y -= 5       # 위로 이동

    # 침입자 미사일이 아래 도달하고,
    # 침입자가 남아 있으면
    if enemymissile.y >= 480 and len(enemies)>0:
      # 임의의 침입자 위치로 미사일 초기 위치 지정
        enemymissile.x =
enemies[random.randint(0, len(enemies) - 1)].x
        enemymissile.y = enemies[0].y
```

```
    # 우주선과 침입자 미사일 충돌 시
    if Collision(hero.x, hero.y, enemymissile.x,
enemymissile.y):
        game_over = True      # 게임 종료
    # 우주선 미사일과 침입자 충돌 시
    for count in range(0, len(enemies)):
        if Collision(ourmissile.x, ourmissile.y,
enemies[count].x, enemies[count].y):
            del enemies[count]      # 침입자 제거
            break       # for 문 탈출
    # 모든 침입자 제거시
    if len(enemies) == 0:
        game_over = True     # 게임 종료
    # 침입자 미사일 그리기
    enemymissile.render()
    # 침입자 미사일 아래로 이동
    enemymissile.y += 5

# 게임 종료 이면 Game Over 출력
if game_over:
    text = font.render("Game Over", 1, white)
    screen.blit(text, [200,200])

# 우주선 그리기
hero.render()
```

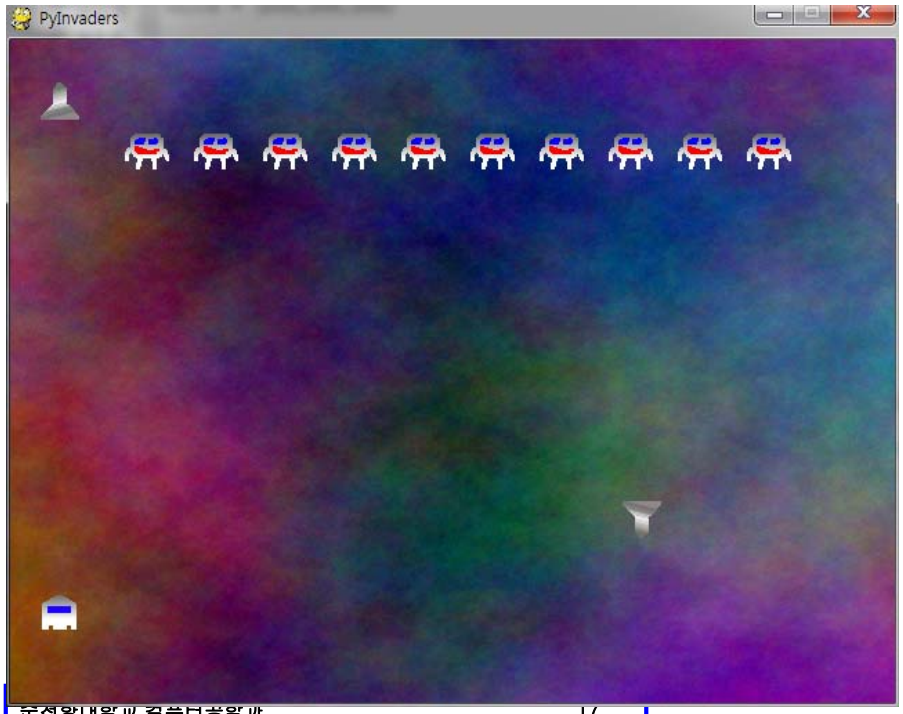


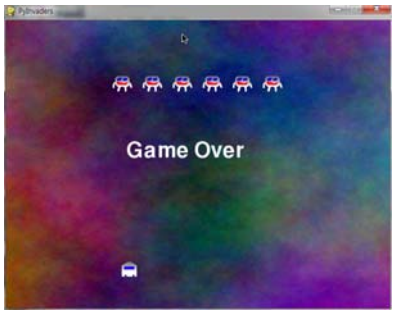
```
    # 침입자 그리기
    for count in range(len(enemies)):
        enemies[count].render()

    clock.tick(70)  # 프레임 속도(전체 스프라이트 이동 속도)
    pygame.display.update()

pygame.quit()
```

시험주행





게임 예: PyInvader

과제

1. 지금까지 배운 내용을 활용한 텀 프로젝트 부분 프로그램 작성
   - 이 전 텀 프로젝트 부분 과제에 추가된 내용 설명
   - 지금까지 배운 내용 활용한 부분 프로그램 소스
   - 실행 결과

게임 예: PyInvader