CSED101. Programming & Problem solving
Spring 2023

Programming Assignment #4
(80 points)

김준우 (kjw8515@postech.ac.kr)

■ **제출 마감일: 2023.06.01 23:59**

■ **개발 환경: Python 3.x**

■ **제출물**

- .py 소스 코드 (assn4.py)
  - 프로그램의 소스 코드에 채점자의 이해를 돕기 위한 주석을 반드시 붙여주세요.
- 보고서 파일 (.docx, .hwp 또는 .pdf; assn4.docx, assn4.hwp 또는 assn4.pdf)
  - 보고서는 AssnReadMe.pdf를 참조하여 작성하시면 됩니다.
  - 명예 서약 (Honor code): 표지에 다음의 서약을 기입하여 제출해 주세요: "나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다." 보고서 표지에 명예 서약이 기입되어 있지 않은 과제는 제출되지 않은 것으로 처리됩니다.
  - 작성한 소스 코드와 보고서 파일은 PLMS를 통해 제출해 주세요.

■ **주의 사항**

- 구문 오류(Syntax Error)가 발생하거나 실행이 되지 않는 과제는 0점으로 채점됩니다.
- 제출 기한보다 하루 늦게 제출된 과제는 최종 20%, 이틀 늦게 제출된 과제는 최종 40% 감점됩니다. 제출 기한보다 사흘 이상 늦으면 제출 받지 않습니다 (0점 처리).

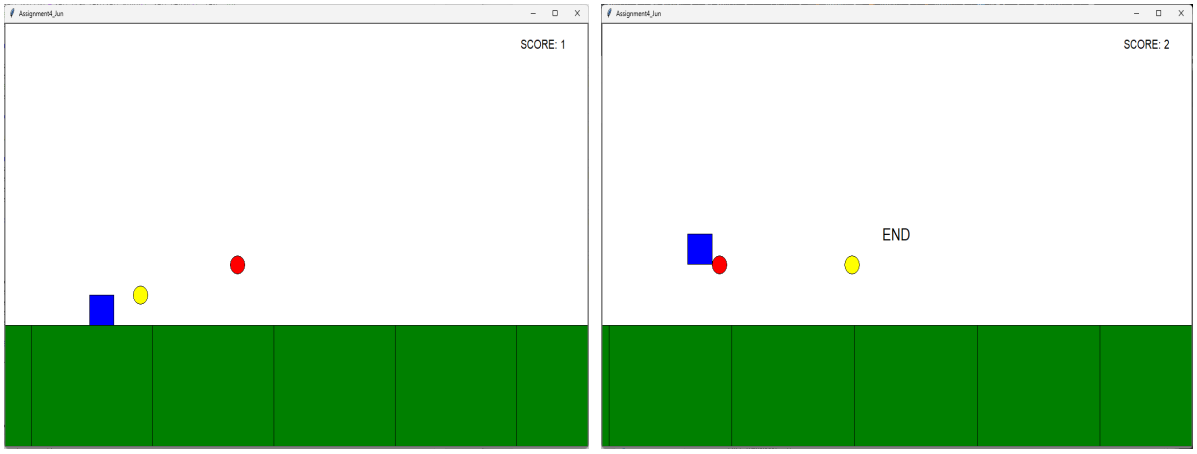
  늦은 제출시 PLMS에 기록된 최종 수정일시를 기준으로 감점합니다.
- 각 문제의 제한 조건과 요구 사항을 반드시 지켜 주시기 바랍니다.
- 이번 과제는 추가 기능 구현과 관련된 추가 점수가 따로 없습니다.
- 보고서 작성 시, 참고 링크도 포함해주세요. 부정 행위 적발 시 0점 처리됩니다.
- 부정행위에 관한 규정은 POSTECH 전자컴퓨터공학부 학부위원회의 "POSTECH 전자컴퓨터공학부 부정행위 정의"를 따릅니다 (PLMS의 본 과목 공지사항에 등록된 글 중, 제목이 [document about cheating]인 글에 첨부되어 있는 disciplinary.pdf를 참조하세요).

■ Problem: 쿠키런 (Cookie Run)

[문제]

이번 과제는 간단한 쿠키런 게임을 구현합니다. 지면위의 랜덤한 위치에서 생성되는 파이어 볼과 코인이 다가옵니다. 플레이어는 지면위에서 점프를 하여 파이어 볼을 피하고, 코인을 획득하여 점수를 높이는 게임입니다.

(**지면**: 초록색 사각형, **플레이어**: 파란색 사각형, **코인**: 노란색 원, **파이어 볼**: 빨간색 원)



[목적]

- Python의 tkinter 라이브러리를 이용하여 GUI 프로그래밍을 익힙니다.
- 클래스 정의 및 인스턴스 생성을 익힙니다.
- 클래스 상속 및 메서드 오버라이딩을 익힙니다.

[주의사항]

(1) 소스코드 저장 시 이름은 assn4.py 로 작성합니다.

(2) 보고서는 assn4.docx, assn4.hwp 또는 assn4.pdf 로 저장합니다.

(3) 문서에 클래스 상속 구조 및 각 클래스에서 정의하여야 할 변수와 메서드가 설명되어 있으니 확인 후 구현합니다.

(4) 명시된 작업 외의 작업은 점수 채점 요소가 아니니 고려하지 않아도 됩니다.

(5) 명시된 함수는 반드시 설명된 내용이 포함되어야 하며, 주어진 코드는 변경이 불가합니다.

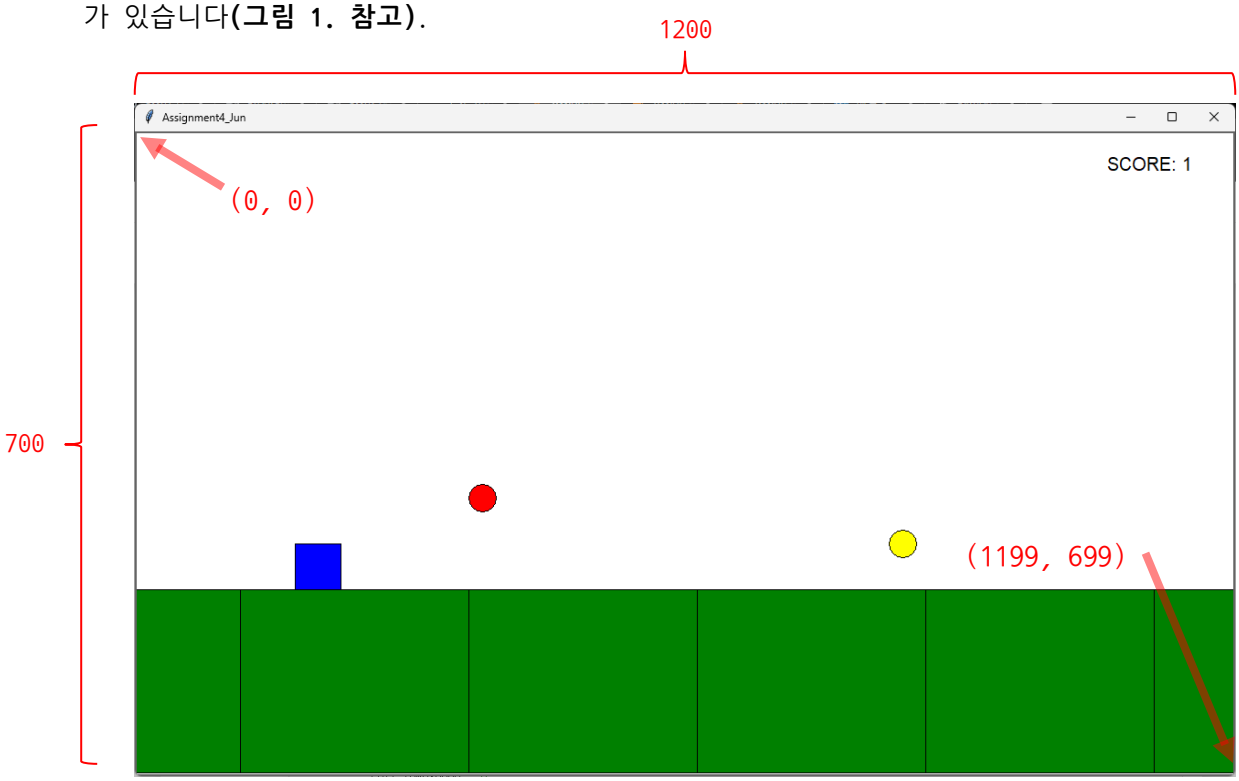
(6) 문제의 출력 형식은 채점을 위해 아래의 실행 예시와 최대한 비슷하게 작성해 주세요.

(7) 구현 전, 마지막 페이지 [참고] 에 설명 된 내용을 먼저 확인하시기 바랍니다.

[설명 및 요구사항]

1. 게임 구성

게임의 맵에는 흰 배경위에 플레이어(파란색 사각형), 지면(초록색 사각형), 파이어 볼(빨간색 원), 코인(노란색 원)이 있으며, 화면의 우상단에는 현재 획득한 코인의 점수를 나타내는 텍스트가 있습니다**(그림 1. 참고)**.



<그림 1. 게임 구성>

게임 구성 요소의 위치, 크기 및 색은 다음과 같습니다. 구현 시, 아래 명시된 값을 이용하세요.

(1) 게임 전체 화면: 가로 1200, 세로 700 (단위: 픽셀)

(2) 중심이 (x, y)이며, width/height 또는 radius 를 가진 사각형이나 원을 그려서 이용합니다.

(힌트) tkinter 모듈의 create_rectangle, create_oval 메서드를 사용할 것

- 지면(Terrain) 1개: y = 600, width = 250, height = 200, color = 'green'
- 플레이어(Player): x = 200, y = 475, width = 50, height= 50, color = 'blue'
- 파이어 볼(FireBall): x = 1200, y = 400 or 450, radius = 15, color = 'red'
- 코인(Coin): x = 1200, y = 400 or 450, radius = 10, color = 'yellow'

※ 파이어 볼과 코인은 생성시, y 위치는 400 또는 450 중에 하나가 랜덤하게 선택된다.

(3) 초기 게임 상태

- 지면(Terrain)은 빈 공백 없이 꽉 채워져야 한다.
- Coin/FireBall 은 각각 1 개 생성
- 점수 출력(Score Text)은 게임 화면의 오른쪽 상단에 "SCORE: 0"으로 초기화
- 게임 종료 시, 게임 중앙에 출력되는 화면(End Text)은 ""(빈 문자열)로 초기화

(힌트) Score Text와 End Text는 구현 시, create_text 메서드 사용할 것

2. 게임 동작 기능 (동영상 참고)

2.1. 플레이어

플레이어는 파란색의 사각형으로 만들어지며, 고정된 x 축 위치에서 y 방향으로만 움직입니다. ‘Space’ key 의 입력을 받으면, y 축 방향으로 일정 높이만큼 점프하고 낙하합니다. 점프의 높이는 두 개의 생성 위치 중 아래에서 생성되는 파이어 볼은 점프로 피할 수 있으며, 위에서 생성되는 파이어 볼에는 충돌하는 정도의 높이로 적당히 정하면 됩니다. 점프 후, 착지할 때는 지면 위에 있어야 합니다. 점프 중에 또 다른 점프는 불가능하며, 점프는 플레이어가 지면에 있을 때만 가능합니다.

2.2. 지면

지면은 초록색의 사각형으로, 맵에는 총 7 개의 지면으로 이루어집니다. 이 지면들은 연속적으로 이어져 있어야 합니다. 지면은 항상 공백이 없어야 합니다. 지면은 우측에서 좌측으로 이동하며, 지면의 우측면이 화면에서 사라지게 되면 사라진 앞쪽 지면을 삭제하고, 새로운 지면을 가장 우측의 지면에 이어 생성합니다.

2.3. 코인

코인은 노란색의 원으로 만들어지며, 맵에는 1 개 이상의 코인이 반드시 존재합니다. 코인은 지면과 동일한 속도로 우측에서 좌측으로 이동하며, 코인의 가장 우측 면이 화면에서 사라지거나 플레이어와 충돌하면 코인을 삭제하고, 새로운 코인이 화면의 우측에서 생성됩니다. 생성되는 코인은 두 가지의 y 위치 중 하나로 랜덤하게 생성됩니다. y 위치는 플레이어가 점프하지 않았을 때와 점프했을 때의 두 가지로 임의로 정합니다**(그림 2. 참고)**.

2.4. 파이어 볼

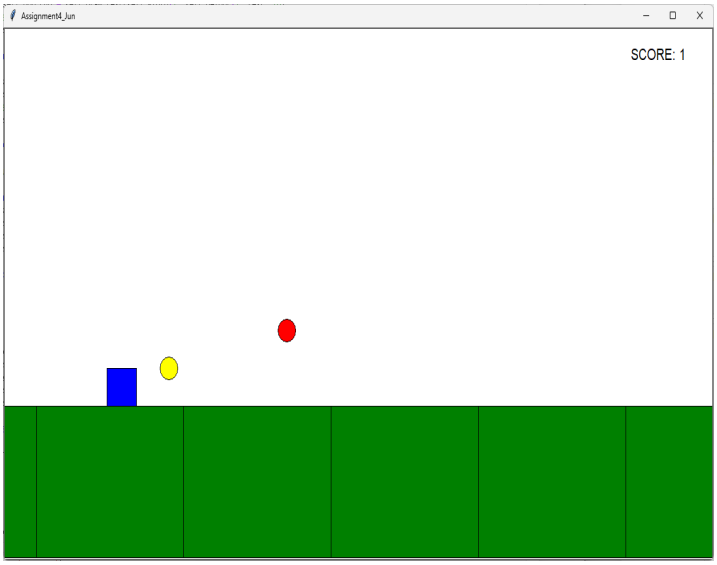
파이어 볼은 빨간색 원으로 만들며, 맵에는 1 개 이상의 파이어 볼이 반드시 존재합니다. 파이어 볼은 **지면과 코인보다 2 배 빠른 속도**로 우측에서 좌측으로 이동하며, 파이어 볼의 가장 우측 면이 화면에서 사라지면 파이어 볼을 삭제하고, 새로운 파이어 볼이 화면의 우측에서 생성됩니다. 생성되는 파이어 볼은 두 가지의 y 위치 중 하나로 랜덤하게 생성됩니다**(코인과 동일)**. 파이어 볼과 플레이어가 충돌하면, 게임은 종료되며 모든 동작이 정지됩니다.

2.5. 점수 텍스트

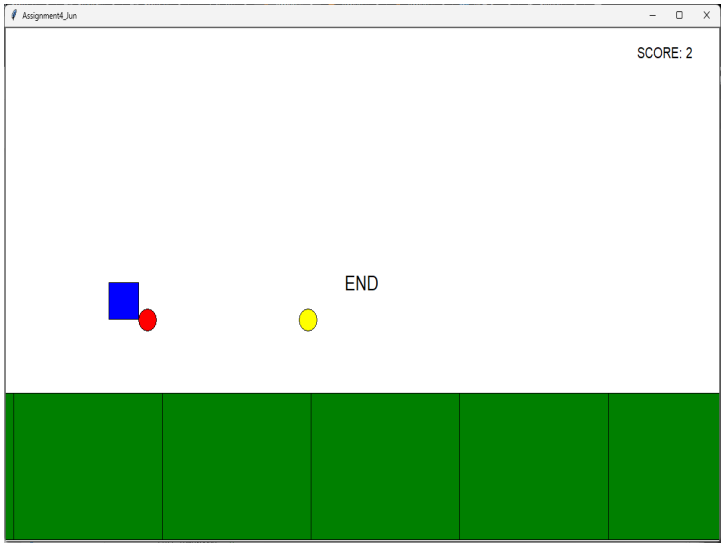
플레이어가 코인과 충돌하면 점수는 1 점 추가되며, **“SCORE: xx”(xx: 획득한 코인의 수)**의 텍스트를 우상단에 업데이트합니다. 텍스트는 매 루프마다 최신화 해주는 것이 아니라, 코인을 획득할 때만 점수를 업데이트합니다.

2.6. 종료 텍스트

파이어 볼과 플레이어가 출동하여 게임이 종료되면, **“ END ”**라는 텍스트를 화면 중앙에 배치합니다(**그림 2. (b) 참고**).



(a) 게임 플레이 화면



(b) 게임 종료 화면

<그림 2. 게임 동작: (a) 게임 플레이 화면, (b) 게임 종료 화면>

3. 구현 시 주의사항

(1) 반드시 assn4_problem.ipynb(또는 assn4_problem.py)를 ASSN4과제에서 다운로드 받아서, 과제 문서와 소스코드의 주석의 설명에 맞도록 소스코드를 채워서 제출하세요.

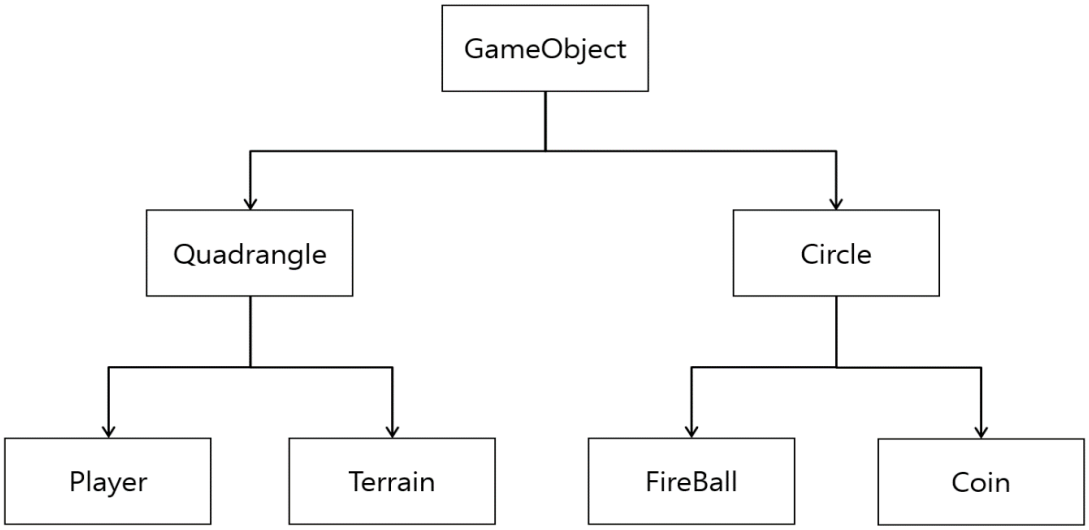
(2) 설명된 클래스 구조에 맞게 구현하세요.

(3) 명시하지 않은 객체의 위치는 임의로 지정할 수 있으나, 해당 동작을 테스트할 수 있게 구현하세요.

(4) 미리 정의된 변수와 메서드는 반드시 이용하세요. 단, 필요시 메서드나 변수는 수정 및 추가가 가능하며, 보고서 작성 시 내용을 포함해주세요. 설명이 빠지면 감점이 될 수 있습니다.

[클래스 구현 시 요구사항]

본 과제에서 사용되는 클래스의 상속 구조는 아래 **그림 3**과 같습니다.



<그림 3. 구현 클래스 구조>

(1) 메인 실행

```
if __name__ == '__main__':
    root = tk.Tk()
    root.title('Assignment 4_학번')      # 프로그램 이름 설정(각자 학번 입력)
    game = Game(root, 1200, 700)         # 가로 1200, 세로 700 창 생성
    game.mainloop()
```

(2) 클래스 설명

다음 변수 및 메서드들을 반드시 사용하여 프로그램을 구현해야 합니다. 아래 명시된 변수 이름, 메서드 이름 및 코드는 변경이 불가합니다. 메서드의 매개변수의 개수 및 리턴 값 등은 자유롭게 변경 가능 합니다. 이외에 필요한 변수 및 메서드는 추가로 정의해서 사용할 수 있으며, 추가한 내용에 대해서는 소스코드와 보고서에 설명을 포함하여 작성해주세요.

- **GameObject**
  - 인스턴스 변수

    ```
    self.canvas              # canvas 를 입력 받는 변수
    self.item                # canvas 에 그려지는 오브젝트
    ```

  - 메서드

    ```
    __init__(self, canvas, item):   # canvas, item 을 초기화하는 함수
    get_position(self):             # 특정 item 의 위치 값을 반환하는 함수
    delete(self):                   # 특정 item 을 삭제
    move(self, x, y):               # 특정 item 을 (x, y)만큼 이동
    ```

- **Quadrangle(GameObject)**
  - 인스턴스 변수

    ```
    self.x          # 사각형의 x 위치
    self.y          # 사각형의 y 위치
    self.width      # 사각형의 가로 길이
    self.height     # 사각형의 세로 길이
    self.color      # 사각형의 색상
    ```

  - 메서드

    `__init__(self, canvas, x, y, width, height, color):  # GameObject class 를 이용하여 변수 초기화`

- **Player(Quadrangle)**
  - 인스턴스 변수

    `self.state  # player 의 상태; 0: Running, 1: Jumping, 2: GameOver`

  - 메서드

    `__init__(self, canvas, x, y)  # Qudrangle class 를 이용하여 변수 초기화`

    `check_collision(self, tPos):  # 플레이어와 타겟 오브젝트가 충돌이 나면 1, 일어나지 않으면 0 을 리턴하는 함수(tPos: 플레이어와의 충돌 여부를 확인하기 위한 오브젝트의 위치 좌표 값), 구현 시 **마지막페이지의 [참고] 2 번 참고(AABB Collision)**하여 구현, 이외의 충돌 감지 알고리즘을 이용은 가능하나 보고서에 설명을 명확히 할 것 !`

- **Terrain(Quadrangle)**
  - 메서드

    `__init__(self, canvas, x, y)  # Qudrangle class 를 이용하여 변수 초기화`

- **Circle(GameObject)**
  - 인스턴스 변수

    ```
    self.x          # x 좌표
    self.y          # y 좌표
    self.radius     # 원의 반지름
    self.color      # 원의 색
    ```

  - 메서드

    `__init__(self, canvas, x, y, radius, color):  # GameObject class 를 이용하여 변수 초기화`

- **Coin(Circle)**
  - 메서드

    `__init__(self, canvas, x, y):  # Circle class 를 이용하여 변수 초기화`

- **FireBall(Circle)**
  - 메서드

    def __init__(self, canvas, x, y): # Circle class 를 이용하여 변수 초기화

- **Game(tk.Frame):** # tk.Frame class 를 상속
  - 인스턴스 변수

    self.width # 화면 창의 가로 길이

    self.height # 화면 창의 세로 길이

    self.canvas # 흰색 창을 가지는 canvas

    self.score # Coin 획득 점수

    self.mapSpeed # player 를 제외한 오브젝트들의 이동 속도(우측→좌측)

    self.jumpSpeed # player 의 점프 속도

    self.player # player

    self.terrains # map 의 모든 terrain 들을 포함하는 리스트

    self.fireballs # map 의 모든 fireball 들을 포함하는 리스트

    self.coins # map 의 모든 coin 들을 포함하는 리스트

    self.end_text # 게임 종료 시, 나타나는 텍스트

    self.score_text # Coin 획득 점수를 나타내는 텍스트
  - 메서드

    __init__(self, master, width, height): # Canvas 화면 창을 띄우는 함수

    setup_game(self): # 게임 기본 맵 및 기능 세팅

    gameInit(self): # 게임에서 동작하는 파라미터 및 맵 초기화

    game_loop(self): # 게임 종료 상태가 아니라면 게임 실행하는 함수

    gameSystem(self): # 전체적인 게임 동작

    manage_map(self): # terrains, fireballs, coins 의 전체적인 맵 관리

    manage_terrains(self): # terrain 의 우측면이 화면에 벗어나면 지우고 추가

    manage_fireballs(self): # fireball 의 우측면이 화면에 벗어나면 지우고 추가, 충돌 시 게임 종료

    manage_coins(self): # coin 의 우측면이 화면에 벗어나거나 획득 시 지우고 추가

    random_posY(self): # coin 이나 fireball 의 생성 시 랜덤한 2 개의 위치값을 반환

    check_gameState(self): # 게임 종료 상태면 “ END ” 텍스트 출력 및 점프 상태에 따른 점프 동작 함수를 실행

    update_jumping(self): # ‘ space ’ key 입력 시, 실행되는 함수, 점프 상태로 업데이트하는 함수

    jumping(self): # jump 동작 구현

    move_map(self): # terrains, coins 는 mapSpeed 의 속도로 이동하며, fireball 은 mapSpeed 의 2 배 속도로 이동함

[참고]

1. Canvas Class 참고 (이 외의 함수도 참고 가능)

https://tkinter-docs.readthedocs.io/en/latest/widgets/canvas.html

- coords
- delete
- move
- create_rectangle
- create_oval
- create_text
- itemconfig

2. AABB Collision 참고

https://developer.mozilla.org/en-US/docs/Games/Techniques/3D_collision_detection