이름: 신중근 학번: 2020213311

**시나리오**

1. 사용자가 프로그램을 실행시킨다

1-1 화면에는 빨간색, 초록색, 파란색 된 바가 있고, 그 위에 거북이가 하나씩 있다.

1-2 배경은 흰색으로 시작한다.

2. 사용자가 거북이 하나를 드래그한다.

2-1 거북이를 위로 이동할수록 해당 색이 진하게, 아래로 이동할수록 해당 색이 연하게 거북이 색이 변한다

3. 사용자가 드래그 한 거북이 위치에 의해 r,g,b 비율을 조정해 배경 색을 변경한다

**함수**

*turtle.과 screen.은 생략*

resizemode(): 크기 조정 모드

auto: pensize 값에 맞춰 거북이의 외관을 조정

user: shapesize로 설정된 stretchfactor와 outlinewidth 값에 따라 거북이의 외관을 조정

noresize: 거북이의 외관 조정이 일어나지 않음

shapesize(stretch\_wid=None, stretch\_len=None, outline=None): 거북이의 크기를 변경

stretch\_wid: 너비

stretch\_len: 길이

outline: 외곽선

pensize(): 선 두께

color([r,g,b]): RGB 값으로 색 설정

speed(): 펜 속도(0~10, 0이 가장 빠름)

left(angle): angle 값만큼 거북이를 왼쪽(반시계)로 회전

left(90): 반시계로 90도 회전

penup(),pu(): 펜이 이동할 때 그리지 않음

goto(x,y): 펜을 좌표(x, y)로 이동

pendown(), pd(): 펜이 이동할 때 그림

sety(y): 펜의 y좌표를 y로 설정

pencolor(): 펜 색상 설정

ondrag(fun, btn=1, add=None): 거북이의 마우스 이동 이벤트에 fun을 연결

fun: 캔버스에서 클릭한 점의 좌표로 호출되는 두 개의 인자가 있는 함수

btn: 마우스 버튼 수, 기본값은 1(마우스 왼쪽 버튼)

add: 새 연결을 추가하고, 그렇지 않으면 이전 연결을 대체

max(): 인자 값들을 비교하여 값이 가장 큰 값을 반환

min(): 인자 값들을 비교하여 값이 가장 작은 값을 반환

max(0, min(y,1)): y<0이라면 0, 0<y<1이라면 y, y>1이라면 1 반환

turtle.ycor(): 거북이의 y 좌표를 반환

fillcolor((r,g,b)): 채우기 색상을 RGB 색상으로 설정

delay(): delay를 밀리초 단위로 설정하는 함수

setworldcoordinates(llx, lly, urx, ury) 사용자 정의 좌표계를 설정

llx: 캔버스 왼쪽 아래 모서리의 x좌표

lly: 캔버스 왼쪽 아래 모서리의 y좌표

urx: 캔버스 오른쪽 위 모서리의 x좌표

ury: 캔버스 오른쪽 위 모서리의 y좌표

hideturtle(): 거북이를 보이지 않게 설정

write(arg, move=False, align='left', font=('Arial', 8, 'normal')): 거북이의 현재 위치에서 텍스트를 기록

arg: TurtleScreen에 기록될 객체

move: True면, 펜이 텍스트의 오른쪽 아래 모서리로 이동, 기본적으로 False

align: "left", "center", "right" 문자열 중 하나, 거북이 위치를 기준으로 글자 위치 설정

font(fontname, fontsize, fonttype): 폰트 설정

if \_\_name\_\_ == "\_\_main\_\_": 인터프리터에서 직접 실행된 경우에만 if문 실행

**코드**

# colormixer

# 시나리오

# 1. 사용자가 프로그램을 실행시킨다

#   1-1 화면에는 빨간색, 초록색, 파란색 된 바가 있고, 그 위에 거북이가 하나씩 있다.

#   1-2 배경은 흰색으로 시작한다.

# 2. 사용자가 거북이 하나를 드래그한다.

#   2-1 거북이를 위로 이동할수록 해당 색이 진하게, 아래로 이동할수록 해당 색이 연하게 거북이 색이 변한다

# 3. 사용자가 드래그 한 거북이 위치에 의해 r,g,b 비율을 조정해 배경 색을 변경한다

from turtle import Screen, Turtle, mainloop

class ColorTurtle(Turtle):

    # 초기화 메소드

    def \_\_init\_\_(self, x, y):

        Turtle.\_\_init\_\_(self)

        self.shape("turtle")

        # resizemode(): 크기 조정 모드

        # auto: pensize 값에 맞춰 거북이의 외관을 조정

        # user: shapesize로 설정된 stretchfactor와 outlinewidth 값에 따라 거북이의 외관을 조정

        # noresize: 거북이의 외관 조정이 일어나지 않음

        self.resizemode("user")

        # shapesize(stretch\_wid=None, stretch\_len=None, outline=None)

        self.shapesize(3, 3, 5)

        # pensize(): 선 두께

        self.pensize(10)

        # private 변수 생성

        self.\_color = [0, 0, 0]

        # 매개변수 x를 self.x에 저장 (x는 0, 1, 2 중 하나)

        self.x = x

        # \_color[x]에 매개변수 y 값을 저장

        self.\_color[x] = y

        # \_color 값을 이용하여 RGB 값으로 색 설정

        self.color(self.\_color)

        # 펜 속도 (0이 가장 빠름)

        self.speed(0)

        # left(angle): angle 값만큼 거북이를 왼쪽(반시계)로 회전

        # left(90): 반시계로 90도 회전

        self.left(90)

        # penup(): 펜이 이동할 때 그리지 않음

        self.pu()

        # 펜을 좌표(x, 0)로 이동

        self.goto(x, 0)

        # pendown(): 펜이 이동할 때 그림

        self.pd()

        # sety(y): 펜의 y 좌표를 y로 설정

        self.sety(1)

        # penup(): 펜이 이동할 때 그리지 않음

        self.pu()

        # sety(y): 펜의 y좌표를 y로 설정

        self.sety(y)

        # 펜 색상 설정

        self.pencolor("gray25")

        # ondrag(fun, btn=1, add=None): 거북이의 마우스 이동 이벤트에 fun을 연결

        # fun: 캔버스에서 클릭한 점의 좌표로 호출되는 두 개의 인자가 있는 함수

        # btn: 마우스 버튼 수, 기본값은 1(마우스 왼쪽 버튼)

        # add: 새 연결을 추가하고, 그렇지 않으면 이전 연결을 대체

        self.ondrag(self.shift)

    def shift(self, x, y):

        # max(): 인자 값들을 비교하여 값이 가장 큰 값을 반환

        # min(): 인자 값들을 비교하여 값이 가장 작은 값을 반환

        # max(0, min(y,1)): y<0이라면 0, 0<y<1이라면 y, y>1이라면 1 반환

        self.sety(max(0, min(y, 1)))

        # turtle.ycor(): 거북이의 y 좌표를 반환

        # 거북이의 y 좌표를 \_color[self.x]에 저장

        self.\_color[self.x] = self.ycor()

        # fillcolor((r,g,b)): 채우기 색상을 RGB 색상으로 설정

        self.fillcolor(self.\_color)

        setbgcolor()

def setbgcolor():

    # 배경색을 RGB 값으로 설정

    # r: red의 y좌표

    # g: green의 y좌표

    # b: blue의 y좌표

    # red, green, blue는 각각 main 함수에서 전역변수로 생성

    screen.bgcolor(red.ycor(), green.ycor(), blue.ycor())

def main():

    # 전역변수 screen, red, green, blue 생성

    global screen, red, green, blue

    screen = Screen()

    # delay를 밀리초 단위로 설정하는 함수

    screen.delay(0)

    # setworldcoordinates(llx, lly, urx, ury) 사용자 정의 좌표계를 설정

    # llx: 캔버스 왼쪽 아래 모서리의 x좌표

    # lly: 캔버스 왼쪽 아래 모서리의 y좌표

    # urx: 캔버스 오른쪽 위 모서리의 x좌표

    # ury: 캔버스 오른쪽 위 모서리의 y좌표

    screen.setworldcoordinates(-1, -0.3, 3, 1.3)

    # 좌표 (0, 0.5)에 RGB값 (0,5, 0, 0) 거북이 생성

    red = ColorTurtle(0, .5)

    # 좌표 (1, 0.5)에 RGB값 (0, 0.5, 0) 거북이 생성

    green = ColorTurtle(1, .5)

    # 좌표 (2, 0.5)에 RGB값 (0, 0, 0.5) 거북이 생성

    blue = ColorTurtle(2, .5)

    # bgcolor(0.5, 0.5, 0.5)

    setbgcolor()

    writer = Turtle()

    # hideturtle(): 거북이를 보이지 않게 설정

    writer.ht()

    # penup(): 펜을 올림, 거북이가 움직여도 그리지 않음

    writer.pu()

    # 거북이를 (1, 1.15) 좌표로 이동

    writer.goto(1, 1.15)

    # write(arg, move=False, align='left', font=('Arial', 8, 'normal')): 거북이의 현재 위치에서 텍스트를 기록

    # arg: TurtleScreen에 기록될 객체

    # move: True면, 펜이 텍스트의 오른쪽 아래 모서리로 이동, 기본적으로 False

    # align: "left", "center", "right" 문자열 중 하나, 거북이 위치를 기준으로 글자 위치 설정

    # font(fontname, fontsize, fonttype): 폰트 설정

    writer.write("DRAG!", align="center", font=(

        "Arial", 30, ("bold", "italic")))

    return "EVENTLOOP"

# 인터프리터에서 직접 실행된 경우에만 아래 코드 실행

if \_\_name\_\_ == "\_\_main\_\_":

    msg = main()

    print(msg)

    # 프로그램이 끝나지 않고 이벤트들이 계속 발생할 수 있도록 계속 실행시키는 함수

    mainloop()

**실행결과**

차트이(가) 표시된 사진

자동 생성된 설명

텍스트, 시계이(가) 표시된 사진

자동 생성된 설명차트이(가) 표시된 사진

자동 생성된 설명차트이(가) 표시된 사진

자동 생성된 설명차트이(가) 표시된 사진

자동 생성된 설명