컴퓨터과학과 코딩
게임 만들기(팀 프로젝트) 레포트

<목차>

<조원>
202312720 안수빈
202312732 오성진
202312854 허예린

<1. 회의록>

[1차 회의]

조원: 안수빈, 오성진, 허예린
일시: 5월 23일 (화) 4시
장소: 공대 7호관 무한상상실

<회의 내용>

1) 게임 정하기
그동안 배운 파이썬 내용과 프로젝트 II의 내용을 기반으로 어떤 게임을 구축하고 싶은지에 대한 이야기를 나누었고, 구축하고 싶은 게임으로
1. 쿠키런
2. 테트리스
3. 유성 피하기
4. 리듬 게임
이 후보로 나왔고, 만들 게임의 난이도와 실현 가능성, 각 게임과 관련된 코드들을 찾아보는 과정을 거쳤습니다. 그 결과 '유년 시절에 많이 하던 쿠키런과 비슷한 게임을 구축해보자'라는 결론이 나왔습니다. 쿠키런 게임에서 아이디어를 얻어와 러너가 장애물을 뛰어넘으며 아이템을 획득하여 점수를 얻는 프로그램을 만들어보자는 목표를 설정하였습니다.

2) 다음 회의 때 회의할 안건
게임 프로그램 구축의 목표로 잡았던 요소들이 들어간 오픈 소스를 미리 조사해오기로 하였습니다. 다음 회의 때 서로 조사해온 자료들을 비교해보며 게임의 기반이 될 코드를 선정하고, 게임에 들어갈 세부 요소들을 선정하기로 하였습니다.

3) 역할 분담
게임 구축에 대한 역할 분담: 아직 기반이 될 소스 코드를 찾지 못하였으므로 각자 조사해본 후에 다음 회의 때 정하기로 하였습니다.

레포트에 대한 역할 분담:
안수빈: 회의록, 역할 분담, 참고 자료, 레포트의 세부 요소 등 작성
오성진: 회의록을 기반으로 한 게임 선택이유, 구축 방식, 이번 학기에 배운 것들을 어떻게 적용하였는지 등 작성
허예린: 레포트 정리와 코드 설명 작성

[2차 회의]

조원: 안수빈, 오성진, 허예린
일시: 5월 25일 (목) 1시
장소: 중앙도서관 3층 스터디룸

<회의 내용>
1) 게임에 들어갈 세부 요소 정하기
각자 찾아온 자료에서 겹치는 오픈소스가 있어 그 자료를 게임의 기반이 될 코드로 선정하였습니다. 또한 게임에 추가하고 싶은 기능과 세부 요소에 관하여 이야기를 나누었고, 게임이 '컴퓨터 인공지능학부'와 관련이 있으면 좋겠다는 의견이 나왔습니다. 그리하여 게임을 플레이할 때 '컴퓨터 인공지능학부'가 연상될 수 있는 요소들을 찾아서 러너와 장애물, 배경, 아이템 등에 적용하기로 하였습니다.

[세부 요소 추가]
배경: 강의실 배경
캐릭터: 교수님의 얼굴을 합성한 졸라맨
장애물: 리눅스 펭귄
아이템: 커피, 핫식스

또한 기능의 추가로는 쿠키런을 패러디하여 만들었다는 느낌을 받을 수 있도록 쿠키런의 배경음악을 깔고, 추가적인 기능을 더하여 게임 플레이에 재미를 더할 수 있도록 만들기로 하였습니다.
[기능 추가]
1) 러너가 장애물을 넘으면서 아이템을 획득하여 점수를 올리고, 아이템을 먹으면 목숨이 하나 생기는 기능 (아이템을 여러 개 획득해도 추가 목숨은 한 개, 중첩되지 않음)
2) 게임을 플레이할 때 러너가 뛰거나 죽을 때 사운드와 배경음악, 아이템 획득할 때 사운드
3) 장애물의 종류와 크기를 다르게 하고, 장애물의 높이에 따라 러너의 뛰는 높이도 다르게 조종할 수 있는 기능
4) 러너가 달리다가 아이템을 처음 획득하게 되면 러너의 색이 초록색으로 바뀌고, 색이 바뀐 채로 달리다 장애물과 충돌을 하게 되면 한번 살아난 뒤 원래 색으로 돌아오는 기능, 살아난 후 원래 색으로 달리다 아이템을 획득하게 되면 색이 초록색으로 다시 바뀌는 기능
5) 게임이 시작하기 전 카운트 다운 기능

2) 역할 분담
앞서 회의한 내용을 기반으로 오픈코드에 각자 역할을 맡은 부분에 대한 코드를 써오기로 하였습니다.
안수빈- 러너의 이미지 합성, 배경, 장애물 이미지 바꾸기와 장애물의 높이에 따라 러너의 뛰는 높이 다르게 하는 기능 추가
오성진- 게임 기본 배경음악, 러너가 뛸 때, 죽을 때, 아이템 획득 사운드 기능 추가, 게임 시작 전 카운트 다운 기능 추가
허예린- 러너가 뛰면서 아이템을 획득하여 점수를 올리는 기능 추가, 러너가 아이템을 획득한 후에 러너의 색이 초록색으로 변하게 하는 기능, 러너가 충돌하고 나서도 다시 한번 살아날 수 있게 하는 기능 추가

[3차 회의]

조원: 안수빈, 오성진, 허예린
일시: 6월 1일 (목) 2시
장소: 중앙도서관 2층 스터디룸

<회의 내용>
1) 각자 해온 부분 실행시켜 보면서 서로 확인
각자 해온 부분을 실행시켜 보고, 실행한 프로그램에 대한 피드백을 서로 나누었습니다. 기반으로 잡았던 코드에 사운드와 아이템 효과를 넣는 것에 어려움이 있어 배경과 장애물 등을 넣어 놓은 코드에 기능을 추가하는 방법으로 코드를 수정하였습니다.

3) 세부 역할 분담
안수빈- 레포트의 회의록, 참고 자료, 역할 분담 등 세부 요소 작성
오성진- 레포트의 아이디어 논의 과정 작성
허예린- 각자 작성해온 코드 모으기, 레포트의 코드 설명 주석 작성

[4차 회의]

조원: 안수빈, 오성진, 허예린
일시: 6월 8일 (목) 2시
장소: 중앙도서관 2층 스터디룸

<회의 내용>
1) 효과음 추가 상의
합친 코드를 실행해보고 난 후에 효과음을 추가하기 위해 상의하는 시간을 가졌습니다. 코드에서 추가한 효과음은 러너가 아이템을 획득하는 소리, 러너가 장애물을 만났을 때 죽는소리를 추가하였습니다.

2) 코드와 레포트 수정 후 제출
상의한 효과음을 코드에 추가하였고, 그에 따라 레포트를 수정하는 과정을 거친 후에 과제를 제출하였습니다.

<2. 아이디어 논의 과정>

[목표 설정 과정]

무슨 게임을 만들면 좋을지에 대해 회의하다 '우리가 어렸을 때 즐겨 했던 게임을 우리가 직접 코딩하여 만들어 보면 좋겠다'라는 의견이 있었고, 그리하여 유년 시절에 많이 하던 게임을 찾아보며 회의 한 결과 쿠키런과 비슷한 게임을 만들기로 결정하였습니다. 그 결과 쿠키런 게임에서 아이디어를 빌려 캐릭터가 장애물을 뛰어넘으며 아이템을 획득하여 점수를 얻는 프로그램을 만들어 보자는 목표를 설정하게 되었습니다.

[아이디어 발전 과정]

그 후에 '쿠키런에서 나오는 쿠키 캐릭터와 장애물 아이템을 어떻게 해야 효과적으로 사용할 수 있고, 무엇으로 대체하면 좋을까?'에 대한 고민이 있었습니다. 그러한 고민 끝에 '우리가 컴퓨터인공지능학과의 학생이니 우리 학과에서 흔히 볼 수 있고, 공감대를 형성하기 쉬운 이미지를 사용하자'라는 의견이 나왔고, 그 의견을 적극 반영하여 우리가 매일 가는 강의실의 이미지를 배경화면으로 넣었고, 우리가 코딩할 때 즐겨 마시는 커피와 핫식스를 아이템으로 만들었으며, 2학년 때 우리가 해야 할 리눅스의 펭귄을 장애물로 설정하고, 캐릭터로는 파이썬을 가르쳐주시는 교수님의 모습이 들어가면 좋지 않을까 싶어 교수님의 허락을 받고 교수님의 사진을 붙인 졸라맨을 사용하게 되었습니다. 또한 쿠키런에서 아이디어를 얻어왔다는 느낌을 플레이하는 사용자에게 주기 위하여 쿠키런 게임의 기본 배경음악을 삽입하였습니다.

[기능 발전 과정]

쿠키런을 모티브로 해서 만든 게임 이긴 하지만, 아이템 효과까지 모든 것이 똑같으면 게임을 만들었다고 하기보단 쿠키런이라는 게임 그 자체를 따라 한 것과 다름이 없기에, 쿠키런과는 명백히 다른 게임이라는 것임을 알리고 그것에 관한 차이점을 두기 위해서 러너가 아이템을 획득하면 러너의 색깔이 바뀌고, 아이템을 활용해서 러너의 목숨을 획득하는 방식으로 바꿔보았습니다.

<3. 코드 설명>

```
import pygame
import sys
import time
import random

#pygame 초기화, screen (폭, 높이)설정
pygame.init()
pygame.display.set_caption('Jumping game')
MAX_WIDTH = 800
MAX_HEIGHT = 400

start_num = 0 #점프 효과음 제어 변수

# 펭귄 생성 간격 및 위치
PENGUIN_INTERVAL = 2000  # 펭귄 생성 간격 (2초)
PENGUIN_DISTANCE = 400  # 펭귄 간의 거리 (300px)

# sound
pygame_bg_music = pygame.mixer.Sound("bg_music.mp3")              #배경음악
pygame_count_down_sound = pygame.mixer.Sound("count_down.wav")  #카운트다운 효과음
pygame_jump_sound = pygame.mixer.Sound("jump.ogg")                #점프 효과음
pygame_obstacle1_sound = pygame.mixer.Sound("obstacle1.ogg")       #장애물 효과음
pygame_item_sound = pygame.mixer.Sound("item.ogg")                #아이템 효과음
pygame_obstacle2_sound = pygame.mixer.Sound("obstacle2.ogg")     #장애물 효과음(아이템 사용중)
pygame_dead_sound = pygame.mixer.Sound("dead.ogg")                 #죽음 효과음
pygame_count_down_sound.play()                                    #카운트다운 효과음 실행

screen = pygame.display.set_mode((MAX_WIDTH, MAX_HEIGHT))   #디스플레이 설정
fps = pygame.time.Clock()   #초당프레임 설정을 위한 Clock()함수 저장

# 카운트다운 텍스트 출력하기
font = pygame.font.SysFont(None, 100)   # 글자 크기 및 폰트
counter = 3                             # 카운트할 숫자
text = font.render(str(counter), True, (0, 128, 0)) #카운트 글자 설정(초록)

timer_event = pygame.USEREVENT+1     #타이머 이벤트의 ID
pygame.time.set_timer(timer_event, 1000)     #1초로 타이머 세팅

run = True  #초기 상태 True로 달리기 x
```

```
while run:  #run이 참일 동안
    fps.tick(60)    #초당프레임(60)
    for event in pygame.event.get():
        if event.type == pygame.QUIT:   #나가기 누를시
            pygame.quit()   #pygame 종료
            sys.exit()  #프로그램 종료
        if event.type == timer_event: #타이머 이벤트 실행시
            counter -= 1    #카운트 숫자 -1
            text = font.render(str(counter), True, (0, 128, 0)) #카운트 숫자 출력
            if counter == 0:   # counter=0일 때, run=False가 되어서 시작
                pygame.time.set_timer(timer_event, 0)
                run = False #달리기 시작

    screen.fill((255, 255, 255))    #배경 설정(흰색)
    text_rect = text.get_rect(center = screen.get_rect().center)    #텍스트 위치 저장
    screen.blit(text, text_rect)    #저장된 위치에 글자 출력
    pygame.display.flip()   #화면 전체 업데이트

def main():
    screen = pygame.display.set_mode((MAX_WIDTH, MAX_HEIGHT))   #디스플레이 설정
    fps = pygame.time.Clock()   #초당프레임 설정을 위한 Clock()함수 저장

    # 배경 이미지 로드
    background_img = pygame.image.load('교실 배경화면.png') #배경 사진
    background_img = pygame.transform.scale(background_img, (MAX_WIDTH, MAX_HEIGHT))
  #배경 이미지 고정

    #캐릭터 생성, 초기 설정
    imgRun1 = pygame.image.load('합성사진1.png')    #기본 캐릭터 사진
    imgRun2 = pygame.image.load('합성사진2.png')
    imgRuner1 = pygame.image.load('합성사진10.png') #아이템 획득 후 사진
    imgRuner2 = pygame.image.load('합성사진20.png')
    run_width = imgRun1.get_size()[0]   #캐릭터 x축 크기
    run_height = imgRun1.get_size()[1]  #캐릭터 y축 크기
    run_bottom = MAX_HEIGHT - run_height  #캐릭터 y축 위치 계산
    run_x = 50  #캐릭터 x축 위치 50
    run_y = run_bottom  #캐릭터 y축 위치 저장
    jump_top = 200  #최대 점프 높이
    leg_swap = True  #공룡 사진 변경 위한 변수
    is_bottom = True   #달리고 있는지 알려주는 변수
    is_go_up = False #위로 올라가고 있는지 알려주는 변수->초기 False로 안올라감
```

```
    #장애물 생성, 초기 설정
    imgPenguin1 = pygame.image.load('penguin1.png') #펭귄1 사진
    imgPenguin2 = pygame.image.load('penguin2.png') #펭귄2 사진
    penguin1_height = imgPenguin1.get_size()[1] #펭귄1 y축 크기
    penguin2_height = imgPenguin2.get_size()[1] #펭귄2 y축 크기
    penguin_x1 = MAX_WIDTH + random.randint(300, 600)  # 펭귄1이 나오는 x축 위치 랜덤으로
설정
    penguin_x2 = penguin_x1 + PENGUIN_DISTANCE  # 펭귄2는 펭귄1과 일정한 거리를 유지
    penguin_y1 = MAX_HEIGHT - penguin1_height   #장애물 y축 위치 계산, 저장
    penguin_y2 = MAX_HEIGHT - penguin2_height

    #아이템 생성, 초기 설정
    imgItem = pygame.image.load('coffee.png')   #아이템 사진
    imgItem = pygame.transform.scale(imgItem, (50, 50)) #아이템 크기 설정, 고정(50X50)
    item_width = imgItem.get_size()[0]  #아이템 x축 크기
    item_height = imgItem.get_size()[1] #아이템 y축 크기
    item_x = MAX_WIDTH - item_width #아이템 x축 위치
    item_y = MAX_HEIGHT - random.randint(100, 300)  #아이템 y축 위치

    run_state = "nomal" #캐릭터 초기 상태

    game_over = False  # 게임 오버 상태
    score = 0  # 점수
    start_time = time.time()  # 게임 시작 시간
    end_time = 0  # 게임 종료 시간
    jump_count = 0  # 스페이스바를 누른 횟수
    jump_top1 = 130  # 첫 번째 점프 높이
    jump_top2 = 110  # 두 번째 점프 높이

    last_penguin_spawn_time = time.time()  # 이전 펭귄 생성 시간

    penguin1_visible = True  # 펭귄 1의 가시성
    penguin2_visible = True  # 펭귄 2의 가시성

    pygame_bg_music.play(-1)     #배경음악 시작(반복)

    num = 0  #점프 효과음 실행 횟수 제어 변수 생성

    while True:
        screen.blit(background_img, (0, 0))  #스크린 설정(배경 사진, (x, y))
```

```
        #사용자 입력에 따른 실행
        for event in pygame.event.get():     #이벤트 동작을 했을시
            if event.type == pygame.QUIT:    #나가기 클릭시
                pygame.quit()    #pygame 종료
                sys.exit()  #프로그램 종료
            elif event.type == pygame.KEYDOWN:  #키보드를 눌렀다 뗄 때 실행
                if event.key == pygame.K_SPACE and is_bottom and not game_over: #게임오버
가 아니고 달리는 중일 때 스페이스바를 누를시
                    is_go_up = True #올라가기 시작
                    is_bottom = False    #달리기 멈춤
                    jump_count += 1 #점프 횟수 +1
                elif event.key == pygame.K_SPACE and game_over: #게임오버 후 스페이스바 누를
시
                    main()  #게임 다시 시작

        #캐릭터 상태에 따른 실행
        if not game_over:    #게임오버가 아닐시
            if is_go_up: #올라가는 중이면
                if num == 0 : #num=0일때
                    pygame_jump_sound.play()     #점프 효과음 실행
                    num = num + 1    #점프 효과음 제어 변수 +1

                if jump_count == 1: #점프 횟수가 1일시
                    jump_top = jump_top1     #최대 점프 높이 130으로 변경
                elif jump_count >= 2:    #점프 횟수가 2이상일시
                    jump_top = jump_top2     #최대 점프 높이 110으로 변경
                run_y -= 10.0    #캐릭터 위로 10만큼 올라감
            elif not is_go_up and not is_bottom: #올라기지도 바닥에 있지도 않으면
                run_y += 10.0    #캐릭터 아래로 10만큼 내려감
                num = 0  #점프 효과음 제어변수 초기화

            if is_go_up and run_y <= jump_top:     #올라가는 중이고 최대 높이 이하면
                is_go_up = False #올라가는 변수 거짓->내려가기 시작

            if not is_bottom and run_y >= run_bottom: #바닥에 있지 않고 최소 높이 이상이면
                is_bottom = True     #달리기 시작
                run_y = run_bottom  #캐릭터 y축 위치 최소 높이로 변경

            current_time = time.time()  #현재 시간 기록
            time_since_last_spawn = current_time - last_penguin_spawn_time  #펭귄 생성 시간
```

간격 계산

```
            if time_since_last_spawn >= PENGUIN_INTERVAL:   #펭귄 생성 시간 간격 >= 2초이면
                penguin_x1 = MAX_WIDTH + random.randint(300, 600)  # 펭귄 1의 위치를 랜덤으로 설정
                penguin_x2 = penguin_x1 + PENGUIN_DISTANCE  # 펭귄 2는 펭귄 1과 일정한 거리를 유지
                last_penguin_spawn_time = current_time  #마지막 펭귄 생성 시간 = 현재 시간

            penguin_x1 -= 12    #펭귄 왼쪽으로 12만큼 이동
            penguin_x2 -= 12
            if penguin_x1 <= 0: #펭귄 x축 위치 <= 0 이면
                penguin_x1 = MAX_WIDTH  #펭귄 x축 위치 오른쪽 끝으로
            if penguin_x2 <= 0:
                penguin_x2 = MAX_WIDTH

            item_x -= 12    #아이템 왼쪽으로 12만큼 이동
            run_rect = pygame.Rect(run_x, run_y, run_width, run_height) #캐릭터 위치, 크기 저장
            item_rect = pygame.Rect(item_x, item_y, item_width, item_height) #아이템 위치, 크기 저장
            if item_rect.colliderect(run_rect): #충돌하면
                run_state = "power" #캐릭터 상태 변경(power)
                pygame_item_sound.play()   #아이템 효과음 실행
                item_x = MAX_WIDTH - item_width #아이템 x축 위치 재설정
                item_y = MAX_HEIGHT - random.randint(100, 300)  #아이템 y축 위치 재설정
            if item_x <=0: #아이템 왼쪽 끝으로 가면
                item_x = MAX_WIDTH - item_width #아이템 x축 위치 재설정
                item_y = MAX_HEIGHT - random.randint(100, 300)  #아이템 y축 위치 재설정

            # 펭귄 1과 충돌 검사
            #충돌할시(펭귄1 보임 & 캐릭터 x축 위치+캐릭터 가로 크기>펭귄1 x축 위치 & 캐릭터 x축 위치<펭귄1 x축 위치+펭귄1 가로 크기 & 캐릭터 y축 위치+캐릭터 세로 크기>펭귄1 y축 위치 & 캐릭터 y축 위치<펭귄1 y축 위치+펭귄1 높이면)

            if penguin1_visible and run_x + imgRun1.get_width() > penguin_x1 and run_x < penguin_x1 + imgPenguin1.get_width() and run_y + imgRun1.get_height() > penguin_y1 and run_y < penguin_y1 + penguin1_height:
                if run_state == "power":    #캐릭터 상태 power면
                    pygame_obstacle2_sound.play()   #장애물 효과음(아이템 사용중) 실행
                    run_state = "nomal" #캐릭터 상태 nomal로 변경
```

```
                penguin_x1 -= run_width+30  #펭귄1 왼쪽으로 run_width+30만큼 이동
                item_x -= run_width+30  #아이템 왼쪽으로 run_width+30만큼 이동
            else:   #캐릭터 상태 nomal이면
                game_over = True     #게임오버
                pygame_obstacle1_sound.play() #장애물 효과음 실행
                pygame_bg_music.stop()         #배경음악 멈춤
                pygame_dead_sound.play()       #죽음 효과음 실행
                end_time = time.time()  # 게임 종료 시간 기록

        else:   #충돌하지 않을시
            #펭귄1 x축 위치+펭귄1 가로 크기<캐릭터 x축 위치 & 올라가는 중이 아님 & 펭귄1 보이는 중이면
            if penguin_x1 + imgPenguin1.get_width() < run_x and not is_go_up and penguin1_visible:
                score += 5  #점수 5점 증가
                penguin_x1 = penguin_x2 + PENGUIN_DISTANCE  # 펭귄 1이 지나간 후 새로운 펭귄 1의 위치 설정
                penguin1_visible = True  # 펭귄 1 다시 보이도록 설정

            # 펭귄 2와 충돌 검사
            #펭귄2 보임 & 캐릭터 x축 위치+캐릭터 가로 크기>펭귄2 x축 위치 & 캐릭터 x축 위치<펭귄2 x축 위치+펭귄2 가로 크기 & 캐릭터 y축 위치+캐릭터 세로 크기>펭귄2 y축 위치 & 캐릭터 y축 위치<펭귄2 y축 위치+펭귄2 높이면
            if penguin2_visible and run_x + imgRun1.get_width() > penguin_x2 and run_x < penguin_x2 + imgPenguin2.get_width() and run_y + imgRun1.get_height() > penguin_y2 and run_y < penguin_y2 + penguin2_height:
                if run_state == "power":     #캐릭터 상태 power면
                    pygame_obstacle2_sound.play()   #장애물 효과음(아이템 사용중) 실행
                    run_state = "nomal" #캐릭터 상태 nomal로 변경
                    penguin_x2 -= run_width+30  #펭귄2 왼쪽으로 run_width+30만큼 이동
                    item_x -= run_width+30  #아이템 왼쪽으로 run_width+30만큼 이동
                else:
                    game_over = True     #게임오버
                    pygame_obstacle1_sound.play() #장애물 효과음 실행
                    pygame_bg_music.stop()         #배경음악 멈춤
                    pygame_dead_sound.play()       #죽음 효과음 실행
                    end_time = time.time()  # 게임 종료 시간 기록
            else:   #충돌하지 않을시
                #펭귄2 x축 위치+펭귄2 가로 크기<캐릭터 x축 위치 & 올라가는 중이 아님 & 펭귄2 보이는 중이면
                if penguin_x2 + imgPenguin2.get_width() < run_x and not is_go_up and
```

```
penguin2_visible:
                    score += 5 #점수 5점 증가
                    penguin_x2 = penguin_x1 + PENGUIN_DISTANCE  # 펭귄 2가 지나간 후 새로
운 펭귄 2의 위치 설정
                    penguin2_visible = True  # 펭귄 2 다시 보이도록 설정


            # 펭귄 그리기
            if penguin1_visible:     #펭귄1 가시성==True면
                screen.blit(imgPenguin1, (penguin_x1, penguin_y1))
            if penguin2_visible:     #펭귄2 가시성==True면
                screen.blit(imgPenguin2, (penguin_x2, penguin_y2))

            #아이템 그리기
            screen.blit(imgItem, (item_x, item_y))

        #캐릭터 액션(leg_swap이 True면->1번 이미지 사용), 그리기
        if run_state == "nomal":     #캐릭터 상태가 nomal이면->imgRun 이미지 사용
            if leg_swap:
                screen.blit(imgRun1, (run_x, run_y))
                leg_swap = False
            else:
                screen.blit(imgRun2, (run_x, run_y))
                leg_swap = True
        else:    #캐릭터 상태가 power이면->imgRuner 이미지 사용, 그리기
            if leg_swap:
                screen.blit(imgRuner1, (run_x, run_y))
                leg_swap = False
            else:
                screen.blit(imgRuner2, (run_x, run_y))
                leg_swap = True

        # 점수 표시
        font = pygame.font.Font(None, 36)    #폰트 설정
        score_text = font.render("Score: {}".format(score), True, (0, 0, 0))  #텍스트 설정(검정색)
        screen.blit(score_text, (10, 10))    #x:10, y:10 위치에 점수 표시

        # 게임이 종료되지 않은 경우 경과 시간 계산 및 표시
        if not game_over:  #게임오버가 아니면
            elapsed_time = int(time.time() - start_time)     #시간 계산
            time_text = font.render("Time: {} sec".format(elapsed_time), True, (0, 0, 0))     #시간
```

```
설정(검정색)
            screen.blit(time_text, (10, 40))     #x:10, y:50위치에 시간 표시


        else:    #게임오버시
            # "Game Over" 텍스트 표시
            game_over_text = font.render("[Game Over]", True, (255, 0, 0))  #게임오버 문자열 설정(빨간색)
            game_over_text = pygame.transform.scale(game_over_text, (game_over_text.get_width() * 2, game_over_text.get_height() * 2))  #게임오버 크기 설정
            game_over_rect = game_over_text.get_rect(center=(MAX_WIDTH // 2, (MAX_HEIGHT // 2) - 50))   #위치 저장(x축 중간, y축 중간-50)
            screen.blit(game_over_text, game_over_rect) #저장된 위치에 문자열 표시

            # 최종 점수 표시
            final_score_text = font.render("Final Score: {}".format(score), True, (0, 0, 0))     #최종점수 설정(검정색)
            final_score_rect = final_score_text.get_rect(center=(MAX_WIDTH // 2, (MAX_HEIGHT // 2) - 10))   #위치 저장(x축 중간, y축 중간-10)
            screen.blit(final_score_text, final_score_rect) #저장된 위치에 문자열 표시

            # 경과 시간 표시
            final_time_text = font.render("Final Time: {} sec".format(int(end_time - start_time)), True, (0, 0, 0))   #최종시간 설정(검정색)
            final_time_rect = final_time_text.get_rect(center=(MAX_WIDTH // 2, (MAX_HEIGHT // 2) + 20))   #위치 저장(x축 중간, y축 중간+20)
            screen.blit(final_time_text, final_time_rect) #저장된 위치에 문자열 표시

            # "Try Again" 텍스트 표시
            try_again_text = font.render("<<<Try Again>>>", True, (0, 0, 0))     #try again 설정(검정색)
            try_again_rect = try_again_text.get_rect(center=(MAX_WIDTH // 2, (MAX_HEIGHT // 2) + 150))   #위치 저장(x축 중간, y축 중간+150)
            screen.blit(try_again_text, try_again_rect) #저장된 위치에 문자열 표시

            # 안내 문구 표시
            instruction_text = font.render("Press SPACE to play again!", True, (0, 0, 0))   #안내문구 설정(검정색)
            instruction_text = pygame.transform.scale(instruction_text, (int(instruction_text.get_width() * 0.8), int(instruction_text.get_height() * 0.8)))    #크기 고정(가로 크기*0.8, 세로 크기*0.8)
```

```
            instruction_rect = instruction_text.get_rect(center=(MAX_WIDTH // 2,
(MAX_HEIGHT // 2) + 180))  #위치 저장(x축 중간, y축 중간+180)
            screen.blit(instruction_text, instruction_rect) #저장된 위치에 문자열 표시

            pygame.display.update() # 사용자지정 화면 업데이트

            while True:
                for event in pygame.event.get():  # 이벤트 동작을 했을시
                    if event.type == pygame.QUIT:  # 나가기 클릭 시
                        pygame.quit()  #pygame 종료
                        sys.exit()  #프로그램 종료
                    elif event.type == pygame.KEYDOWN:  #키 눌렀다 뗄 때
                        if event.key == pygame.K_SPACE:  # 스페이스바 키가 눌렸을 때
                            main() #main함수 실행

        pygame.display.update()
        fps.tick(30)  #초당 30번 화면 업데이트


if __name__ == '__main__':  #프로그램 실행
    main() #main함수 실행
```

<4. AI 도구의 사용>

[202312720 안수빈]
게임에서 기반이 되는 코드에 부가적인 기능들을 추가하기 위해 코드를 작성할 때에는 구글링을 통해 나온 사이트에서 도움을 받았습니다.

[202312732 오성진]- 사운드 담당
게임 속에 소리를 넣는 데에는 챗GPT 같은 AI 도구의 도움은 받지 않았고 오직 구글링을 통해서만 코드를 짰습니다.

[202312854 허예린]
게임 코드를 짤 때 AI 도구의 도움은 받지 않았고 구글링 해서 나오는 사이트와 유튜브 영상들의 도움을 받아 코드를 작성했습니다.

<5. 역할 분담>

[202312720 안수빈]
자료조사, 장애물, 배경, 러너 이미지 추가, 게임이 끝나면 나오는 문구와 스페이스 바를 누르면 다시 게임이 실행되는 기능 추가, 장애물의 높이에 따라 러너의 뛰는 높이를 다르게 조종할 수 있는 기능 추가, 레포트 작성(회의록, 세부 사항 작성)

[202312732 오성진]
자료조사, 게임이 실행되는 동안 나오는 배경음악과 러너가 장애물을 뛰어넘거나 아이템을 획득했을 때, 러너가 죽었을 때, 게임이 시작하기 전 카운트 다운 사운드 기능 추가, 레포트 작성(아이디어 논의 과정)

[202312854 허예린]
자료조사, 러너가 장애물을 피해 뛰면서 아이템을 획득하여 점수를 올리면서 목숨을 획득하는 기능 추가, 각자 쓴 코드를 한곳에 모아 실행되도록 코드를 수정, 러너가 아이템을 획득하면 러너의 색이 초록색으로 바뀌게 하는 기능 추가, 레포트 작성(코드 설명)

<6. 느낀 점>

[202312720 안수빈]
이번 팀플이 대학교에 와서 처음 팀플이었는데 이제까지 해오던 주제와는 다른 게임 만들기 프로젝트여서 잘 할 수 있을까 막막한 기분이 들었다. 하지만 게임 코드를 조사하고, 어떻게 만들어 나갈지 팀원들과 고민하는 과정을 거치면서 게임을 잘 만들 수 있겠다는 생각이 들었다. 파이썬에서 배운 내용들이 실제로는 어떻게 쓰이는지 궁금했었는데 게임을 직접 만들어 나가는 과정을 통해 알게 되었고, 파이썬에 대한 이해도가 높아진 것 같다. 어려웠던 점은 기능을 추가하면서 오류가 나는 부분이었는데, 코드에서 오류를 하나 둘씩 찾아 해결한 후에 제대로 작동하는 것을 보며 뿌듯함을 느끼기도 하였다. 그동안은 사용자의 입장에서 게임을 해왔었는데 이번 프로젝트를 하고 나니 나중에 게임을 할 때 게임의 기능적인 부분이나 디자인적인 부분에 더 관심을 가지게 될 것 같다. 조원들과 역할 분담을 하여 코드를 작성했기 때문에 목표한 데로 게임을 잘 만들 수 있었던 것 같다.

[202312732 오성진]
서로의 맞는 시간을 찾아 회의하고, 서로의 의견을 통합해 나가는 과정이 혼자서 코딩하고 구현할 때에 비해 신경이 더 쓰이며 힘들기도 하였지만, 혼자서 코딩할 때에는 미처 생각하지 못했던 부분을 알 수 있게 되었으며, 팀원들에게 도움을 받은 것 같아 좋았다. 또한 내가 팀원들과 합심하여 게임을 만드는 데에 도움이 될 수 있었다는 점이 놀랍고 신기했으며, 이를 통해 코딩에 대한 흥미가 더욱 커지게 되었다.

[202312854 허예린]
전에는 파이썬이라는 프로그래밍 언어가 가장 배우기 쉬운 언어로 쉬운 만큼 구현할 수 있는 게 한정적이라고 생각했는데 프로그래밍Ⅱ 과제를 하면서 pygame을 알게 되어 게임같이 다양한 기능을 필요로하는 프로그램도 만들 수 있다고 생각이 변하게 되었다. 또한 프로그램을 만들면서 원래 아이템 리스트를 만들어 끝에 가거나 캐릭터와 충돌하면 리스트 요소를 삭제하는 형식으로 하고 싶었지만, 그러면 아이템을 이동시키고 끝까지 갔는지 확인 후 캐릭터와 충돌하는지까지 리스트의 모든 요소를 반복해야 하기 때문에 렉이 걸려 구현하지 못한 점이 아쉬웠다. 시간이 된다면 과제가 아니어도 pygame과 파이썬에 대해 더 공부해보고 원래 하려던 방식으로 바꿔 구현하고 싶다.

<7. 참고 자료>

<게임의 기반이 된 코드의 깃 주소>
깃 주소 : https://github.com/BlockDMask/Python_dinosaur_game

<배경>
https://kor.pngtree.com/freebackground/school-cartoon-classroom-background_1592473.html

<러너>
교수님 얼굴 사진- 교수님께서 제공, 러너-https://www.pngegg.com/ko/png-depla

<장애물>
펭귄 1-https://www.pngwing.com/ko/free-png-iznya
펭귄 2-https://www.pngwing.com/ko/free-png-spdwr

<아이템>
커피 아이템- https://www.flaticon.com/kr/free-icon/ice-coffee_1111100

<배경음악>
https://downloads.khinsider.com/game-soundtracks/album/cookie-run-ovenbreak-ost-complete-collection-2022

<효과음(점프 소리, 부딪히는 소리, 죽는소리)>
https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=kihwankwon1&logNo=110186004256

<카운트 다운 효과음>
https://www.youtube.com/watch?v=dg8GbxWsEIw

[코드를 작성하는 데 참고한 사이트]

<카운트 효과음을 넣는데 참고한 사이트>
https://stackoverflow.com/questions/30720665/countdown-timer-in-pygame
<배경음악과 효과음을 넣는데 참고한 사이트>
https://makerejoicegames.tistory.com/101
<게임 재시작 기능을 넣는데 참고한 영상>
https://www.youtube.com/watch?v=0a2FMgqyy6w

<러너와 장애물의 충돌감지 기능을 넣는데 참고한 사이트>
https://www.jbmpa.com/pygame/13

<충돌 검사 코드, 아이템 크기 설정 코드 참고>
https://github.com/Seulki-You/pygame_cookierun/blob/45c85bc3a266e67bc4735d2ec145ddaf1f57cdc0/cookierun.py#L130

<pygame 이벤트 이해 참고>
https://kkamikoon.tistory.com/entry/PyGame-%EC%9D%B4%EB%B2%A4%ED%8A%B8-pygameKEYDOWN-pygameKEYUP

<코드 설명 참고>
https://www.youtube.com/watch?v=gQo5YLJ0idY