파이썬 거북이, 변수로 그리는 똑똑한 국기와 오륜기! 📏🚩🌍🐢

등장인물:

- **슬기 선생님:** 친절하고 유머러스한 코딩 선생님
- **지혜:** 똑똑하고 질문이 많은 학생
- **코딩이:** 코딩을 처음 배우지만 호기심 많은 학생

슬기 선생님: 여러분, 안녕하세요! for 반복문 마법으로 정말 다양하고 멋진 그림들을 그려봤죠? 오늘은 우리가 그린 그림들이 더욱 똑똑해지는 방법을 배워볼 거예요. 바로 **변수**를 사용해서 그림의 크기나 위치를 화면 크기에 딱 맞게 조절하는 거랍니다! 프랑스 국기와 올림픽 오륜기를 그리면서 이 비밀을 파헤쳐 볼까요?

코딩이: 변수로 그림을 똑똑하게 만들어요? 그게 뭐예요? 제 거북이가 갑자기 IQ가 높아지는 건가요? 국기랑 오륜기도 그릴 수 있다니, 오늘 king 거북이 완전 국위선양인데요!

슬기 선생님: (웃음) king 거북이가 애국자가 되는 날이네요! 변수를 사용하면, 마치 거북이에게 "화면이 커지면 너도 크게 그려! 화면이 작아지면 너도 작게 그려!" 하고 알려주는 것과 같아요. 화면 크기가 바뀌어도 그림의 비율이 예쁘게 유지된답니다. 먼저 프랑스 국기를 그려볼게요!

1. 변수로 그리는 프랑스 국기 (삼색기)!

```
clearscreen()

screen = Screen()
screen_width = 800  # 화면의 가로 크기를 변수에 저장!
screen_height = 500 # 화면의 세로 크기를 변수에 저장!
screen.setup(screen_width, screen_height) # 변수를 사용해서 화면 크기 설정!

king = Turtle()
king.speed(10)

# 파란색 부분 그리기
king.penup()
# 왼쪽 맨 위 구석으로 이동! (-가로/2, +세로/2)
king.goto(-screen_width/2, screen_height/2)
king.pendown()

king.color("darkblue")
king.begin_fill()
for i in range(2): # 사각형은 두 번 반복하면 그려져요
  king.forward(screen_width/3) # 국기 한 칸의 너비는 화면 가로의 1/3!
  king.right(90)
```

```
    king.forward(screen_height)  # 국기의 높이는 화면 세로 전체!
    king.right(90)
king.end_fill()

# 빨간색 부분 그리기 (가운데 흰색 부분은 남겨두고 이동)
king.penup()
# 파란색 시작점에서 오른쪽으로 (화면 가로의 2/3)만큼 이동하면 빨간색 시작 위치!
king.goto(-screen_width/2 + (screen_width/3)*2, screen_height/2) # 수정된 goto
king.pendown()

king.color("red")
king.begin_fill()
for i in range(2):
    king.forward(screen_width/3) # 국기 한 칸의 너비
    king.right(90)
    king.forward(screen_height)  # 국기의 높이
    king.right(90)
king.end_fill()
```

지혜: 와! screen_width랑 screen_height라는 변수에 숫자를 넣어두고, screen.setup() 할 때도 쓰고, 사각형 그릴 때 king.forward()나 king.goto() 할 때도 막 나눠 쓰고 곱해 쓰고 있어요!

슬기 선생님: 맞아요, 지혜 학생! 이게 바로 오늘 배울 핵심이에요! 만약 우리가 프랑스 국기를 그릴 때, 파란색 부분의 너비를 그냥 king.forward(266) 이렇게 숫자로 딱 정해버렸다고 생각해 보세요. 그런데 나중에 "아, 그림이 너무 작은 것 같아. 화면을 더 크게 하고 싶어!" 해서 screen.setup(1200, 750) 이렇게 화면 크기를 바꾸면 어떻게 될까요?

코딩이: 어… 파란색 부분은 여전히 266만큼만 그려지니까… 화면은 커졌는데 국기는 그대로 작게 남아있을 것 같아요! 비율이 완전 엉망이 되겠는데요?

슬기 선생님: 정확해요! 파란색 부분 너비도 고치고, 높이도 고치고, 빨간색 부분 위치랑 너비도 다~ 새로 계산해서 숫자를 바꿔줘야겠죠? 정말 번거로울 거예요. 하지만 지금 코드처럼 screen_width / 3 이나 screen_height 같은 **변수를 사용해서 비율로 그려주면,** 맨 위에 있는 screen_width = 800 이나 screen_height = 500 이 숫자들만 바꿔주면 국기의 모든 부분들이 알아서 그 새로운 크기에 맞춰 비율대로 척척 그려진답니다! 마치 마법 같죠?

지혜: 아하! 변수를 쓰면 나중에 크기를 바꾸고 싶을 때 맨 위에 있는 변수 값만 고치면 되니까 엄청 편하겠네요! 하나하나 다 안 고쳐도 되고요! 프랑스 국기에서 파란색이랑 빨간색은 그렸는데, 가운데 하얀색은 어떻게 그린 거예요?

슬기 선생님: 좋은 질문이에요! 프랑스 국기는 파랑, 하양, 빨강 순서죠? 우리는 파란색 사각형을 화면 왼쪽에 screen_width/3 너비로 그렸어요. 그리고 king.goto(-screen_width/2 + (screen_width/3)*2, screen_height/2) 명령으로 거북이를 빨간색 사각형이 시작될 위치, 즉 화면 왼쪽에서부터 전체 너비의

2/3만큼 떨어진 곳으로 옮겼죠. 그러면 자연스럽게 파란색과 빨간색 사각형 사이에 screen_width/3 너비만큼의 빈 공간이 생기는데, 거북이 화면의 기본 배경색이 보통 흰색이니까 그 부분이 하얀색 띠처럼 보이게 되는 거랍니다!

코딩이: 와! 일부러 안 그리고 하얀색을 만들다니! 똑똑한데요? 그럼 이제 올림픽 오륜기도 변수로 그려볼까요?

슬기 선생님: 좋아요! 올림픽 오륜기는 다섯 개의 동그란 원이 서로 얽혀있는 모양이죠? 이번에도 화면 크기를 변수로 정하고, 원의 크기나 위치도 그 변수를 이용해서 그려볼게요.

2. 변수로 그리는 올림픽 오륜기!

```
clearscreen()

screen = Screen()
screen_width = 400  # 이번엔 화면 가로 크기를 400으로!
screen_height = 250 # 화면 세로 크기는 250으로!
screen.setup(screen_width, screen_height)

king = Turtle()
king.pensize(screen_width/40) # 펜 굵기도 화면 크기에 비례하게! (예: 400/40 = 10)
king.speed(10)

# 오륜기 색 순서: 파랑, 검정, 빨강 (윗줄) / 노랑, 초록 (아랫줄)
# 원의 반지름도 화면 크기에 비례하게!
ring_radius = screen_width/10 # 예: 400/10 = 40

# 검은색 링 (윗줄 가운데) - 먼저 그려서 기준으로 삼아요!
king.penup()
# 링의 중심이 (0, ring_radius/2) 정도에 오도록 시작 위치 조정
king.goto(0, -ring_radius/2) # 수정된 y좌표: 원의 중심을 고려
king.pendown()
king.color("black")
king.circle(ring_radius)

# 파란색 링 (윗줄 왼쪽)
king.penup()
# 검은색 링 중심에서 왼쪽으로 (반지름*2 + 약간의 간격) 만큼 이동, y좌표는 동일
king.goto(-ring_radius*2.2, -ring_radius/2) # 수정된 x, y좌표
king.pendown()
king.color("blue")
king.circle(ring_radius)
```

```
# 빨간색 링 (윗줄 오른쪽)
king.penup()
king.goto(ring_radius*2.2, -ring_radius/2) # 수정된 x, y좌표
king.pendown()
king.color("red")
king.circle(ring_radius)

# 노란색 링 (아랫줄 왼쪽)
king.penup()
# 파란색 링 중심과 검은색 링 중심 사이의 아래쪽
king.goto(-ring_radius*1.1, -ring_radius*1.5) # 수정된 x, y좌표
king.pendown()
king.color("yellow")
king.circle(ring_radius)

# 초록색 링 (아랫줄 오른쪽)
king.penup()
# 검은색 링 중심과 빨간색 링 중심 사이의 아래쪽
king.goto(ring_radius*1.1, -ring_radius*1.5) # 수정된 x, y좌표
king.pendown()
king.color("green")
king.circle(ring_radius)
```

(선생님이 오륜기 각 원의 위치를 잡는 `goto` 좌표를 좀 더 명확한 기준으로 수정하며 설명합니다. 원래 제공된 코드의 첫 번째 `king.circle`은 위치나 색상 지정 없이 실행되므로 검은색 원으로 해석하고, 이를 기준으로 다른 원들의 위치를 조정하는 논리를 추가합니다.)

지혜: 와! 이번에도 `screen_width`랑 `screen_height` 변수를 쓰고, 원의 반지름(`ring_radius`)이나 펜 굵기(`pensize`)도 `screen_width`를 나눠서 만들었어요! `goto`로 원의 위치를 잡을 때도 `ring_radius`를 곱하거나 나눠서 쓰고 있고요!

슬기 선생님: 맞아요! 오륜기의 각 원들은 크기가 같고, 서로 조금씩 겹쳐져 있죠? 먼저 기준이 되는 검은색 원(윗줄 가운데)을 `king.goto(0, -ring_radius/2)`로 가서 그렸어요. 이렇게 y좌표를 `-ring_radius/2`로 한 것은 원의 중심이 대략 y축의 0 근처에 오도록 하기 위함이에요. (`circle` 함수는 현재 위치에서 왼쪽으로 반지름만큼 떨어진 곳을 중심으로 원을 그리니까요). 그 다음 파란색 원(윗줄 왼쪽)은 검은색 원 중심에서 왼쪽으로 일정한 간격만큼 (`-ring_radius*2.2`) 떨어진 곳에, 빨간색 원(윗줄 오른쪽)은 오른쪽으로 같은 간격만큼 (`ring_radius*2.2`) 떨어진 곳에 그렸어요. y좌표는 검은색 원과 같게 해서 윗줄 세 개의 원 중심이 나란히 있도록 했죠. 아랫줄의 노란색과 초록색 원은 윗줄 원들의 사이사이 아래쪽에 위치해야 하니까, x좌표는 윗줄 원들 사이(`-ring_radius*1.1`, `ring_radius*1.1`), y좌표는 윗줄 원들보다 아래(`-ring_radius*1.5`)로 이동시켜서 그렸답니다. 이렇게 모든 크기와 위치를 `screen_width`나 `ring_radius` 같은 **변수를 기준으로 계산**하면 어떤 장점이 있을까요?

코딩이: 알겠다! 만약에 `screen_width = 400`을 `screen_width = 600`으로 바꾸면, `ring_radius`도 자동으로 커지고, 원들이 찍히는 `goto` 좌표들도 다 알아서 비율대로 멀어지거나 커지니까… 오륜기가 전체적으로 커지면서 모양은 그대로 유지되는 거죠! 숫자를 하나하나 다 안 바꿔도 돼서 짱 편하겠어요!

슬기 선생님: 바로 그거예요! "마법의 숫자"처럼 코드 중간중간에 직접 숫자를 쓰는 대신, 이렇게 변수를 사용하고 그 변수들로 계산을 하면 코드가 훨씬 유연해지고 관리하기 쉬워진답니다. 화면 크기를 바꾸거나 그림의 전체적인 크기를 조절하고 싶을 때, 맨 위에 있는 변수 몇 개만 수정하면 되니까요! 이게 바로 똑똑한 프로그래밍의 한 가지 방법이랍니다!

지혜 & 코딩이: 네! 선생님! 이제부터 그림 그릴 때 꼭 변수를 사용해서 똑똑하게 그려야겠어요! 오늘 정말 유익한 걸 배웠어요!