제어문 (1)

PDF	비어 있음
비고	조건문
숫자	8
실습문제	비어 있음
실습문제답안	비어 있음

01. 조건문

01-01. 조건문 개요

01-01-01. 조건문이란

💡 조건에 따라 수행할 코드를 다르게 작성하는 방법으로, 파이썬 키워드를 사용하여 작성한다. 조건문 작성에는 조건식이 필요하며, 조건식은 True 혹은 False의 결과를 내는 구문을 의미한다.

01-02. 조건문 if

01-02-01. if

- if 키워드 뒤의 조건식의 결과 값이 참(True)이면 실행 구문을 실행한다.

```
if 조건식: (실행 구문)
```

  - 파이썬에서는 조건식 뒤에 콜론을 붙이고, "반드시" 들여쓰기(공백 4칸 또는 탭)로 실행 구문을 식별할 수 있도록 작성해야 한다.

- 예시

```python
my_teacher = 'tiger' if my_teacher == 'tiger' : print('우리 선생님은 호랑이 선생님입니다!') # 우리 선생님은 호랑이 선생님입니다!
```

```python
my_teacher = 'deer' if my_teacher == 'tiger' : print('우리 선생님은 호랑이 선생님입니다!') # 실행되지 않음
```

01-02-02. else (if-else)

- if 키워드 뒤의 조건식의 결과 값이 참(True)이면 실행 구문 1을 실행하고,

  조건식의 결과 값이 거짓(False)이면 else 키워드 하위의 실행 구문 2를 실행한다.

```
if 조건식: (실행 구문 1) else: (실행 구문 2)
```

  - 이때도 조건식에 참인 경우와 거짓인 경우 수행할 내용을 구분할 수 있도록 "반드시" 들여쓰기 해주어야 한다.

- 예시

```
my_teacher = 'tiger' if my_teacher == 'tiger' : print('우리 선생님은 호랑이 선생님입니다!') # 실행됨 else: print('우리 선생님은 호랑이 선생님이 아닙니다!') # 실행되지 않음
```

```
my_teacher = 'monkey' if my_teacher == 'tiger' : print('우리 선생님은 호랑이 선생님입니다!') # 실행되지 않음 else: print('우리 선생님은 호랑이 선생님이 아닙니다!') # 실행됨
```

01-02-03. elif (if-elif-else)

- if 키워드 뒤의 조건 1의 결과 값이 참(True)이면 실행 구문 1을 실행하고,

  조건 1의 결과 값이 거짓(False)이면서 조건 2의 결과 값이 참(True)이면 실행 구문 2를 실행하고,

  조건 1과 조건 2의 결과 값이 모두 거짓(False)이면서 조건 3의 결과 값이 참(True)이면 실행 구문 3을 실행한다.

  조건 1, 조건 2, 조건 3의 결과 값이 모두 거짓(False)이면 else 키워드 하위의 실행 구문 4를 실행한다.

  ⇒ elif 로 다른 조건식을 제시하여 여러 개의 조건 중 한 가지를 반드시 실행시켜야 할 때 사용한다.

```
if 조건식 1: (실행 구문 1) elif 조건식 2: (실행 구문 2) elif 조건식 3: (실행 구문 3) . . . else: (실행 구문 4)
```

  - elif는 1개 이상 사용할 수 있다.
  - 이때도 각각의 경우에 수행할 내용을 구분할 수 있도록 "반드시" 들여쓰기 해주어야 한다.

- 예시

```
my_teacher = 'tiger' if my_teacher == 'tiger': print('우리 선생님은 호랑이 선생님입니다!') # 실행됨 elif my_teacher == 'monkey': print('우리 선생님은 원숭이 선생님입니다!') # 실행되지 않음 else: print('우리 선생님은 호랑이도 원숭이도 아닙니다. 누구시죠?!') # 실행되지 않음
```

```
my_teacher = 'monkey' if my_teacher == 'tiger': print('우리 선생님은 호랑이 선생님입니다!') # 실행되지 않음 elif my_teacher == 'monkey': print('우리 선생님은 원숭이 선생님입니다!') # 실행됨 else: print('우리 선생님은 호랑이도 원숭이도 아닙니다. 누구시죠?!') # 실행되지 않음
```

```
my_teacher = 'bear' if my_teacher == 'tiger': print('우리 선생님은 호랑이 선생님입니다!') # 실행되지 않음 elif my_teacher == 'monkey': print('우리 선생님은 원숭이 선생님입니다!') # 실행되지 않음 else: print('우리 선생님은 호랑이도 원숭이도 아닙니다. 누구시죠?!') # 실행됨
```

[참고] 삼항연산자 (Ternary operator 혹은 Conditional operator)

```
# 참일때 값 if 조건식 else 거짓일때 값 print("참" if True else "거짓") print("참" if False else "거짓")
```

```
num = int(input('정수 입력 : ')) print('홀수' if num % 2 != 0 else '짝수')
```

```
a = int(input('첫번째 정수 입력 : ')) b = int(input('두번째 정수 입력 : ')) big = a if a > b else b small = a if a < b else b print(big, small)
```

01-03. 조건문 match

01-03-01. match-case

(자바의 swtich-case 와 유사한 조건문이다.)

- 주어진 값을 case 블록의 값과 비교해 일치하는 case만 실행한다.
  - 즉, match 키워드 뒤에 붙은 변수의 값이나 값 자체가 case 키워드 뒤의 값과 같을 경우, 해당 블럭의 실행문을 수행한다.
  - 마지막에 case _ 블록의 실행문3은 위의 모든 케이스에 해당하지 않는 경우 수행하는 내용이 된다.

```
match 변수 or 값 : case 값1: 실행문1 case 값2: 실행문2 . . . case _: # 다른 조건들을 모두 제외한 값을 의미함 실행문3
```

⇒ 조건을 따지고자 하는 경우가 많을 때, if-elif-else 보다 가독성이 높고 간결하게 작성할 수 있다.

- 예시

```
print("=== OhGiraffers vending machine ===") print("tiger: 우리는 1인 24시간 교대제로 간다") print("monkey: 24시간 문의 환영, 다른 말로 하면?") print("deer: 저도 하루에 너다섯 시간 자는데요?") print("rabbit: 살아남는 사람이 강한 사람입니다.") print("==========================") print("만나고 싶은 선생님을 입력해 주세요") input_teacher = input() print("==========================") sleep_time = 0 match input_teacher: case "tiger": print("호랑이 선생님의 메타버스에 오신 것을 환영합니다.") sleep_time = 3 case "monkey": print("원숭이 선생님의 밀림에 오신 것을 환영합니다.") sleep_time = 3.5 case "deer": print("사슴 선생님의 매력의 늪에 오신 것을 환영합니다.") sleep_time = 5 case "rabbit" : print("토끼 선생님의 다정보스 랩에 오신 것을 환영합니다.") sleep_time = 5.2 case _: print("공부할 마음이 없으신 건 아니죠?") print("매일 " + str(sleep_time) + "시간의 수면이 예상됩니다.") # print("매일", sleep_time, "시간의 수면이 예상됩니다.")
```