2023학년도 1학기 CSED451

**Computer Graphics**

**Assignment #1**

**Report**

담당 교수: 이승용

팀 이름: 주단태

팀 구성원:

컴퓨터공학과 20190829 이태희

컴퓨터공학과 20190879 주의손

1. 팀 소개

-Team name: 주단태

-Team member  
 1) 이태희, 20190829, 컴퓨터공학과, thlee00  
 2) 주의손, 20190879, 컴퓨터공학과, uison1229

2. 프로그램 기능 개요

-모바일 게임 ‘Wind Runner’와 유사한 달리기 게임을 만든다.  
 -캐릭터는 자동으로 앞으로 달려나가며, 유저는 오직 점프 조작만 할 수 있다.  
 -바닥의 구멍을 점프로 회피하지 못하면 게임은 종료된다.  
 -전방에서 날아오는 파이어볼을 회피하지 못하면 게임은 종료된다.  
 -랜덤한 높이에 생성되는 코인을 획득하면 스코어가 증가한다.

3. 프로그래밍 환경

-Visual Studio 2019  
-C++  
-GLUT 3.7.6 (<http://user.xmission.com/~nate/glut.html>)

4. 프로그램 설계 및 구현

-프로그램 구조 (main.cpp)

main(): 게임이 실행되면 창을 생성해 그래픽을 출력하고, 마우스 클릭으로 점프 입력을 받으며 5ms에 한 번씩 TimerFunc를 무한 반복한다.

display(): Window에 지표면, 파이어볼, 코인, 점수판을 출력하며, 게임 종료 조건(파이어볼 충돌 또는 추락) 충족 시 프로그램을 종료한다.

reshape(): Window가 reshape 되면 화면 비율이 일그러지는 현상을 막기 위해 상수 값으로 viewport를 고정시켰다. 또한 캐릭터와 함께 시점을 이동시키기 위해 gluOrtho2D 함수의 x표에 user와 동일한 속도로 증가하는 position 값을 더해주었다.

mouse(): 마우스 입력은 2단 점프를 방지하기 위해 “jump flag가 꺼진 경우에만” 동작하도록 설정하였다.

timer(): 무한히 반복되는 TimerFunc에서는 게임의 상태를 실시간으로 업데이트한다.  
1) 캐릭터의 x좌표를 계속 증가시키며, 점프/추락 여부에 따라 y좌표를 조정한다.  
2) 카메라 시점과 스코어 출력이 캐릭터에게 뒤쳐지지 않도록 함께 이동시켜준다.  
3) 새로운 지표면 블록, 파이어볼, 코인을 생성한다.  
4) 캐릭터가 코인과 접촉하면, 점수를 증가시키고 해당 코인을 삭제한다.

-자료 구조 (obj.h, obj.cpp)

class RectObj: 최상위 부모 클래스로 RectObj를 두어, 직사각형 형태의 모든 object가 공유하는 특성(x,y좌표 및 너비와 높이, 그리기 함수)을 정의한다.

class User: 유저 캐릭터에 쓰이는 User 클래스는 RectObj를 상속받는다. 그 크기와 색깔은 생성자에서 정의되며 ‘캐릭터 밑에 발판이 존재하는지 판별하는 변수’로 InTerrain을 둔다.

class TerrainBlock: 지표면 발판에 쓰이는 TerrainBlock 클래스도 RectObj를 상속받는다. 디자인적 요소로 Soil 블록, Grass 블록을 각각 생성한다. ‘함정으로 구멍을 생성했는지 여부를 나타는 변수’로 exist를 둔다.

class FireBall: 파이어볼 장애물로 쓰이는 Fireball 클래스도 RectObj를 상속받는다. 게임의 재미를 위해 파이어볼이 고정된 높이로만 날아오지 않고, 점프를 통해 회피 여부가 갈리도록 2가지로 조정하였으며, 이를 구별하기 위한 변수로 elev를 둔다.

class Coin: 마지막으로 스코어 획득을 위해 쓰이는 Coin 클래스도 RectObj를 상속받는다.

-알고리즘 (map.h, map.cpp)

Map::init(): 게임판을 초기에 생성할 때, 최초의 Terrain 블록 5개를 한 번에 생성한다.

Map::newTerrainBlock(): 이후로는 캐릭터가 블록 한 칸만큼 이동할 때마다(30ms 마다) 실시간으로 블록 혹은 구덩이를 화면 밖에 한 칸 생성해 연속적으로 게임이 진행되게 한다. 구덩이가 파질 확률은 rand 함수와 나머지 연산으로 1/4로 설정했다.  
Terrain 블록을 무한히 생성하며 달려나가면 메모리적으로 좋지 못하다고 생각해, deque 자료구조를 통해 캐릭터가 이미 지나친 object는 삭제되도록 구현하였다. 이는 파이어볼과 코인 모두에 동일하게 적용된다. 지나쳤다는 판정은 object 별로 상세한 연산은 다르지만, 공통적으로 캐릭터와 해당 object의 x좌표를 비교해 화면 밖으로 나간 object를 체크하는 방식으로 진행했다.

Map::newFireBall(): 파이어볼도 rand 함수를 통해 일정 확률로 생성이 되어 날아오며, 상기한 것과 같이 elev 변수를 통해 높이가 랜덤으로 결정된다.  
또한 x좌표를 5ms마다 0.003씩 감소시켜 map의 이동 속도보다 빠르게 캐릭터에게 다가오도록 하였다.

Map::checkFireBall(): 파이어볼과의 충돌 여부는 캐릭터와 x, y 좌표 비교를 통해 실시간으로 연산된다.

Map::newCoin(): 코인 생성 시 게임의 재미를 위해, 코인을 고정된 높이에만 생성하지 않고 지표면/점프 최고점/점프 중간점 세가지 높이에서 수집할 수 있도록 컨트롤 요소를 가미했다.

Map::EatCoin(): 파이어볼과 마찬가지로 x, y 좌표를 비교해 코인과 캐릭터가 접촉하는지 연산하는데, 추가적으로 코인은 ‘획득’의 개념이므로 deque에서 erase를 해주어 화면에서 사라지게한다.

Map::calUserPosition(): 유저가 좌클릭을 해 jump flag(up)이 들어오면, 높이가 0.35에 도달할 때까지 y좌표를 0.01씩 증가시킨다. 최고점에 도달하면 up flag를 끄고 down flag를 켠다.  
down flag가 켜지면, 높이가 원점으로 돌아올 때까지 y좌표를 0.01씩 감소시킨다. 이 때, 블록위에 정상적으로 착지하면 down flag를 꺼 점프를 중지하지만, 도착점이 허공일 경우 계속 y좌표가 감소해 추락하도록 한다.  
그 외 점프를 하지 않아 추락하는 경우, 점프가 짧아 도착 블록 모서리에 막혀 추락하는 경우 등의 예외 처리를 하였다. 점프 동작과 별개로 캐릭터는 항상 0.01씩 x좌표를 증가시킨다.

Map::findLRTerrainBlocks(): 캐릭터가 허공에 있는지 여부는 캐릭터 왼쪽, 오른쪽 블록 모두가 구덩이일 경우를 계산한다.

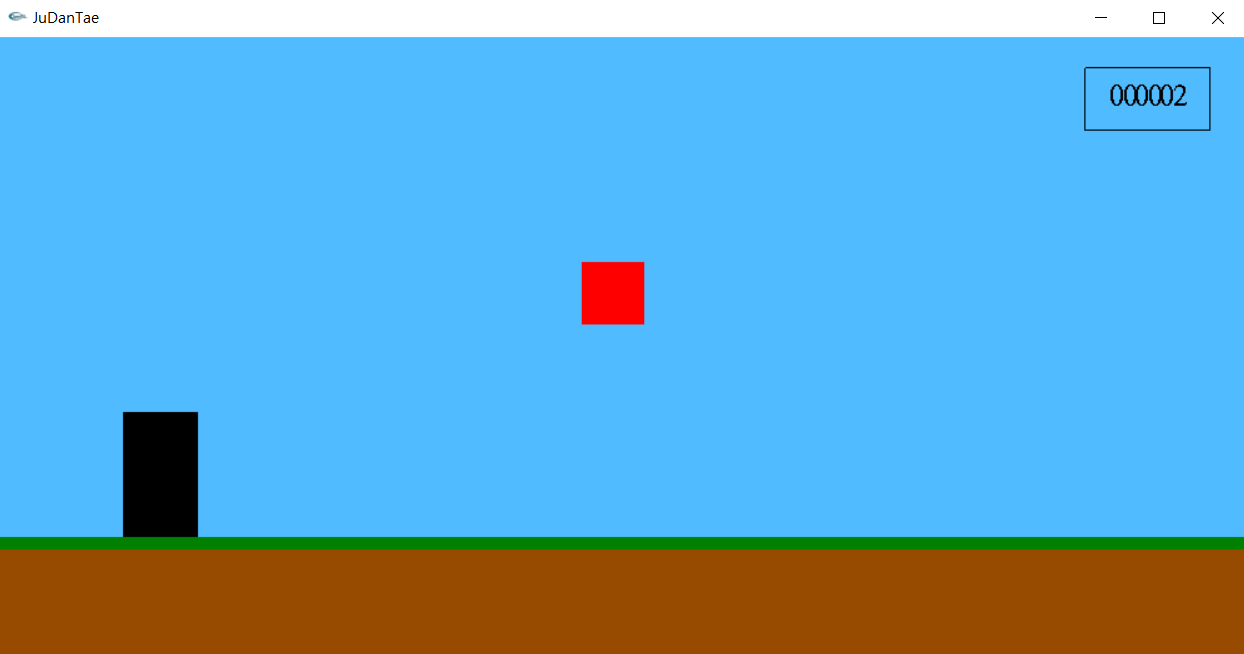
Map::checkEndCondition(): 게임 종료 조건은 y좌표 하락(구덩이에 빠진 상황), 파이어볼 충돌로 총 2가지이다.

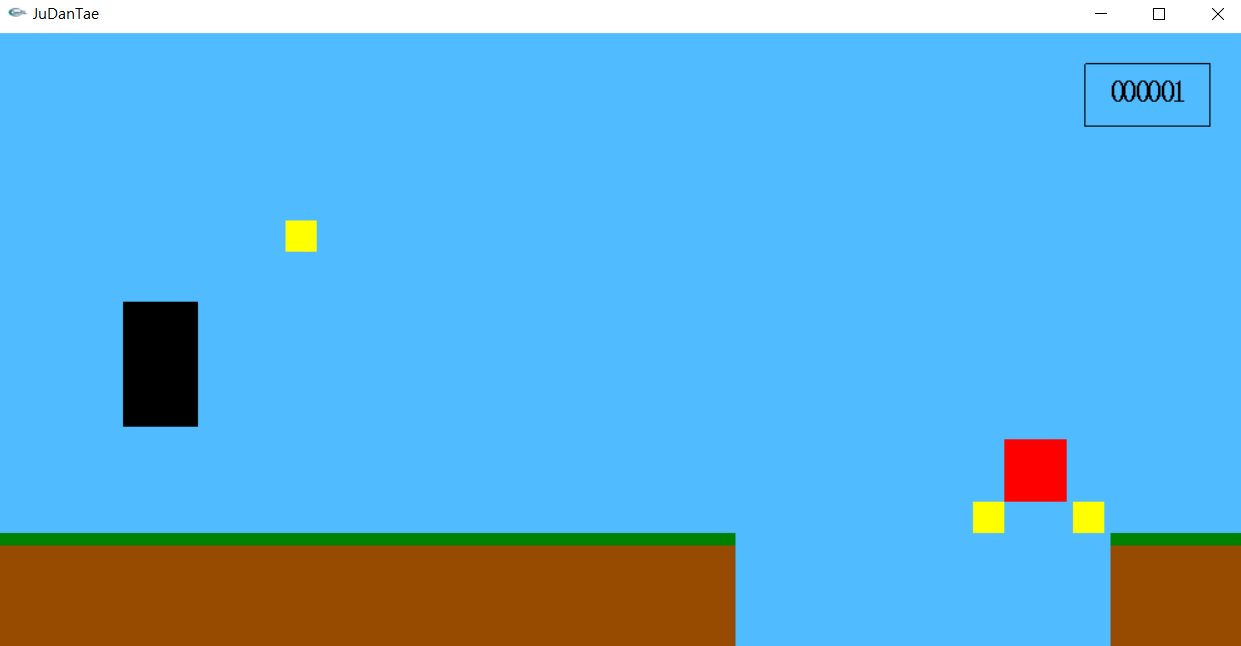
5. 프로그램 실행 방법

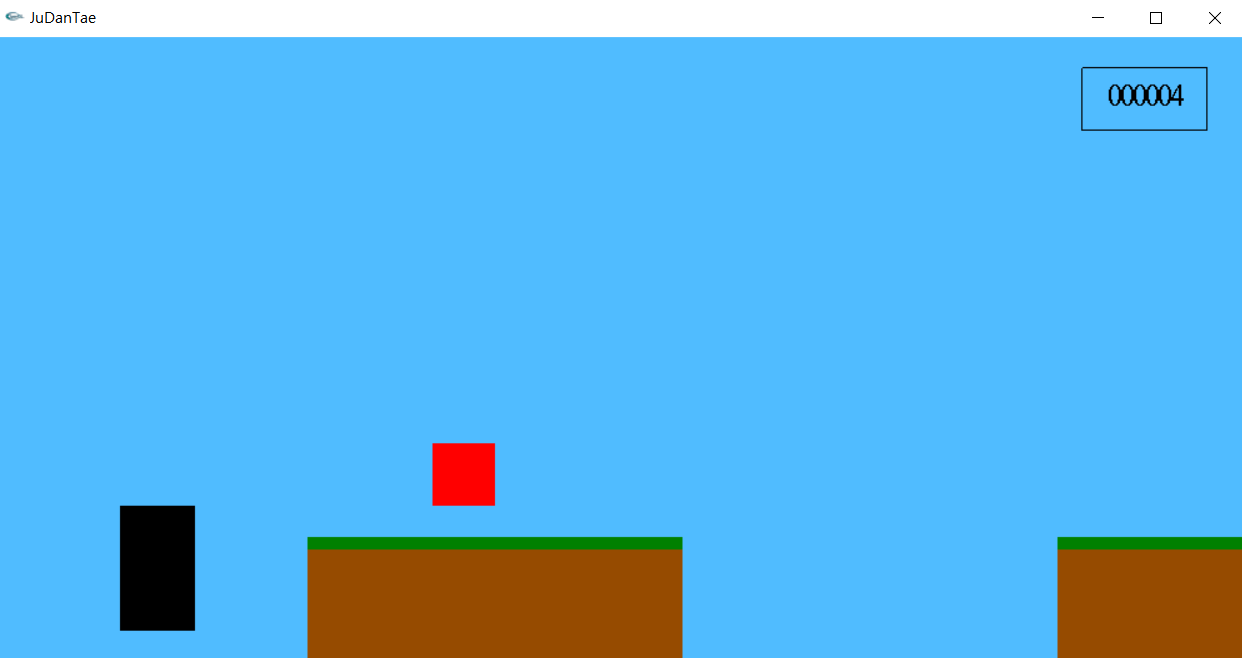
-Visual Studio 2019를 통해 프로젝트(assn1.sln)를 열어 Ctrl+F5로 main.cpp 코드를 컴파일하면 게임이 시작된다.  
 -유저는 ‘마우스 좌클릭’을 통해 캐릭터를 점프시킬 수 있다.  
 -코인을 통해 획득한 스코어는 우측 상단에 표시된다.  
 -파이어볼과 충돌하거나, 구멍에 빠진다면 프로그램이 종료된다.

6. 예제

  
-게임을 시작한 모습 (스크린샷을 찍는 와중 코인을 하나 획득해 우측 상단 스코어가 1증가해있는 모습이다.)

  
-파이어볼이 날아오는 모습

  
-다양한 높이에서 생성되는 코인을 먹기위해 점프하는 모습

  
-점프를 제때 하지 못해 바닥으로 추락하는 모습

7. 토론

-2단 점프를 방지하기 위해 캐릭터가 지표면 y좌표에 있을 경우만 점프가 가능하도록 구현했더니, ‘점프를 너무 빨리 해 구덩이를 다 넘지 못하고 중간에 추락하는 상황’에서 연속적으로 클릭을 하면 지표면 y좌표와 동일해지는 지점에서 허공을 밟고 점프를 하는 현상을 발견하였다. 이를 해결하기 위해 점프 최고점에서 중력의 영향으로 내려올 때 밑에 블록이 없다면 추락하도록 예외처리를 하였다.

-glutBitmapString 함수를 지원하지 않아 char 배열을 만들어 glutBitmapCharacter를 반복문으로 호출해 점수를 표시하였다.

-게임 난이도가 과하게 어렵다고 판단해, 구덩이와 파이어볼 생성 확률을 대폭 감소시켰다. 또한 캐릭터 크기를 축소하고, 점프력을 증가시켰다.

8. 결론

-OpenGL을 이용해 Window를 생성하고, 원하는 그래픽을 출력하는 방법을 배웠다.

-OpenGL을 이용해 키보드, 마우스 등의 input을 받는 방법을 배웠다.

-원하는 시간마다 자동으로 반복되는 TimerFunc 사용법을 배웠다.

-수업 시간 개념으로 배웠던 “화면의 일부만 변경하지 않는 것이 오히려 번거로우므로, 화면 전체를 매번 다시 그린다”는 교수님의 가르침을 직접 체험하였다.

9. 개선 방향

-가끔 파이어볼과 바닥의 구멍이 겹쳐 회피할 수 없는 ‘무조건 죽는 패턴’이 발생하는데, 예외처리를 많이 까다롭게 한다면 이러한 상황을 제거할 수 있을 것이다.

-코인이 지금은 랜덤하게 생성되지만, 모바일 게임 ‘쿠키런’과 같이 캐릭터의 동선을 유도하는 방향으로 생성할 수도 있을 것이다.

-게임 시작 시 초기 화면과 게임 종료 화면을 추가로 구현할 수 있을 것이다. 현재 게임 종료 시에는 바로 프로그램이 종료되는데, 게임 종료 화면에서 점수를 출력한 뒤 재도전을 할 수 있도록 개발할 수도 있을 것이다.

-그 외에도 2단 점프, 슬라이드, 아이템 추가 등 게임성을 개선할 수 있는 방법은 무궁무진하다.