<[파이썬 수학] 제9장. for 루프의 슈퍼파워! range와 중첩 루프>

선생님: (더 화려한 마법사 지팡이를 보여주며) 얘들아, 안녕! 지난 시간에 우리는 `total = total + i` 처럼 저금통에 계속 돈을 더하는 '누적 패턴'을 배웠지? 프로그래머들은 종종 이런 코드를 더 짧고 멋지게 쓰는 걸 좋아한단다.

```
# a = a + 3 을 더 짧게!
a = 10
a += 3 # a = a + 3 이라는 뜻이야!
print("a += 3의 결과:", a)

# i = i + 1 도 더 짧게!
i = 10
i += 1
print("i += 1의 결과:", i)
```

코딩새싹 🌱: 와! `a += 3` 이라고 쓰니까 더 간단하고 멋져 보여요! `*=` 나 `-=` 도 똑같이 쓸 수 있겠네요?

선생님: 바로 그거야! 작은 마법이지만 코드를 훨씬 깔끔하게 만들어준단다. 자, 그럼 이제 `for` 반복문의 새로운 단짝 친구를 만나볼까?

<1. 숫자 생성기, range() 함수!>

선생님: 우리가 만약 "안녕!"이라는 말을 5번 반복하고 싶다면 어떻게 해야 할까? `[0, 1, 2, 3, 4]` 같은 리스트를 만들어서 반복할 수도 있겠지만, 숫자가 100번, 1000번이 되면 너무 힘들겠지?

궁금이 💡: 맞아요! 1000개의 숫자가 들어있는 리스트를 직접 만들 순 없잖아요!

선생님: 그래서 파이썬에는 **`range()`** 라는 아주 편리한 '숫자 생성기' 함수가 있단다! `range()`는 우리가 원하는 범위의 숫자들을 순서대로 착착 만들어주는 기계와 같아.

```
# range(5)는 0부터 5 "전"까지, 즉 0, 1, 2, 3, 4 숫자를 만들어내!
for i in range(5):
  print("안녕!", i)
```

선생님: `range()` 기계는 여러 가지 모드가 있어.

- **`range(끝번호)`**: 0부터 끝번호 바로 전까지의 숫자를 만들어줘.

- **range(시작번호, 끝번호)**: 시작번호부터 끝번호 바로 전까지!
- **range(시작번호, 끝번호, 걸음수)**: 시작번호부터 끝번호 전까지, 걸음수만큼 건너뛰면서!

```
# 3부터 21 "전"까지, 3칸씩 건너뛰는 숫자들을 리스트로 만들어볼까?
my_nums = list(range(3, 21, 3)) # list()로 감싸주면 range가 만든 숫자들을 리스트로 볼 수 있어!
print("\n3의 배수 리스트:", my_nums)
```

<2. 반복문에 조건(if) 추가하기!>

선생님: for 반복문 안에 if 조건문을 넣으면, 반복 작업을 하다가 특정 조건을 만족하는 경우에만 특별한 행동을 하게 만들 수 있어! 마치 컨베이어 벨트에 있는 물건들을 하나씩 보다가, '이건 합격!' 하는 물건만 골라내는 것과 같지.

[미션] 1부터 20까지의 정수 중에 짝수만 골라서 리스트를 만드세요!

코딩새싹 🌱: 으음… range(1, 21)로 1부터 20까지 숫자를 만들고, if문으로 짝수인지 검사하면 되겠어요! 어떤 수를 2로 나눈 나머지가 0이면 짝수였죠?

선생님: 완벽해!

```
even_nums = [] # 짝수를 담을 빈 리스트 준비!
for i in range(1, 21): # 1부터 20까지 하나씩 꺼내서,
  if i % 2 == 0: # 만약 i를 2로 나눈 나머지가 0과 같다면 (즉, 짝수라면)
    even_nums.append(i) # 짝수 리스트에 추가해줘!

print("\n짝수 리스트:", even_nums)
```

<3. for 반복문으로 그림 그리기!>

선생님: for 반복문과 range()를 이용하면 이렇게 재미있는 그림도 그릴 수 있단다!

```
# 별표(*) 점점 많아지게 찍기
print("\n--- 계단 만들기 ---")
for i in range(1, 6): # i는 1, 2, 3, 4, 5 로 변하겠지?
  print("*" * i) # 문자열 * 숫자는 반복이었지!

# 별표(*) 점점 적어지게 찍기
```

```
print("\n--- 역계단 만들기 ---")
for i in range(5): # i는 0, 1, 2, 3, 4 로 변한다!
  print("*" * (5 - i))
```

궁금이 💡: 와! range()로 변하는 숫자 i를 이용해서 별의 개수를 조절하는 거군요! 정말 창의적인데요!

〈4. 반복문 속의 반복문! 중첩 for 루프!〉

선생님: 자, 이제 for 반복문의 최종 오의(⊠義), **중첩 for 루프**를 배울 시간이야! 이건 반복문 안에 또 다른 반복문을 넣는 기술이지. 마치 시계의 시침과 분침 같아. 시침(바깥 루프)이 한 칸 움직이는 동안, 분침(안쪽 루프)은 한 바퀴(60번)를 전부 돌지?

```
print("\n--- 중첩 루프 맛보기 ---")
for i in range(3): # 바깥 루프가 0, 1, 2 로 총 3번 돈다.
  print(f"바깥 루프 {i}번째 시작!")
  for j in ["a", "b", "c"]: # 안쪽 루프는 "a", "b", "c"를 돈다.
    print(f"  안쪽 루프: {i}, {j}")
  print(f"바깥 루프 {i}번째 끝!\n")
```

코딩새싹 🌱: 정말 바깥 i가 0일 때, 안쪽 j가 a, b, c를 다 돌고, 그 다음에 i가 1이 되네요!

[최종 도전] 중첩 for 루프로 구구단 출력하기!

선생님: 이 중첩 루프를 이용하면 구구단도 아주 멋지게 만들 수 있어! 바깥 루프는 2단, 3단… 9단을 만들고, 안쪽 루프는 각 단에서 ×1, ×2… ×9를 만들면 되겠지?

```
print("--- 구구단 출력! ---")
for i in range(2, 10): # i는 2단부터 9단까지!
  print(f"\n--- {i}단 ---")
  for j in range(1, 10): # j는 1부터 9까지 곱할 숫자!
    print(f"{i} * {j} = {i * j}")
```

〈실전 종합! 학생들 평균 점수 구하기〉

선생님: 자, 우리가 예전에 다뤘던 학생 성적 리스트야. 이번엔 for 반복문으로 각 학생의 평균을 구해서 리스트에 추가해보자!

```
my_class = [
```

```
    ["이름", "국어점수", "수학점수", "영어점수"], # 제목 행
    ["홍길동", 50, 80, 70],
    ["데카르트", 100, 100, 100],
    ["맹구", 10, 20, 10]
]

# 제목 행을 제외한 학생 정보만 반복해야 하니, 슬라이싱 [1:]을 사용!
for student_data in my_class[1:]:
  # student_data는 ["홍길동", 50, 80, 70] 같은 리스트
  scores = student_data[1:]  # 이름 빼고 점수들만 슬라이싱
  total = sum(scores)        # 점수들의 합
  count = len(scores)        # 과목 수
  average = total / count      # 평균 계산
  student_data.append(round(average, 2)) # 계산된 평균을 원래 학생 리스트에 추가!

print("\n--- 평균 점수 추가 완료! ---")
print(my_class)
```

궁금이 💡: 와! `for student_data in my_class[1:]:` 이 부분에서 `[1:]`을 써서 첫 번째 제목 행은 건너뛰고, 각 학생의 리스트(`student_data`)를 하나씩 꺼내서 그 안에서 또 슬라이싱(`[1:]`)으로 점수만 뽑아내고… 정말 배운 걸 다 쓰는 느낌이에요!

선생님: (활짝 웃으며) 맞아! 오늘 배운 `range` 함수, `if` 조건문, 중첩 루프까지! 이 `for` 반복문의 슈퍼파워들을 자유자재로 쓸 수 있게 된다면, 여러분은 정말 복잡하고 어려운 문제도 척척 해결하는 멋진 프로그래머가 될 수 있을 거야! 오늘 정말 어려운 내용을 배우느라 수고 많았어! 🌟