파이썬의 스마트한 판단력: 함수와 True/False, 그리고 elif를 이용한 선택! 🚦🤔🐢🎨

등장인물:

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

슬기 선생님: 여러분, 안녕하세요! if와 else를 이용해서 파이썬이 두 가지 상황 중 하나를 선택하게 만들었죠? 오늘은 함수가 "맞다/틀리다"(True/False)를 직접 알려주도록 만들어보고, 여러 가지 조건 중에서 하나를 똑똑하게 골라내는 elif 마법도 배워볼 거예요! 그리고 이 모든 마법을 합쳐서 우리가 원하는 도형을 원하는 색깔로 그리는 멋진 프로그램까지 만들어 볼 거랍니다!

코딩이: 와! 함수가 True나 False를 직접 알려준다고요? 그럼 함수한테 "이거 짝수야?" 물어보면 "응, 짝수야! (True)" 이렇게 대답해 주는 거예요? elif는 또 뭐예요? else랑 if가 합체한 건가?

슬기 선생님: (웃음) 코딩이 학생, 예리한데요! 비슷해요! 먼저, 짝수인지 아닌지 알려주는 함수부터 만들어볼까요?

1. "이거 짝수니?" 물어보면 True/False로 답하는 함수!

```
# 짝수 판별하는 함수를 작성하고, 짝수이면 True, 홀수이면 False를 반환하세요.
def is_even(num):      # 'num'이라는 숫자를 받아서
  if num % 2 == 0:   # 만약 num을 2로 나눈 나머지가 0이면 (짝수면!)
    return True      # "응, 맞아! (True)" 라고 대답해!
  else:              # 그렇지 않으면 (홀수면!)
    return False     # "아니야! (False)" 라고 대답해!

# 함수를 불러서 확인해 볼까요?
number_to_check = int(input("짝수인지 확인할 숫자를 입력하세요: "))
if is_even(number_to_check): # is_even 함수의 대답이 True라면
  print(number_to_check, "은(는) 짝수입니다!")
else: # is_even 함수의 대답이 False라면
  print(number_to_check, "은(는) 홀수입니다!")

# print(is_even(21)) # 함수가 직접 True 또는 False를 반환하는지 확인
```

지혜: is_even(num) 함수는 num을 2로 나눈 나머지가 0이면 True를 돌려주고, 아니면 False를 돌려주네요! 그래서 if is_even(number_to_check): 부분에서 함수가 돌려준 True나 False 값으로 바로 조건을 판단할 수 있는 거군요!

슬기 선생님: 정확해요! is_even(21)을 실행하면 21 % 2는 0이 아니니까 else 부분의 return False가 실행되겠죠? 이 함수, 사실 더 간단하게 만들 수도 있어요. num % 2 == 0 이 부분 자체가 이미 True 아니면

False를 만들어내는 **비교 연산**이잖아요?

```
def is_even_simple(num):
  return num % 2 == 0 # num을 2로 나눈 나머지가 0이랑 같은지 바로 물어보고, 그 대답(True/False)을 돌려줘!

# print(is_even_simple(20)) # True가 나오겠죠?
# print(is_even_simple(17)) # False가 나오겠죠?
```

코딩이: 와! if랑 else 없이 그냥 return num % 2 == 0 이렇게만 써도 똑같이 동작해요? num % 2 == 0 이게 맞으면 True가 되고 틀리면 False가 되니까, 그 값을 바로 return하는 거네요! 훨씬 깔끔한데요!

슬기 선생님: 그렇죠? 비교 연산의 결과가 불리언 값이라는 것을 잘 이용하면 코드를 더 간결하게 만들 수 있답니다.

2. elif 마법: 여러 갈래 길에서 하나만 선택하기!

슬기 선생님: 자, 우리가 if와 else로 두 갈래 길에서 하나를 선택했죠? 만약 길이 세 갈래, 네 갈래, 혹은 그 이상이라면 어떻게 할까요? 이럴 때 사용하는 것이 바로 elif 마법이에요! elif는 "else if"의 줄임말로, "만약 첫 번째 조건(if)이 아니라면, **그 대신 만약 이 조건이라면**" 이라는 뜻이랍니다.

성적에 따라 등급을 매기는 프로그램을 예로 들어볼게요. 점수가 60점 미만이면 "F", 60점 이상 70점 미만이면 "D", 70점 이상 80점 미만이면 "C", 80점 이상 90점 미만이면 "B", 90점 이상이면 "A"를 주는 거예요.

```
score = int(input("성적을 입력하세요 (0~100): "))

if score < 60:        # 만약 점수가 60점 미만이라면
  print("F")
elif score < 70:      # (60점 미만은 아니고) 만약 점수가 70점 미만이라면 (즉, 60 <= score < 70)
  print("D")
elif score < 80:      # (70점 미만은 아니고) 만약 점수가 80점 미만이라면 (즉, 70 <= score < 80)
  print("C")
elif score < 90:      # (80점 미만은 아니고) 만약 점수가 90점 미만이라면 (즉, 80 <= score < 90)
  print("B")
else:                 # 위의 모든 조건에 해당하지 않는다면 (즉, 90점 이상이라면)
  print("A")
```

지혜: if, elif, elif, elif, else 순서대로 검사하는군요! 만약 score가 75점이면, score < 60은 False니까 넘어가고, 다음 elif score < 70도 False(75는 70보다 작지 않으니까)라서 넘어가고, 그 다음 elif score < 80은 True(75는 80보다 작으니까)라서 "C"를 출력하고, 그 뒤에 있는 elif나 else는 아예 쳐다보지도 않는 거죠?

슬기 선생님: 정확히 이해했어요! if-elif-else 구조에서는 **조건을 만족하는 첫 번째 블록만 실행**되고 나머지는 건너뛴답니다. 그래서 조건의 순서가 중요해요. 지금처럼 작은 범위부터 차례대로 검사하거나, 큰 범위부터 score >= 90 이런 식으로 검사할 수도 있지만, 지금처럼 <를 사용해서 순서대로 나열하면 각 elif는 그 이전 조건들은 이미 만족하지 않았다는 것을 암묵적으로 포함하게 돼요. 아, 그리고 비교할 때 "같다"는 등호

두 개 == 쓰는 것! 절대 잊으면 안 돼요! = 하나는 변수에 값을 넣는 할당이니까요.

코딩이: 순서가 중요하다니! 만약에 `if score < 90:` 이걸 맨 위에 썼으면, 50점도 "B"가 나올 뻔했어요! (아니죠, 50점은 <60 조건이 없으면 계속 내려가서 else의 A가 나올수도 있거나, 다른 조건에 걸릴 수 있음) 아, 아니구나. 순서대로라면 50점은 `score < 60`에서 "F"가 먼저 나오겠네요!

슬기 선생님: 코딩이 학생, 아주 잘 생각했어요! 맞아요. 지금처럼 작은 값부터 순서대로 조건을 배치하면, 예를 들어 55점은 첫 번째 `if score < 60:`에서 True가 되어 "F"가 나오고 프로그램은 거기서 판단을 마칩니다. 순서가 논리적으로 잘 짜여 있죠.

3. 종합선물세트: 도형과 색깔을 선택해서 그리기!

슬기 선생님: 자, 이제 input(), 함수, if-elif-else를 모두 사용해서 사용자가 원하는 도형을 원하는 색깔로 그리는 프로그램을 만들어 볼까요?

- 사용자에게 모양을 숫자로 물어볼 거예요 (1: 삼각형, 2: 사각형, 3: 원).
- 색깔도 알파벳으로 물어볼 거고요 (y: 노란색, b: 파란색, g: 녹색, r: 빨간색, 그 외에는 모두 보라색).
- 다각형(삼각형, 사각형) 그리는 건 함수로 만들면 좋겠죠?

```
clearscreen()

screen = Screen() # 스크린 설정은 생략할게요!
paul = Turtle()   # paul 거북이 준비!
paul.speed(10)    # 속도 설정!

# 다각형 그리는 함수 정의
def polygon(n_sides): # n_sides는 몇 각형을 그릴지 알려주는 숫자
  paul.begin_fill()
  angle = 360 / n_sides
  for i in range(n_sides):
    paul.forward(100)
    paul.left(angle)
  paul.end_fill()

# 사용자에게 모양과 색깔 입력받기
shape_choice_text = input("1(삼각형), 2(사각형), 3(원) 중에 원하는 모양의 숫자를 입력하세요: ")
color_choice_char = input("y(노란색), b(파란색), g(녹색), r(빨간색) 중에 원하는 색깔의 알파벳을 입력하세

# 색깔 정하기 (if-elif-else 사용)
if color_choice_char == "y":
  paul.color("yellow")
elif color_choice_char == "b":
  paul.color("blue")
```

```
elif color_choice_char == "g":
  paul.color("green")
elif color_choice_char == "r":
  paul.color("red")
else: # 위에 해당 안 되면 무조건 보라색!
  paul.color("purple")

# 모양 그리기 (if-elif-elif 또는 if-elif-else 사용)
if shape_choice_text == "1":   # 입력받은 건 글자니까 "1"과 비교!
  polygon(3)                   # polygon 함수에 숫자 3을 전달해서 삼각형 그리기!
elif shape_choice_text == "2": # "2"와 비교!
  polygon(4)                   # polygon 함수에 숫자 4를 전달해서 사각형 그리기!
elif shape_choice_text == "3": # "3"과 비교!
  paul.begin_fill()
  paul.circle(60)              # 원은 반지름 60으로 그려볼까요?
  paul.end_fill()
else:
  print("선택지에 없는 모양이에요! 그림을 그리지 않아요.")
```

지혜: 와! 정말 멋진 프로그램이에요! `input()`으로 모양이랑 색깔을 글자로 받고, `if-elif-else`로 사용자가 입력한 글자에 따라 거북이 색깔을 정해주네요! 그리고 또 `if-elif-elif`로 사용자가 고른 모양 번호(글자)에 따라서 `polygon()` 함수를 부르거나 `circle()` 함수를 불러서 그림을 그려주고요!

코딩이: 제가 "1"이랑 "y"를 입력했더니, 노란색 삼각형이 그려졌어요! "3"이랑 "아무거나" 입력했더니 보라색 원이 그려졌고요! 대박! `input()`으로 받은 건 글자니까 비교할 때도 `shape_choice_text == "1"` 이렇게 따옴표 안에 있는 글자랑 비교해야 하는군요! `polygon(3)` 이렇게 함수에 숫자를 직접 넣어주는 건 괜찮고요!

슬기 선생님: 아주 정확해요! `input()`은 항상 글자(문자열)를 돌려주니까 비교할 때도 문자열끼리 비교해야 하고, 만약 `polygon` 함수처럼 숫자 정보가 필요한 함수에 전달할 때는 `int()`로 변환해야 하지만, 지금처럼 `polygon(3)`과 같이 함수를 부르는 쪽에서 이미 정확한 숫자를 알고 있다면 바로 숫자를 써도 된답니다. 오늘 우리는 함수가 `True/False`를 반환하는 방법, 여러 조건 중 하나를 선택하는 `elif`, 그리고 이 모든 것을 활용해서 사용자와 소통하며 그림을 그리는 프로그램까지 만들어봤어요! 정말 많은 것을 배웠죠?

지혜 & 코딩이: 네! 선생님! 이제 저희도 파이썬이랑 대화하면서 훨씬 더 다양하고 재미있는 프로그램을 만들 수 있을 것 같아요! 너무 신나요!