



DAS 사전교육

- Python 기초 -



기본 개념1



- ▶ 1. 변수
- ▶ 2. 연산자
- ▶ 3. 표준 입출력
- ▶ 4. 선택문
- ▶ 5. 반복문과 기타 제어문
- ▶ 6. 함수



4. 선택문



- ▶ 제어문은 조건식에 만족하는 동안 특정 구간을 한 번 또는 여러 번 반복하여 실행하도록 제어하는 문장
- ▶ 선택문은 조건식이 만족하면 한 번 실행, 만족하지 않으면 해당 구간을 건너뛰게 된다
- ▶ if문을 이용하여 조건식의 결과가 True인지 False인지에 따라 코드를 실행하거나 실행하지 않는 선택적 구조



4. 선택문



▶ 규칙

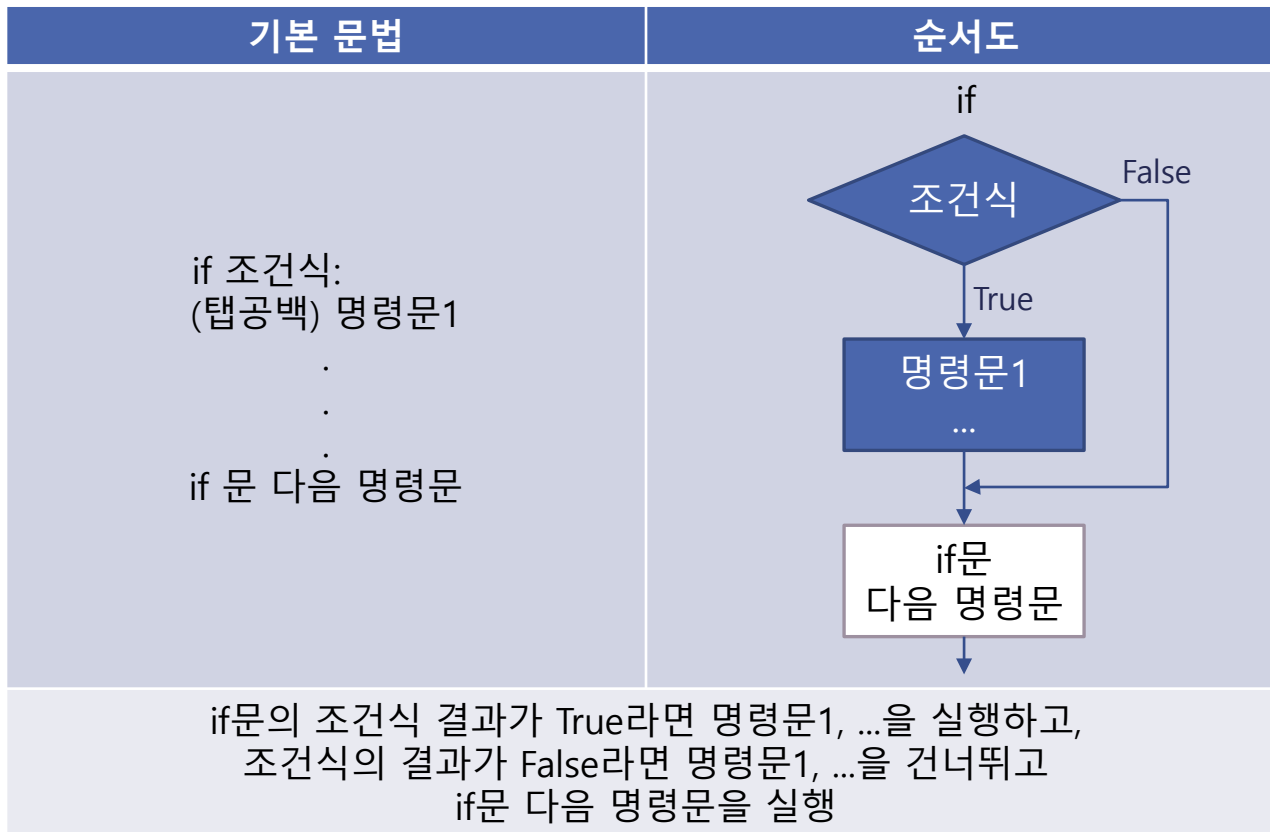
- ▶ if문의 조건식들은 논리 연산이 가능한 문장으로 참 거짓 판별이 가능하도록 작성
- ▶ if문의 조건식 끝에는 항상 콜론(:)이 있어야 한다
- ▶ 실행 코드는 반드시 공백(스페이스바 또는 탭)으로 들여쓰기(indent) 하여
if문에 포함되는(종속) 코드로 작성



4. 선택문



▶ if

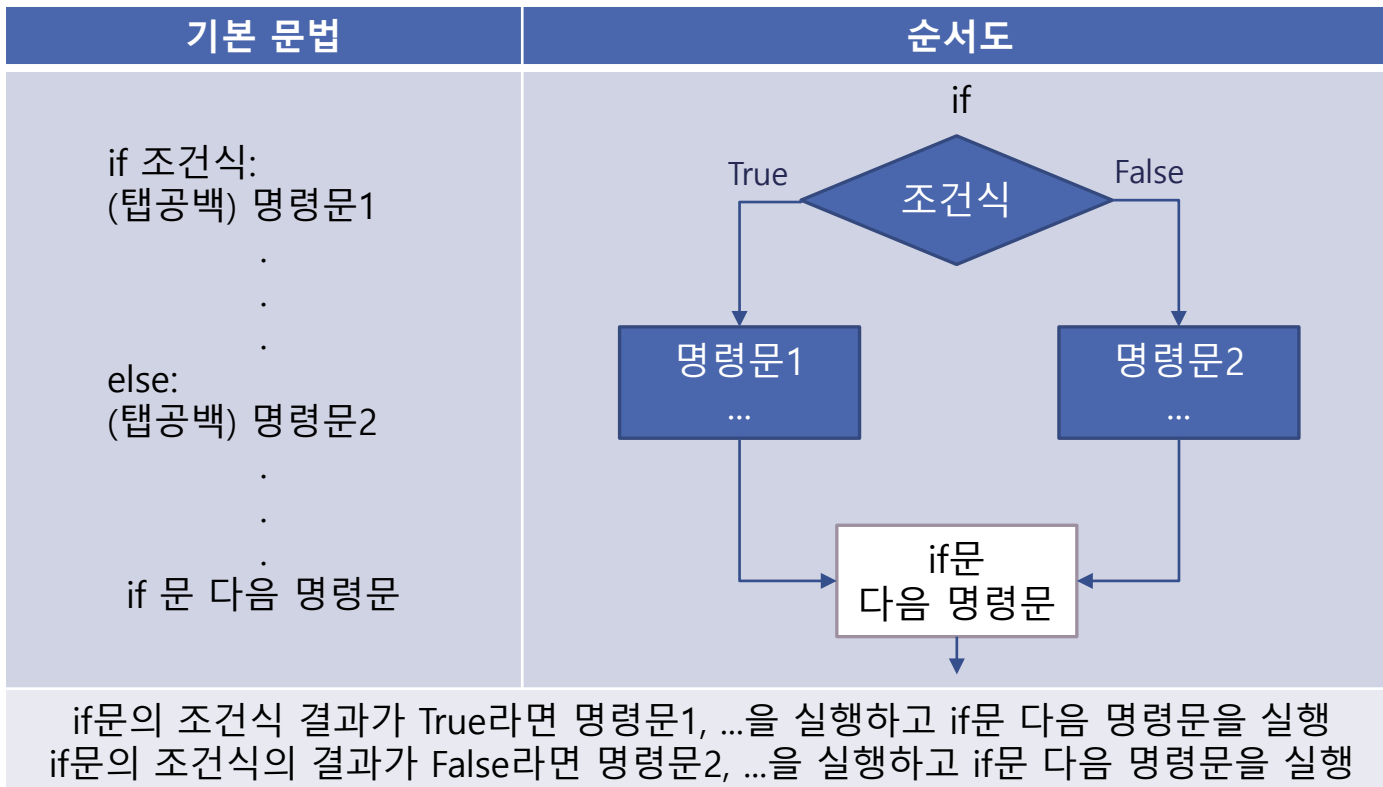




4. 선택문



▶ if~else

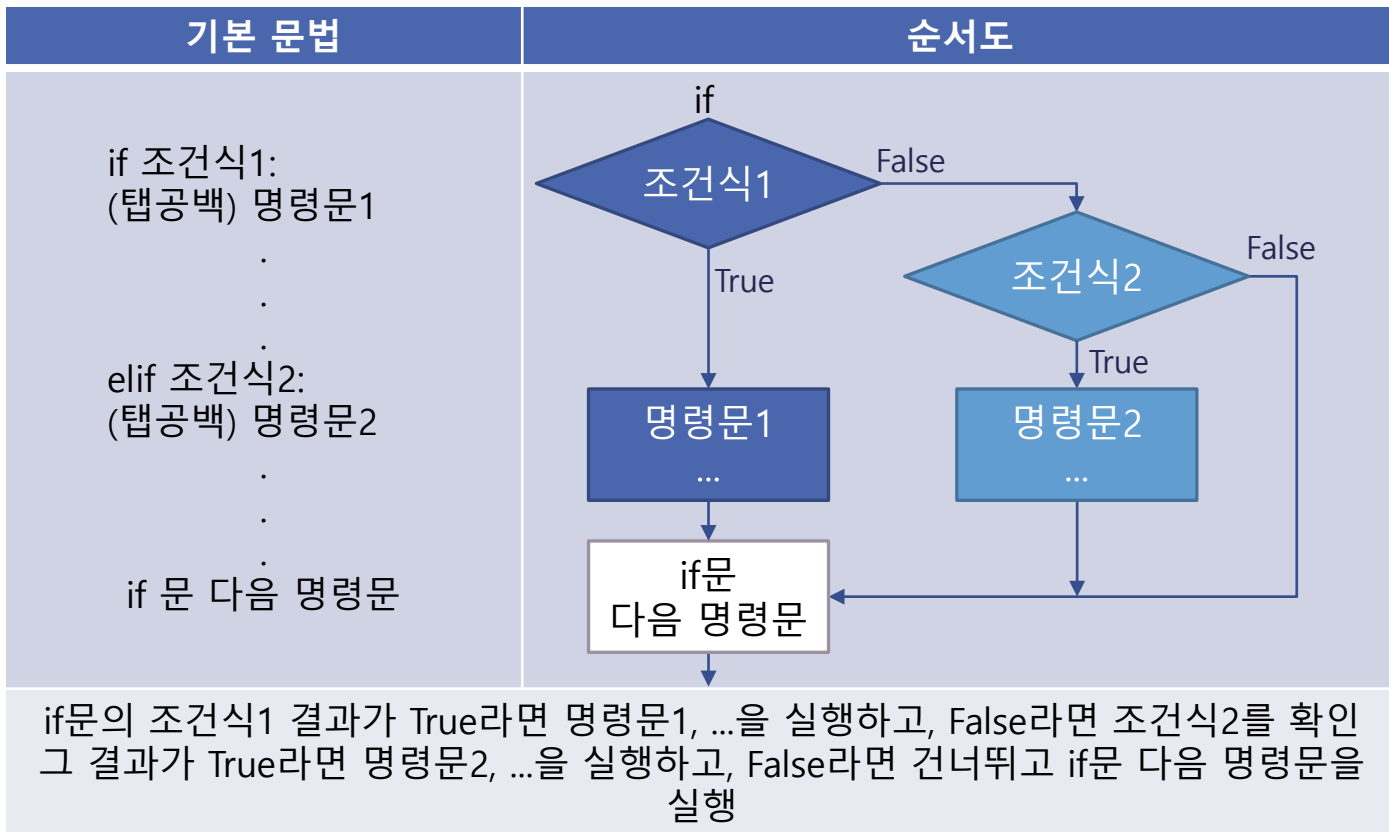




4. 선택문



▶ if~elif

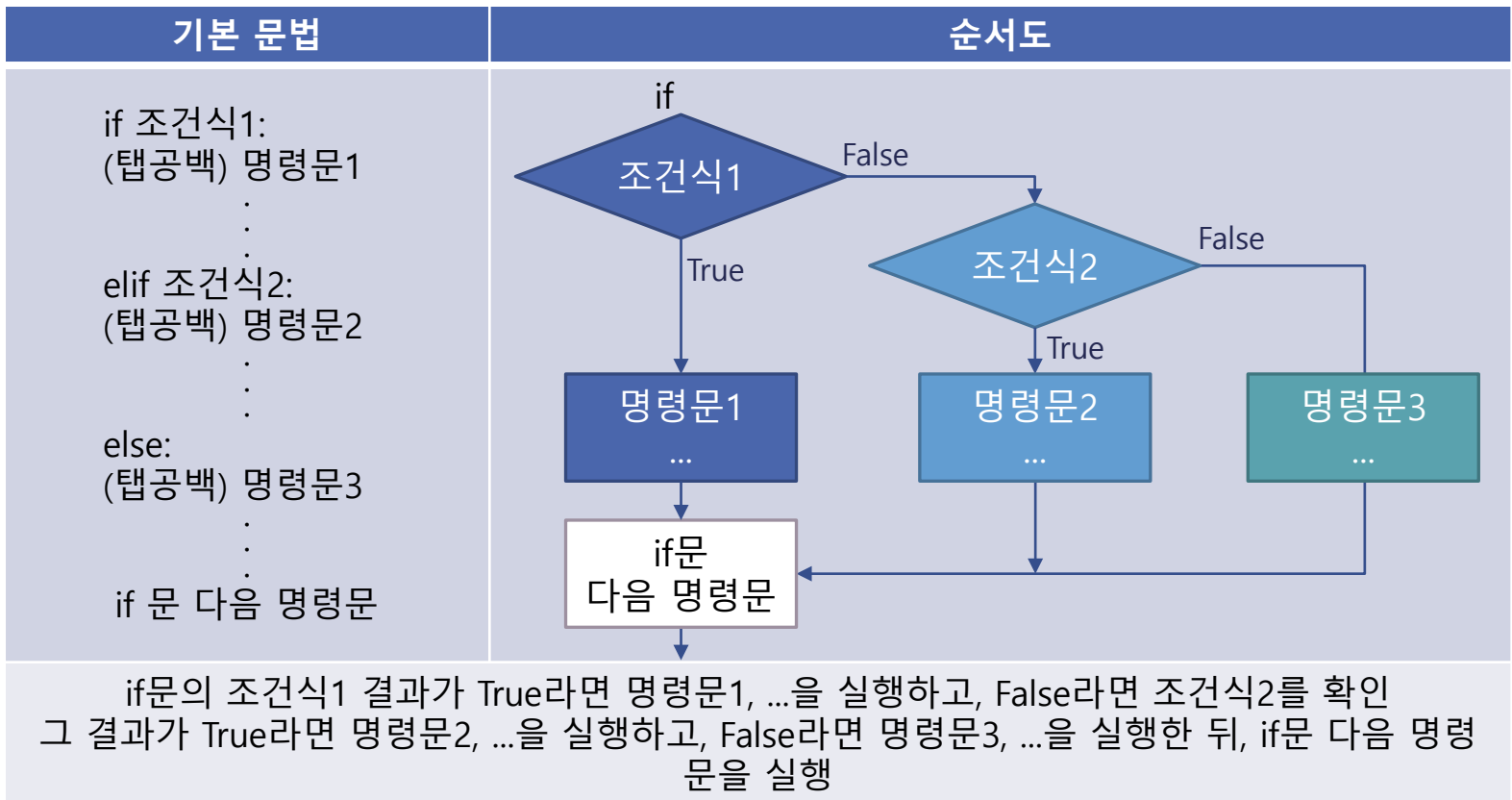




4. 선택문



▶ if~elif~else

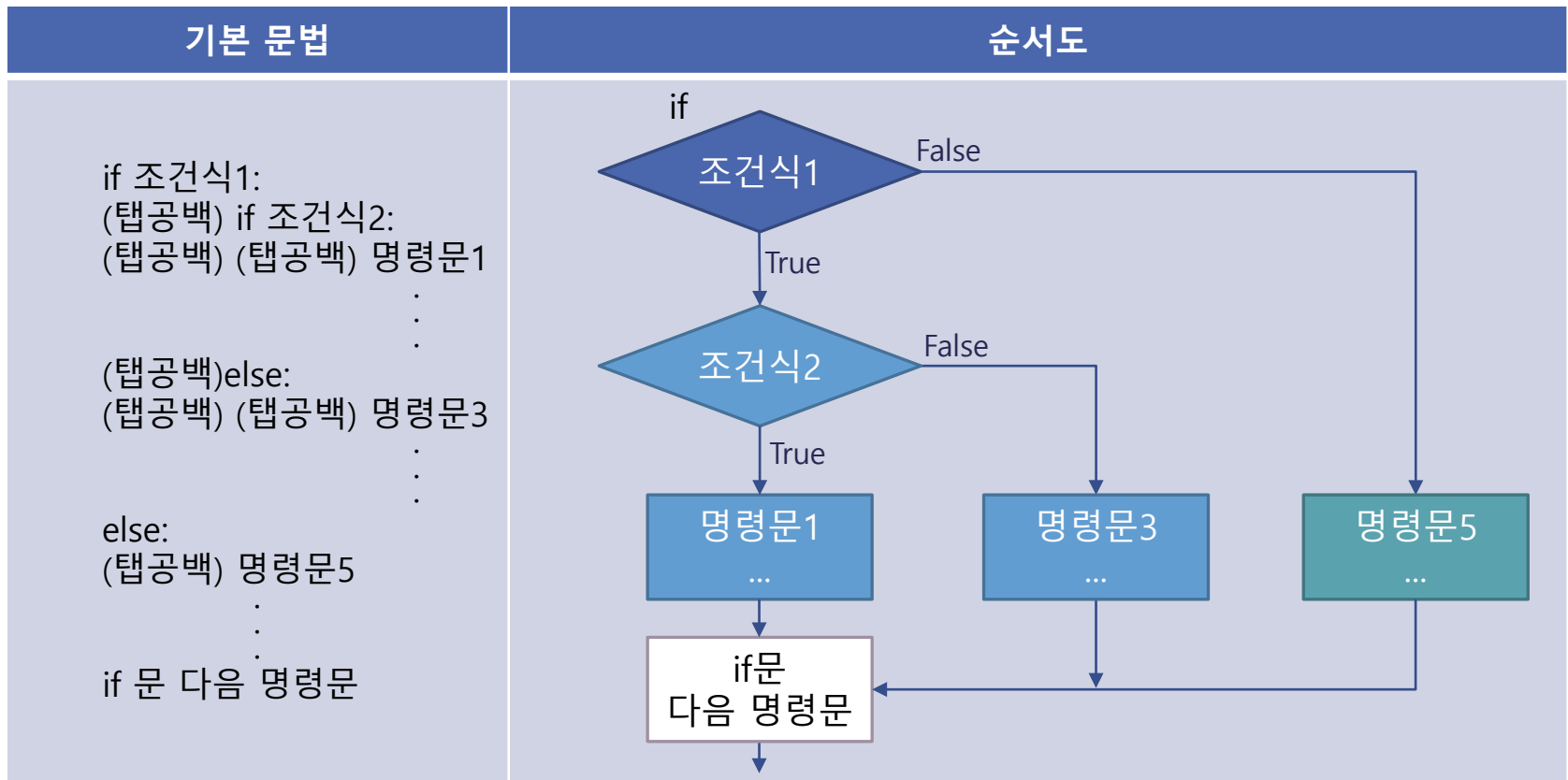




4. 선택문



▶ 중첩 선택문 - if문 안에 또 다른 if문

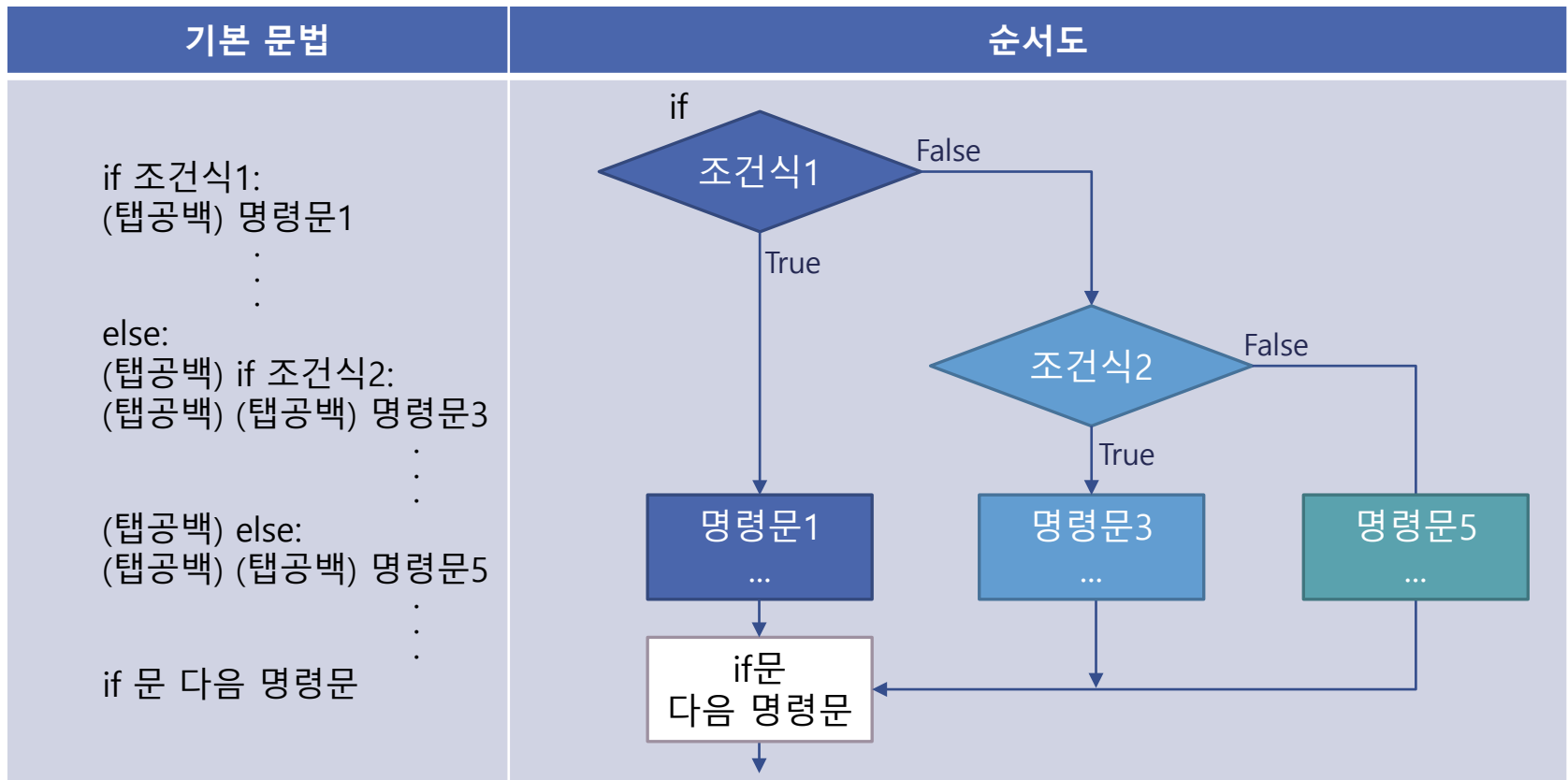




4. 선택문



- ▶ 중첩 선택문 - else문 안에 또 다른 if문





5. 반복문과 기타 제어문



- ▶ 반복문은 특정 문장을 여러 번 반복하여 실행하는 반복적 구조
- ▶ 범위의 횟수 동안 또는 조건식이 만족하는 동안 여러 번 반복하여 실행하는 제어문
- ▶ for문은 범위의 횟수 동안 특정 문장을 여러 번 반복하여 실행하는 반복문
- ▶ while문은 조건식의 결과가 True이면 특정 문장을 여러 번 반복 실행하다가 조건식이 False가 되면 반복을 종료하는 반복문



5. 반복문과 기타 제어문



▶ 규칙

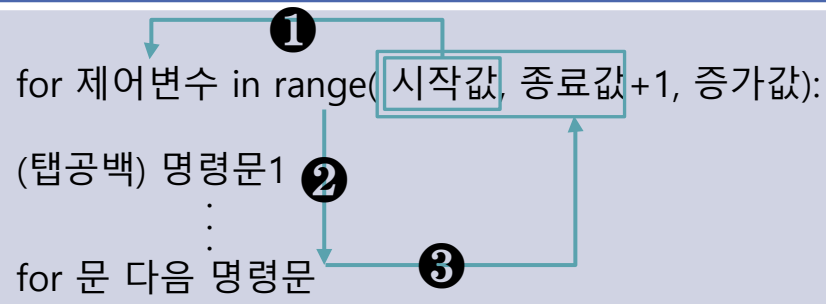
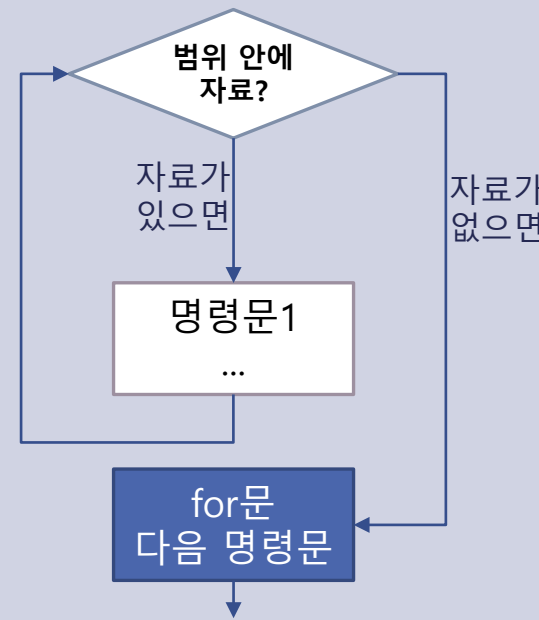
- ▶ for문과 while문의 조건식 끝에는 항상 콜론(:)이 있어야 한다
- ▶ 실행 코드는 반드시 공백(스페이스바 또는 탭)으로 들여쓰기(indent) 하여

for문과 while문에 포함되는(종속) 코드로 작성

for문의 규칙	while문의 규칙
<ul style="list-style-type: none">✓ 제어변수와 특정 범위를 활용하여 반복 실행✓ 특정 범위는 range() 함수, 리스트, 문자열 등을 이용하여 지정	<ul style="list-style-type: none">✓ 조건식의 결과가 참 거짓인지에 따라 특정 문장을 반복 실행하거나 실행하지 않는 조건적 반복 실행 구조✓ while문의 조건식들은 논리 연산이 가능한 문장, 즉 참 거짓 판별이 가능한 것이어야 한다

5. 반복문과 기타 제어문

▶ for

기본 문법	순서도
 <pre> for 제어변수 in range(시작값, 종료값+1, 증가값): (탭공백) 명령문1 ... for 문 다음 명령문 </pre>	 <pre> for 변수 in 범위 범위 안에 자료? - 자료가 있으면: 명령문1, ..., for문 다음 명령문 - 자료가 없으면: for문 다음 명령문 </pre>
<pre> for 제어변수 in 리스트: (탭공백) 명령문1 ... for 문 다음 명령문 </pre>	
<pre> for 제어변수 in '문자열': (탭공백) 명령문1 ... for 문 다음 명령문 </pre>	

5. 반복문과 기타 제어문

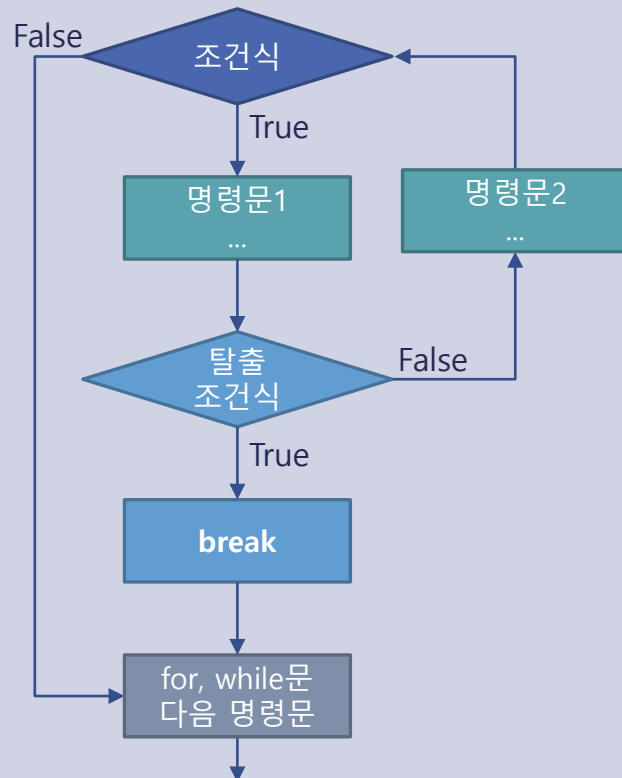
▶ while

기본 문법	순서도
<p>while 조건식: (탭공백) 명령문1 ⋮ while 문 다음 명령문</p>	<p>while</p> <pre> graph TD Entry(()) --> Cond{조건식} Cond -- True --> Body[명령문1 ...] Body --> Entry Cond -- False --> Exit[while문 다음 명령문] Exit --> Exit </pre>
<p>while 조건식: (탭공백) 명령문1 ⋮ else: (탭공백) 명령문2 ⋮ while 문 다음 명령문</p>	<p>while</p> <pre> graph TD Entry(()) --> Cond{조건식} Cond -- True --> Body1[명령문1 ...] Body1 --> Entry Cond -- False --> Body2[명령문2 ...] Body2 --> Exit[while문 다음 명령문] Exit --> Exit </pre>

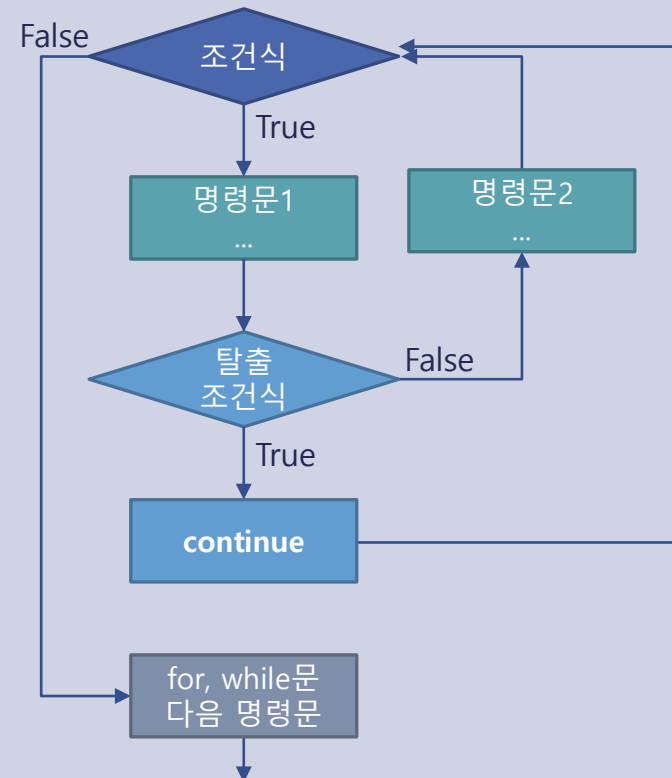
5. 반복문과 기타 제어문

▶ 기타 제어문

break 문을 이용한 강제 종료

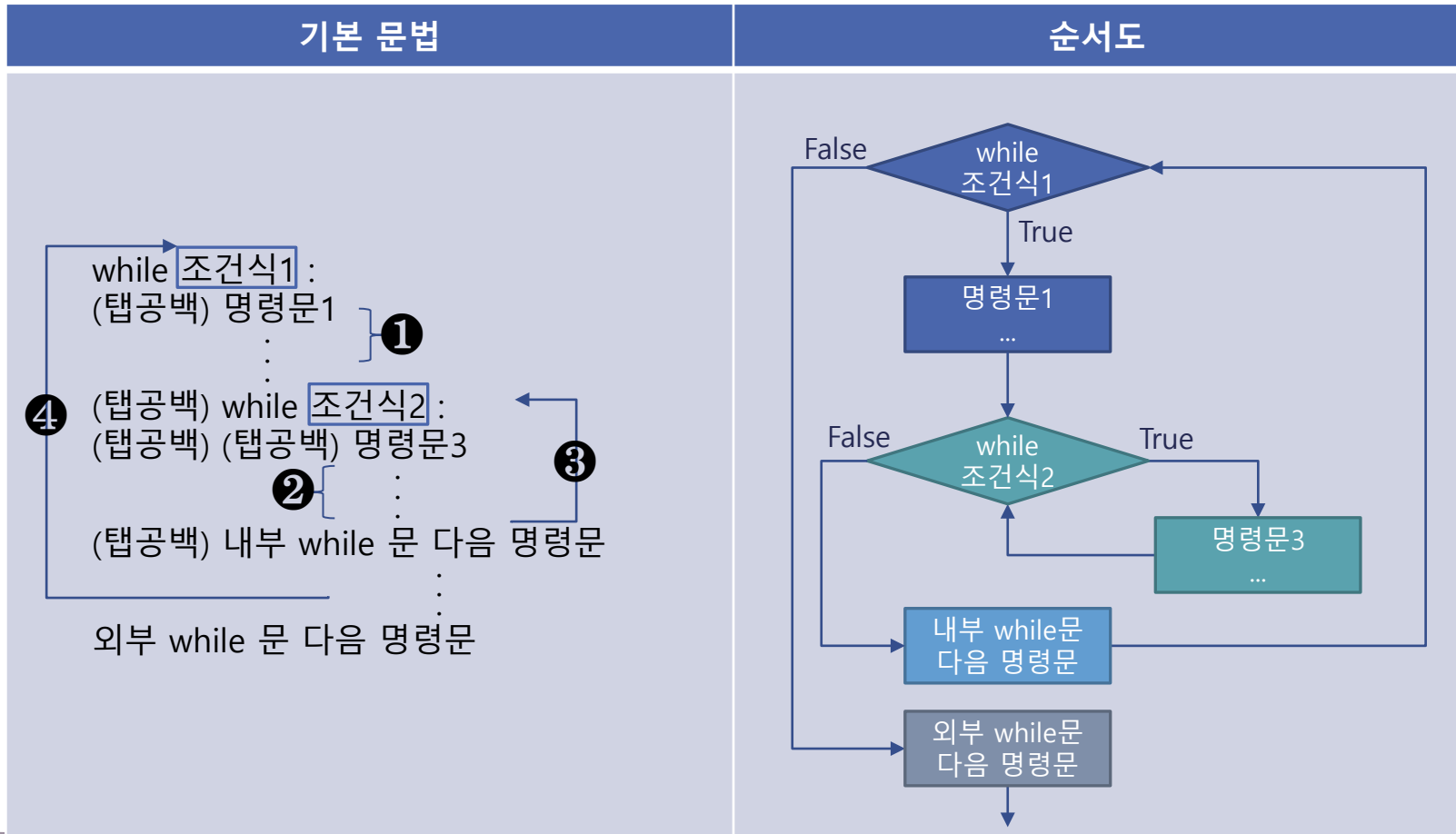


continue 문을 이용한 반복 구간 건너 띄기



5. 반복문과 기타 제어문

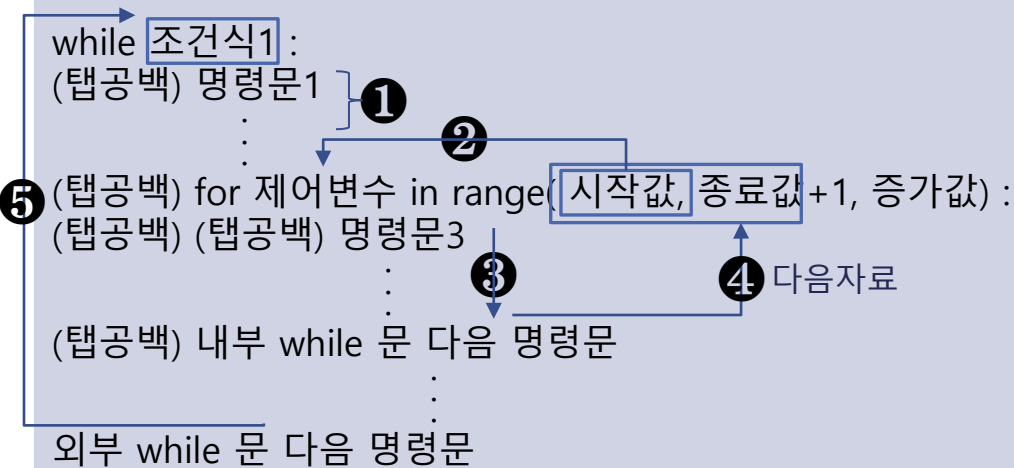
중첩 반복문



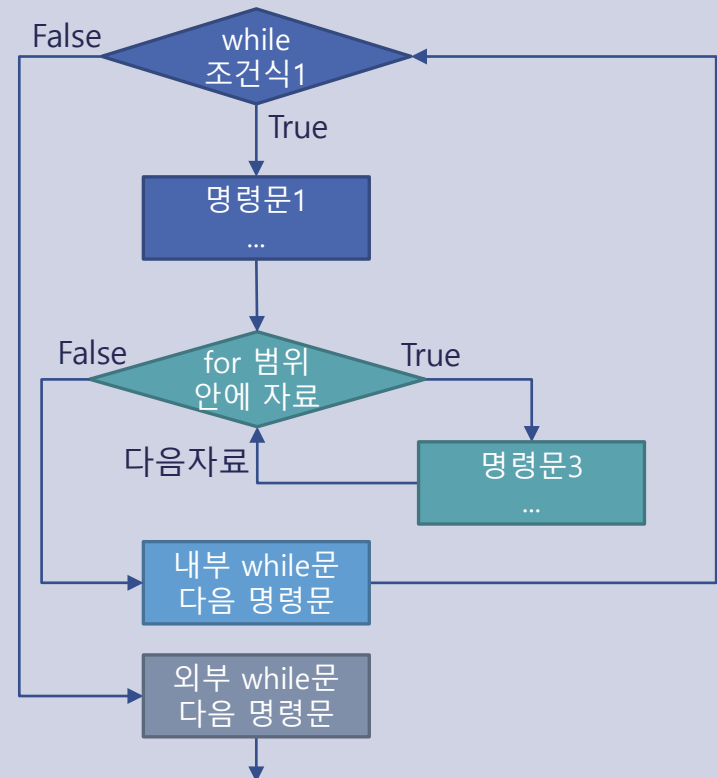
5. 반복문과 기타 제어문

중첩 반복문

기본 문법



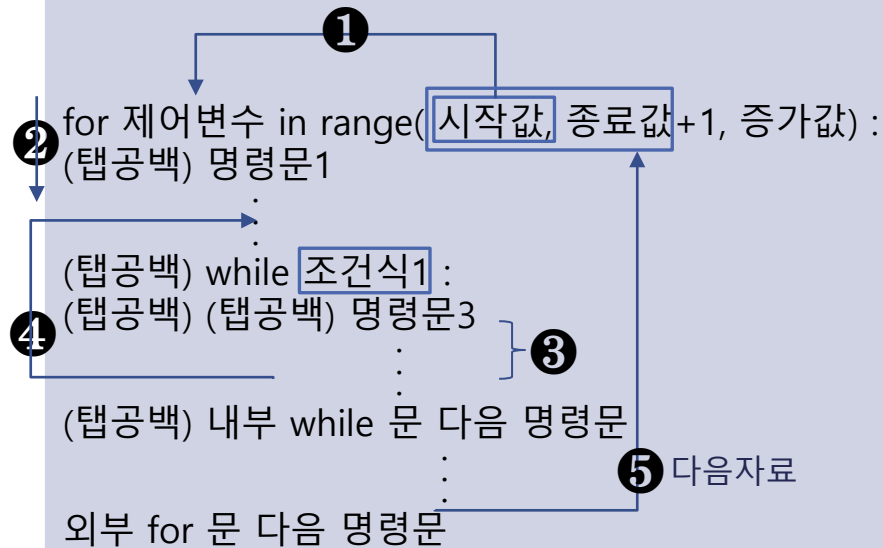
순서도



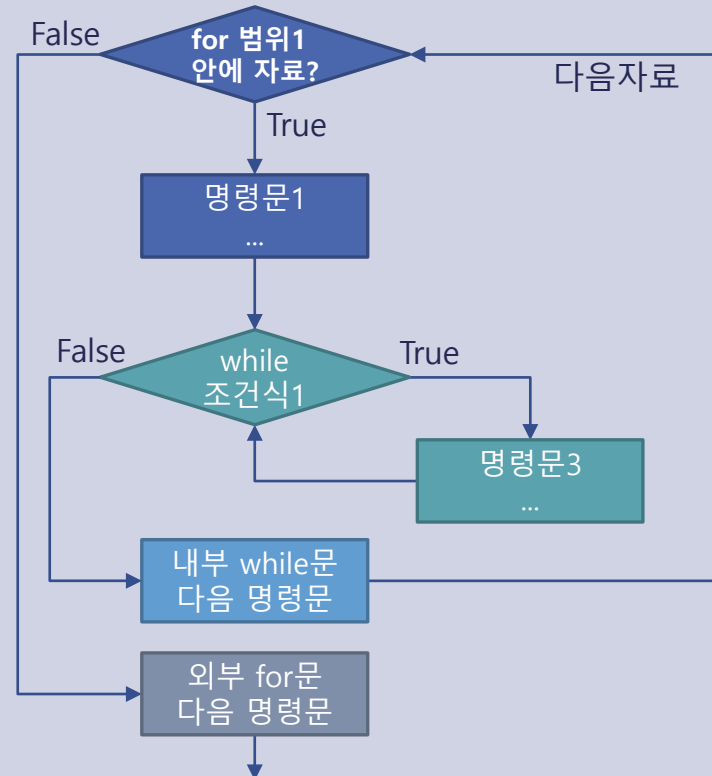
5. 반복문과 기타 제어문

중첩 반복문

기본 문법



순서도



5. 반복문과 기타 제어문

중첩 반복문

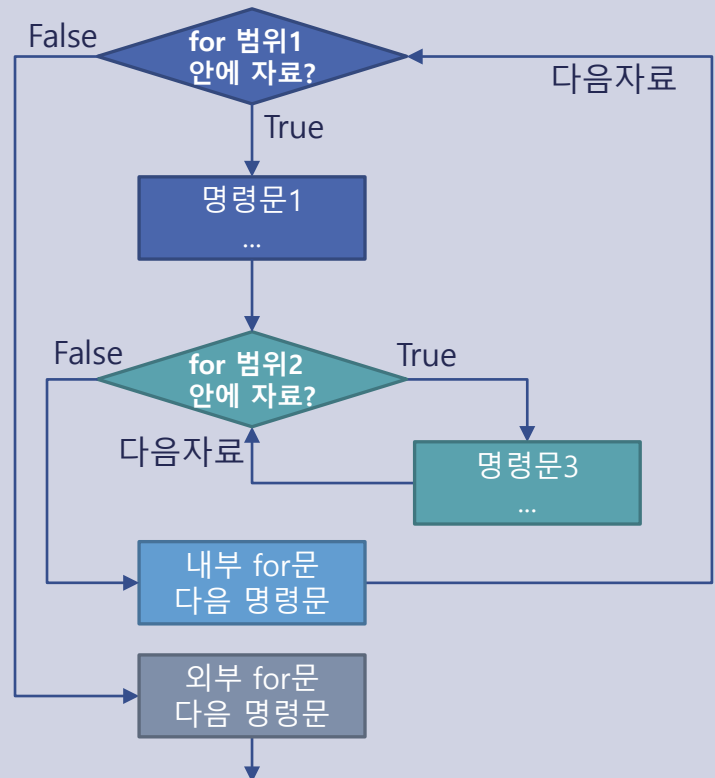
기본 문법

```

1 for 제어변수1 in range(시작값, 종료값+1, 증가값):
2     (탭공백) 명령문1
3     ...
4     (탭공백) for 제어변수2 in range(시작값, 종료값+1, 증가값):
5         (탭공백) (탭공백) 명령문3
6         ...
        (탭공백) 내부 for 문 다음 명령문
        ...
        외부 for 문 다음 명령문
  
```

⑤ 다음자료
⑥ 다음자료

순서도





6. 함수



- ▶ 함수: 특정한 작업을 위해 서로 관련되어 있는 일련의 명령문
- ▶ 함수를 사용하는 이유
 - ▶ 프로그램에서 좋은 코드란 중복성이 없는 코드
 - ▶ 중복된 코드를 별도로 분류하여 함수로 만들게 되면 코드의 양을 줄일 수 있고 가독성이 좋아진다
 - ▶ 간결해진 코드는 전체의 기능을 이해하기에 수월하다
 - ▶ 프로그램의 흐름을 파악하기 쉽기 때문에 관리하기 쉽다
 - ▶ 중요한 기능만 함수로 정의해두면 재사용이 가능하여 매우 효율적으로 사용할 수 있다



6. 함수



- ▶ 내장 함수
 - ▶ 프로그램에 이미 지정된 함수
- ▶ 사용자 정의 함수
 - ▶ 사용자가 필요에 의해서 직접 정의하는 함수



6. 함수



- ▶ 내장 함수
 - ▶ 1) 타입 변환 함수: `int()`, `float()`, `str()`
 - ▶ 2) 수학 함수
 - ▶ 3) Random Numbers



6. 함수



▶ 사용자 정의 함수

함수의 기본 구조

```
def 함수이름(매개변수1, 매개변수2,...):  
    문장1  
    ...  
    return 결과값
```



6. 함수



▶ 람다(lambda) 함수

▶ 1) 람다 표현식 작성

람다(lambda) 함수의 기본 구조

람다(lambda) 매개변수들: 식

▶ 2) 람다 표현식을 인수로 사용