과목명 : 의공학 프로그래밍

과제명 : Assignment3

제출일자 : 2019.06.10

• 애니메이션 설명: 새롭게 만든 Class와 만들어진 객체가 무엇인지, 그리고 어떤 이벤트를 처리하는지 설명
• 입력과 수행의 예: 문제의 입출력에 대한 샘플 예와 수행결과를 캡처하여 출력해야 한다. (각 이벤트에 대해)
• 결과분석 및 토의 : 본인이 작성한 프로그램에 대한 정확한 결과 분석(Complexity와 Creativity)과 본인이 느꼈던 사항들을 기재

• 애니메이션 설명: 새롭게 만든 Class와 만들어진 객체가 무엇인지, 그리고 어떤 이벤트를 처리하는지 설명

자판기 애니메이션을 만들기 위해 Drink와 Coin 두개의 Class를 새롭게 만들었다.
Drink를 하나의 클래스로 만들었다. Drink는 body, label, bottom, top, cap, extra를 속성으로 갖게 된다. 그리고 Drink 객체 bottle과 can 음료를 객체로 갖는다.
이 때, make_drink() 라는 함수를 정의하여 bottle과 can을 생성하도록 하였다.
Coin Class는 coin1과 coin2를 속성으로 갖고 coin이라는 객체를 생성하게 된다.

"동전을 넣으시오" 라는 메시지가 뜨고 마우스로 동전을 클릭하면 동전이 자판기에 투입되면서 캔음료가 하나 나오게 된다.
e.getDescription()을 이용하여 "마우스 클릭"을 입력 받으면 for 반복문으로 코인이 동전투입구로 이동하게 된다. 이후 사이다가 앞의 내용과 마찬가지로 for문으로 음료배출구로 빠져나오게 된다.

이 후 동전이 없으면 발로 자판기를 차라는 문구가 나오면 키보드 q를 입력하면 발이 두 번 자판기를 걷어차게 된다.. Input 함수를 이용하여 q를 입력하면 자판기 옆에 있던 발이 for문과 rotate,move 함수를 이용하여 두 번 이미지가 왕복하면서 자판기를 두 차례 차게 된다.
이후 원래 있던 9개 음료 중 남아 있는 8개의 음료가 우르르 음료배출구로 떨어지는데 for문을 이용해 아래로 떨어지는 move와 좌우로 움직이는 move가 계속해서 반복하면서 덜컹거리면서 빠르게 떨어지는 애니메이션을 볼 수 있다.

이 때 for문에서 속도조절과 move방향 조절을 하여 동전을 투입하고 음료를 구입하게 되면 천천히 부드럽게 음료가 떨어지지만 발로 걷어차서 음료를 꺼내게 되면 음료가 덜컹거리며 빠른 속도로 떨어지게 된다.

```
from BMP_Packages.BMEgraphics import *

canvas = Canvas(600, 600)
canvas.setBackgroundColor("light blue")      #캔버스 배경

machine = Layer()
tire1 = Rectangle(150, 250, Point(80,60))   #자판기 제작
tire1.setFillColor('skyblue')
machine.add(tire1)
tire2 = Rectangle(200, 60, Point(100,260))
tire2.setFillColor('black')
machine.add(tire2)
body = Rectangle(210, 400, Point(100, 100))
body.setFillColor('red')
body.setDepth(60)
machine.add(body)
line1 = Rectangle(140, 10, Point(80, 60))
line1.setFillColor('red')
machine.add(line1)
line2 = Rectangle(140, 10, Point(80, 0))
line2.setFillColor('red')
machine.add(line2)
line3 = Rectangle(140, 10, Point(80, 165))
line3.setFillColor('red')
```

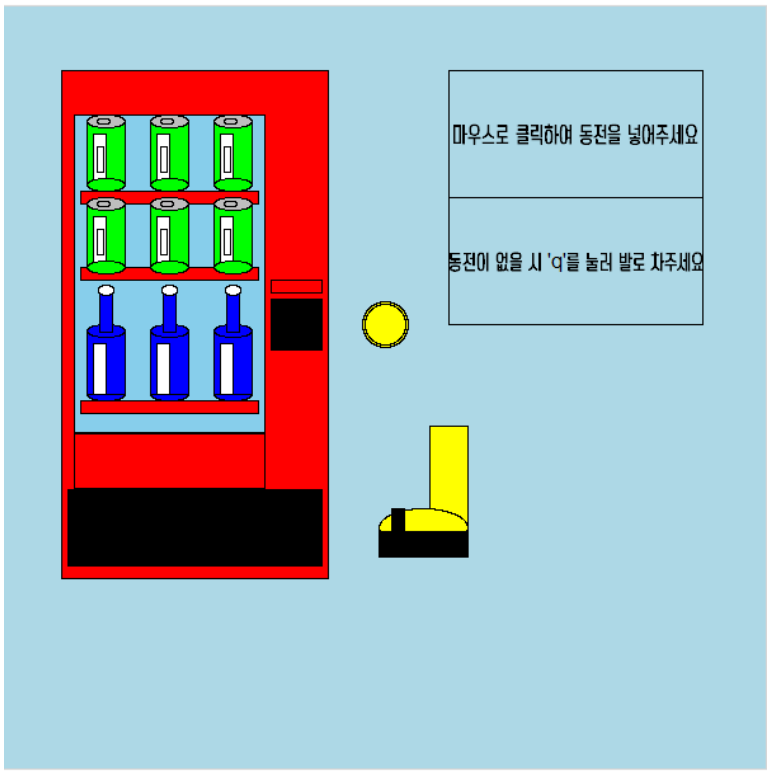
```
machine.add(line3)
line4 = Rectangle(40, 10, Point(180, 70))
line4.setFillColor('red')
machine.add(line4)
coinin = Rectangle(40, 40, Point(180,100))
coinin.setFillColor('black')
machine.add(coinin)

machine2 = Layer()
body2 = Rectangle(150, 43, Point(130,357))     # 자판기 음료배출구 뚜껑 제작
body2.setFillColor('red')
machine2.add(body2)


class Coin(object):
    def __init__(self, coin1=None, coin2 = None):   # coin Class
        layer = Layer()
        self.layer = layer
        self.coin1 = coin1
        self.coin2 = coin2

def make_coin():
    layer = Layer()
    coin1 = Circle(18)
    coin1.setFillColor('yellow')              # coin 만드는 함수 정의
    layer.add(coin1)
    coin2 = Circle(16)
    coin2.setFillColor('yellow')
    layer.add(coin2)

    co = Coin()
    co.layer = layer
    co.coin1 = coin1
    co.coin2 = coin2
    return co

def move(self, dx, dy):
    self.layer.move(dx, dy)                     # move 함수 정의

class Drink(object):                             # Drink Class

    def __init__(self, body=None, bottom = None, top = None, cap = None, extra = None):
        layer = Layer()
        self.layer = layer
        self.body = body
        self.bottom = bottom
        self.top = top
        self.cap = cap
        self.extra = extra



def make_drink(bottle = False):
    layer = Layer()
    if bottle:
        body = Rectangle(30,50)
        body.setFillColor('blue')
        label = Rectangle(10,40)
```

```
        label.setFillColor('white')
        label.move(-5,5)                        # Drink(bottle, can) 제작하는 함수 정의
        top = Ellipse(30,10)
        top.setFillColor('blue')
        top.move(0,-25)
        bottom = Ellipse(30,10)
        bottom.setFillColor('blue')
        bottom.setDepth(60)
        bottom.move(0,25)
        extra = Rectangle(10,35)
        extra.setFillColor('blue')
        extra.move(0,-42)
        cap = Ellipse(12,8)
        cap.setFillColor('white')
        cap.move(0,-57)
    else:
        body = Rectangle(30,50)
        body.setFillColor('green')
        label = Rectangle(10,40)
        label.setFillColor('white')
        label.move(-5,5)
        extra = Rectangle(5,20)
        extra.setFillColor('white')
        extra.move(-5,5)
        top = Ellipse(30,10)
        top.setFillColor('gray')
        top.move(0,-25)
        bottom = Ellipse(30,10)
        bottom.setFillColor('green')
        bottom.move(0,25)
        cap = Ellipse(10,5)
        cap.setFillColor('gray')
        cap.move(-2,-25)
    layer.add(body)
    layer.add(label)
    layer.add(top)
    layer.add(bottom)
    layer.add(extra)
    layer.add(cap)

    dr = Drink()
    dr.layer = layer
    dr.body = body
    dr.label = label
    dr.top = top
    dr.bottom = bottom
    dr.extra = extra
    dr.cap = cap
    return dr                           #return dr


foot = Layer()
leg = Rectangle(30,80, Point(350, 370))
leg.setFillColor('yellow')
foot.add(leg)
foot1 = Ellipse(70,30, Point(330,410))
foot1.setFillColor('yellow')
foot.add(foot1)
```

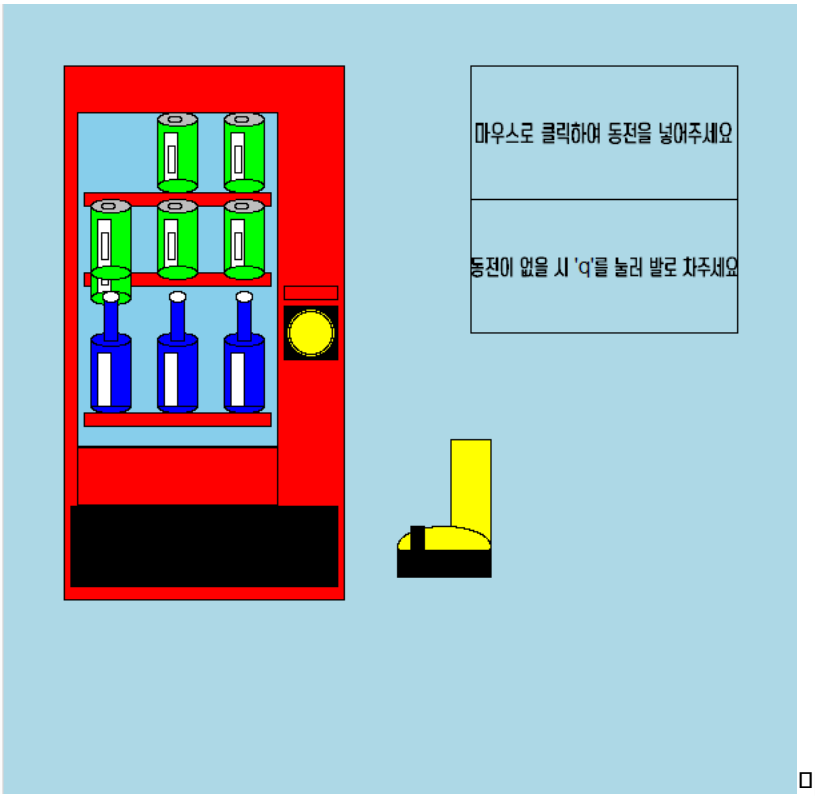
```
foot2 = Rectangle(70,20,Point(330, 423))
foot2.setFillColor('black')
foot.add(foot2)
foot3 = Rectangle(10,20, Point(310, 405))          # 다리와 발 제작
foot3.setFillColor('black')
foot.add(foot3)



bottle1 = make_drink(True)
bottle2 = make_drink(True)                  # 앞서 정의한 함수 이용하여 bottle 3 개 만들기
bottle3 = make_drink(True)
can0 = make_drink()
can1 = make_drink()
can2 = make_drink()                         # 앞서 정의한 함수 이용하여 can 6 개 만들기
can3 = make_drink()
can4 = make_drink()
can5 = make_drink()
bottle2.layer.move(50,0)
bottle3.layer.move(100,0)
can0.layer.move(80,115)
can1.layer.move(0,-100)                 #can 과 bottle 위치 입력
can2.layer.move(50,-100)
can3.layer.move(100,-100)
can4.layer.move(50,-165)
can5.layer.move(100,-165)

vending1= Layer()
vending2 = Layer()
vending3= Layer()
vending2.add(can1.layer)
vending2.add(can2.layer)
vending2.add(can3.layer)
vending1.add(can4.layer)           #vending1, vending2, vending3  layer 만들고 음료 layer 에 추가
vending1.add(can5.layer)
vending3.add(bottle1.layer)
vending3.add(bottle2.layer)
vending3.add(bottle3.layer)
vending1.move(80,280)
vending2.move(80,280)
vending3.move(80,280)
#coin = Layer()
coin11 = make_coin()
coin11.layer.move(300,250)       ## coin 제작
#coin.add(coin11.layer)
machine.move(50, 150)


canvas.add(machine)
canvas.add(can0.layer)
canvas.add(vending1)
canvas.add(vending2)             #캔버스에 자판기. 동전, vending1, vending2, vending3, 발, 음료 추가
canvas.add(vending3)
canvas.add(coin11.layer)
canvas.add(foot)
canvas.add(machine2)

button1 = Layer()
```

```
button11 = Rectangle(200, 100)
button12 = Text("마우스로 클릭하여 동전을 넣어주세요", 15)  #마우스 입력버튼
button1.add(button11)
button1.add(button12)
button1.move(450, 100)

canvas.add(button1)

button2 = Layer()
button21 = Rectangle(200, 100)
button22 = Text("동전이 없을 시 'q'를 눌러 발로 차주세요", 14)
button2.add(button21)
button2.add(button22)              #키보드 입력 버튼
button2.move(450, 200)

canvas.add(button2)

def click():
    e = canvas.wait()
    d = e.getDescription()             #마우스 입력 함수 정의
    if d == 'mouse click':
        for i in range(35):
            coin11.layer.move(-2, 0)      #동전 투입
        for i in range(100):
            can0.layer.move(0, 3)          #음료 구매


def keyboard_input():
    P = input("q를 눌러 발로 자판기를 차주세요")  #키보드 입력 함수 정의
    if P == 'q':
        for i in range(20):
            foot.move(-2, -1)
            foot.rotate(0.5)
        for i in range(20):
            foot.move(2, 1)       #자판기를 발로 참(2번)
        foot.rotate(-0.5)
        for i in range(20):
            foot.move(-2, -1)
            foot.rotate(0.5)
        for i in range(20):
            foot.move(2, 1)
            foot.rotate(-0.5)

        for i in range(6):
            vending1.move(0, 48)
            vending1.move(0.8, 0)
            vending2.move(0, 40)          #남은 음료들이 배출구로 떨어짐
            vending2.move(1, 0)
            vending3.move(0, 28)
            vending3.move(1.6, 0)

click()
keyboard_input()
```

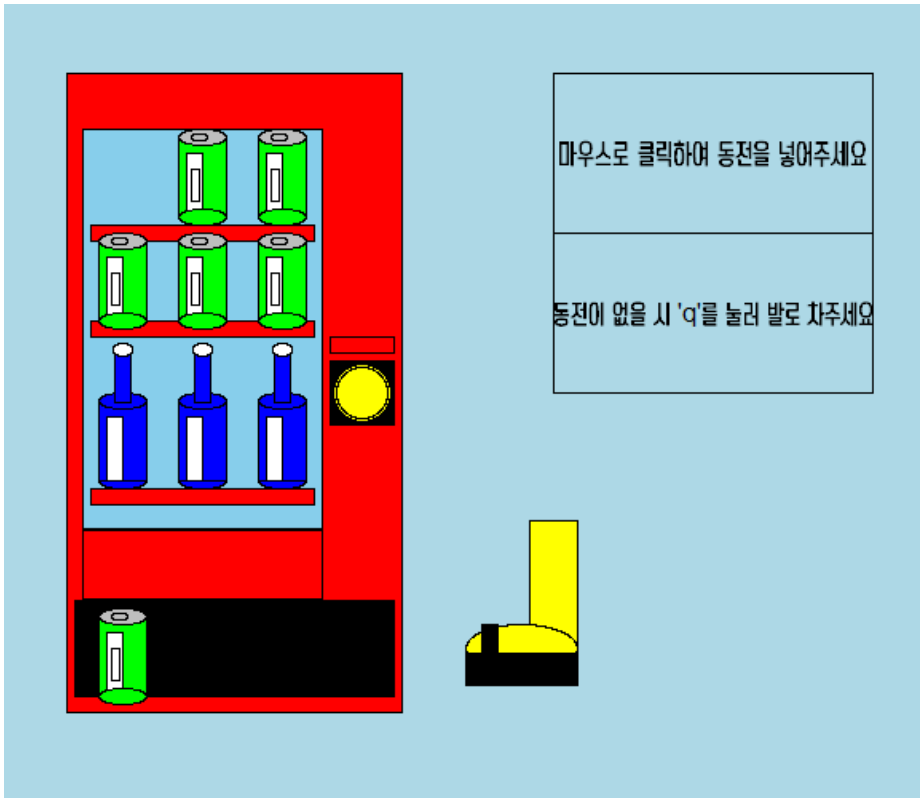
• 입력과 수행의 예: 문제의 입출력에 대한 샘플 예와 수행결과를 캡처하여 출력해야 한다. (각 이벤트에 대해)



마우스입력 전



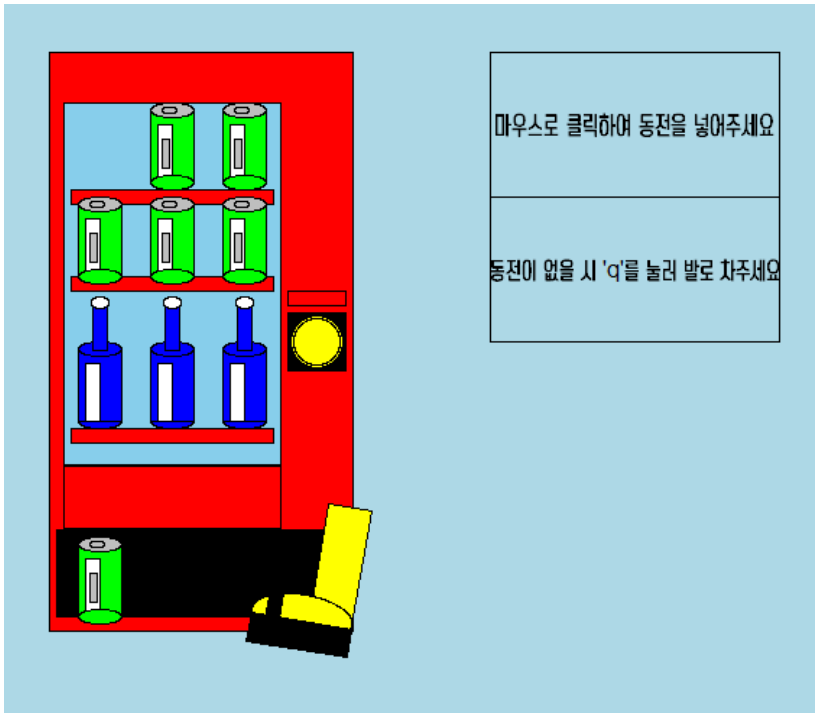
마우스 입력 후 동전 투입



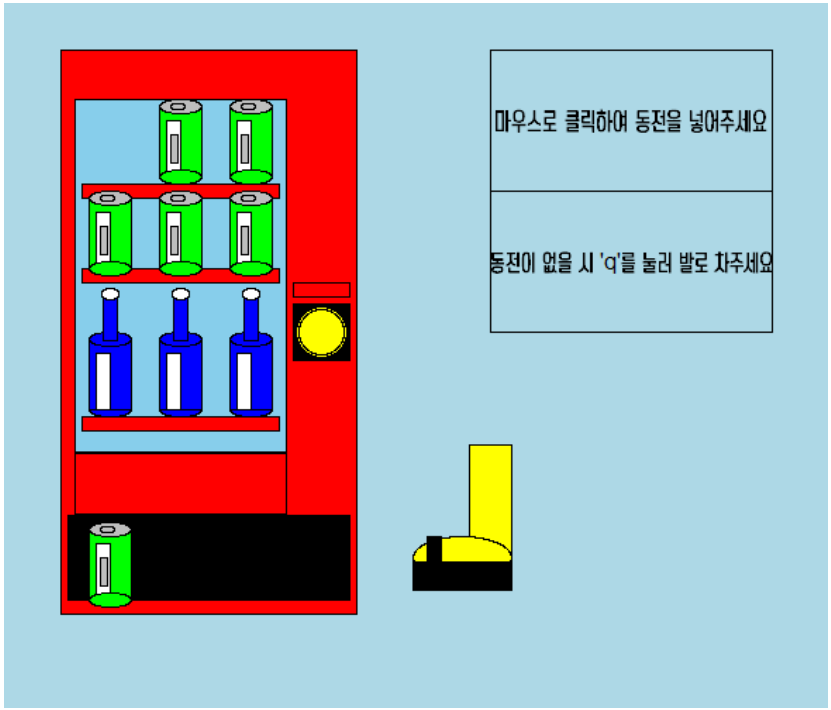
동전 투입, 캔 음료 빠져나옴

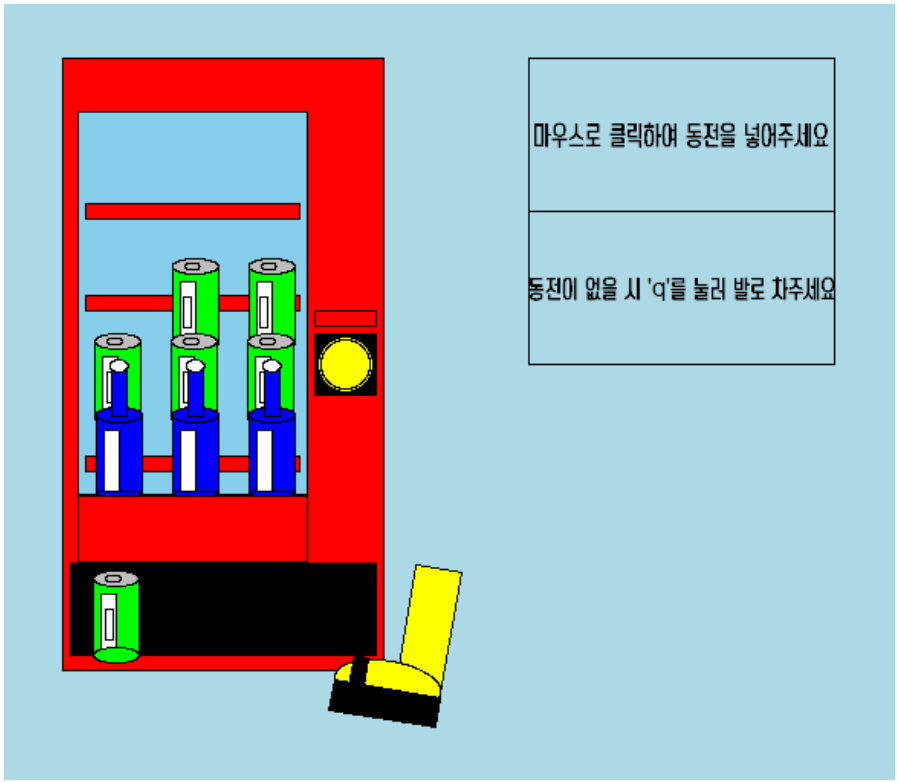
```
C:₩Users₩user₩BMP_Labs₩venv₩Scripts₩python.exe C:/Users/user/BMP_Labs/3333333.py
q를 눌러 발로 자판기를 차주세요q
Close canvas windows to end program.
```

키보드 q 입력

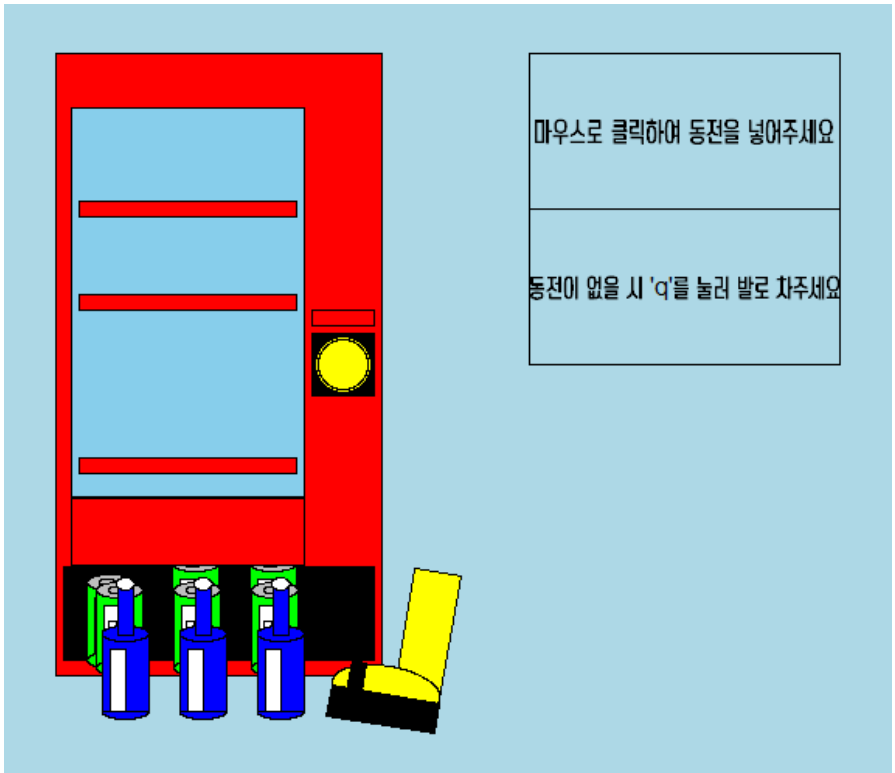

마우스로 클릭하여 동전을 넣어주세요
동전이 없을 시 'q'를 눌러 발로 차주세요

발로 두차례 걷어참


마우스로 클릭하여 동전을 넣어주세요
동전이 없을 시 'q'를 눌러 발로 차주세요



발로 두차례 걷어참 ➔ 음료 배출되기 시작



남아있던 음료가 전부 떨어짐

• 결과분석 및 토의: 본인이 작성한 프로그램에 대한 정확한 결과 분석(Complexity와 Creativity)과 본인이 느꼈던 사항들을 기재

Complexity 측면 :

vending machine과 bottle, can, coin, foot, button 등 많은 수의 이미지를 사용하여 애니메이션을 제작하였다. 또한 이미지들이 비대칭인 경우가 많아 일일이 좌표를 지정하여 위치를 정하면서 정교한 이미지 제작을 할 수 있었다. 이미지 요소가 많은 이미지를 제작하였다.

음료수자판기에도 유리창, 가판대, 동전투입구, 음료배출구, 기계 등 여러 요소가 많다. 음료도 캔과 밑바닥, 윗면, 뚜껑, 상품로고, 병목 등 여러 요소가 포함되어 있다. 이처럼 여러 복잡하고 많은 요소가 담긴 이미지를 제작하였다.

또한 이미지의 이동을 많이 하고 섬세하게 제작하여 정교한 애니메이션을 구현하였다, 동전 투입, 캔음료 구입, 발로 자판기 걷어차기, 남은 음료들 다 떨어짐, 등 다양한 애니메이션을 구현하였다.

또한 bottle과 can의 이미지를 제작하는 과정에서 Drink Class를 새롭게 만들었다. Drink Class에서 make_drink()라는 함수를 정의하여 많은 양의 bottle과 can을 만들 수 있도록 하였다. 이 때, body, label, bottom, top, cap, extra 등을 속성으로 가져 요소가 많은 다소 복잡한 class를 제작하였다.

그리고 발로 자판기를 걷어차는 과정에서 발과 다리가 흔들리는 것은 rotate를 이용하고 move 함수 정의하여 이용하였고 그 밖의 이미지들은 따로 직접 정의한 move 함수를 이용하거나 moveTo를 이용하여 애니메이션을 구현하였다.

자판기 안의 can과 bottle을 포함한 음료의 개수가 총 9개로 각각의 위치와 애니메이션을 지정하는 것이 다소 복잡하였다. 각 자판기의 층별로 layer를 지정하여 각각 좌표와 애니메이션을 입력하여 발로 걷어찼을 때 음료들이 떨어지는 애니메이션을 구현하였다..

Creativity 측면:

Bottle 과 can의 형태가 비슷한 것을 이용하여 Drink Class를 지정하였다. Body, bottom, top, cap, label을 공통인 속성으로 갖는 것을 이용하였고 다른 점인 bottle의 목부분과 can의 추가 label을 속성 'extra'로 지정하여 Drink를 총 6가지의 properties를 가지도록 제작할 수 있었다. 이를 이용하여 그렇지 않았을 때보다 코드를 좀 더 간단하게 표현할 수 있었고 그에 따른 버그 발생도 줄일 수 있었다.

추가로 발로 걷어차서 음료를 떨어지게 하는 과정에서 음료 세 층을 하나의 layer로 하면 떨어지는 위치가 달라지므로 각 층별로 layer를 지정하였다. Can0.layer, Vending1, Vending2, Vending3 이렇게 총 4가지 layer를 제작하여 각각의 떨어지는 위치와 속도를 조절할 수 있었다.

뿐만 아니라 동전투입시와 발로 걷어찰 때 음료가 떨어지는 애니메이션에 차이를 주었다. 투입시에는 천천히 매끄럽게 떨어지지만 발로 걷어찰 때는 걷어차는 행동에 의미를 주기 위해 좌우로 움직이는 애니메이션을 for문을 사용해 덜컹거리면서 빠르게 떨어지도록 하여 애니메이션 효과를

극대화하였다.
음료가 음료배출구로 떨어지는 과정에서 음료수자판기의 안쪽으로 떨어지지만 음료배출구 밖으로 음료가 빠져나오는 것처럼 보일 수 있도록 음료매출구 뚜껑 layer를 추가로 제작하였다. 이를 통해 최종적으로 음료자판기(음료배출구)➔음료➔음료배출구뚜껑 순으로 layer dept조절을 하여 음료가 음료배출구뚜껑 밑으로, 음료배출구 위로 지나가도록 애니메이션을 제작하였다.
뿐만 아니라 마우스 입력과 키보드 입력을 위해 새로운 click(), keyboard_input() 함수를 정의하였다. 이 함수에서는 While True 와 if, input함수를 이용하여 마우스와 키보드 입력시에 각각 동전투입과 발로 걷어차는 애니메이션이 나타나도록 구현하였다.

지난 과제와는 다르게 지정 되어있는 프로그램이 아니라 처음부터 끝까지 원하는 시나리오를 제작하고 이에 따른 애니메이션을 직접 제작했다는 것이 더욱 의미가 있었던 것 같다. 뿐만 아니라 애니메이션을 프로그래밍 하면서 끊임없이 마주하는 오류를 확인하고 여러 방안을 찾아보고 실수도 해보고 여러 시도를 해보면서 문제를 해결했다는 뿌듯함이 훨씬 컸다.
그리고 문자와 숫자로 작성한 코딩이 이미지의 형태로 canvas에 바로 나타나 실제로 눈에 보인다는 것이 재미있기도 하였다.
물론 프로그래밍을 하면서 마주치는 수많은 오류와 버그로 인해 지치고 힘들기도 했지만 이러한 문제를 딛고 과제를 해결했기에 더욱 의미가 남는 것 같다. 특히 애니메이션의 특정 상 이미지를 구현하는 과정에서 도형들의 좌표를 찾고 입력하는 과정이 쉽지 많은 않았다. 이러한 어려움과 그 어떤 오류들이 나타났을 때 원인을 찾고 분석하고 계속하여 디버깅을 거치면서 더욱더 의미있는 과제가 되었던 것 같다.