

파이썬 프로그래밍

14차시

임의의 수인 난수와
반복을 제어하는
break문, continue문



⚠ 학습개요

- … 난수를 위한 모듈 random
- … 모듈 사용 방법 구문
- … break와 continue문

⚠ 학습목표

- … 난수를 위한 random 모듈의 함수 randint()를 사용할 수 있다.
- … 구문 import와 from 구문을 사용할 수 있다.
- … 반복에서 break와 continue를 활용할 수 있다.

Chapter 1.

난수를 위한 모듈 random

P Y T H O N P R O G R A M M I N G

⚠ 임의의 수를 발생하는 난수

+ 모듈 random의 함수 randint(시작, 끝)

- 시작과 끝 수 사이에서 **임의의 정수를 반환**
 - 주사위는 1에서 6까지 하나의 수를 정할 수 있다. 주사위를 사용하면 신이 아닌 이상 누구도 다음에 어떤 수가 나올지 예측할 수 없다.
 - 바로 주사위에서 결정되는 수처럼 **난수(random number)**는 주어진 범위의 수 중에서 예측할 수 없는 **임의의 수**를 말한다.
 - 난수는 여러 다각형 주사위나 로또 복권의 추첨 기계처럼 기기를 이용해 만들 수도 있고, 아파트 분양처럼 많은 당첨 번호가 필요할 때는 주로 컴퓨터를 사용해 만들 수 있다.

⚠ 임의의 수를 발생하는 난수

+ 모듈 random의 함수 randint(시작, 끝)

- 시작과 끝 수 사이에서 임의의 정수를 반환

```
>>> import random
>>> random.randint(1, 3)
3
>>> random.randint(1, 3)
1
```

```
>>> import random
>>> for i in range(6):
...     n = random.randint(1, 45)
...     print(n, end=' ')
...
35 30 3 38 39 4
```


Chapter 2.

모듈 사용 방법 구문

P Y T H O N P R O G R A M M I N G

⚠ 일상 코딩: 로또 복권 모의 실험

[코딩실습] 1에서 45까지의 6개 수를 맞추는 로또

난이도 응용

```
1. winnumber = 11, 17, 28, 30, 33, 35
2. print(' 모의로또 당첨번호 '.center(28, '='))
3. print(winnumber)
4. print()
5. print(' 내 번호 확인 '.center(30, '-'))
6. cnt = 0
7.
8. import random
9. for i in range(6):
10.     n = random.randint(1, 45)
11.     if n in winnumber:
12.         print(n, 'O ', end = ' ')
13.         cnt += 1
14.     else:
15.         print(n, 'X ', end = ' ')
16.
17. print()
18. print(cnt, '개 맞음')
```

⚠ 일상 코딩: 로또 복권 모의 실험

주의 print()는 빈 줄을 삽입하기 위해 사용한다.

결과

==== 모의 로또 당첨 번호 ====
(11, 17, 28, 30, 33, 35)

----- 내 번호 확인 -----
33 O 45 X 17 O 5 X 35 O 7 X
3 개 맞음

==== 모의 로또 당첨 번호 ====
(11, 17, 28, 30, 33, 35)

----- 내 번호 확인 -----
24 X 4 X 17 O 22 X 35 O 38 X
2 개 맞음



좀 더 알아봅시다!

+ random.randint()를 randint()로 사용할 수 없나요?

- 코딩은 가능하면 **간결**한 것이 좋다.
from 모듈을 사용하면 코드가 간결해질 수 있으므로 먼저 from import를 간략히 살펴 보자.
- import는 모듈을 가져오는 구문이며, import 이후 모듈에 속한 함수는 random.randint(1,5)와 같이 모듈, 함수명()으로 사용한다.

```
import random # import 모듈  
random.randint(1, 5) # 모듈.함수()
```



좀 더 알아봅시다!

+ random.randint()를 randint()로 사용할 수 없나요?

- from 모듈 import 함수 구문은 모듈의 특정 함수를 모듈명 없이 바로 사용할 수 있다.
- from random import randint와 같이 from 뒤에 모듈 이름을 지정하고 import 뒤에 가져올 함수를 기술한다.
- from import 이후에 가져온 함수를 사용할 때는 randint()와 같이 모듈의 이름을 붙이지 않고 바로 사용할 수 있다.

```
from random import randint # from 모듈 import 변수  
randint(1, 5) # 모듈 없이 바로 함수() 사용 가능
```

Chapter 3.

break와 continue문

P Y T H O N P R O G R A M M I N G

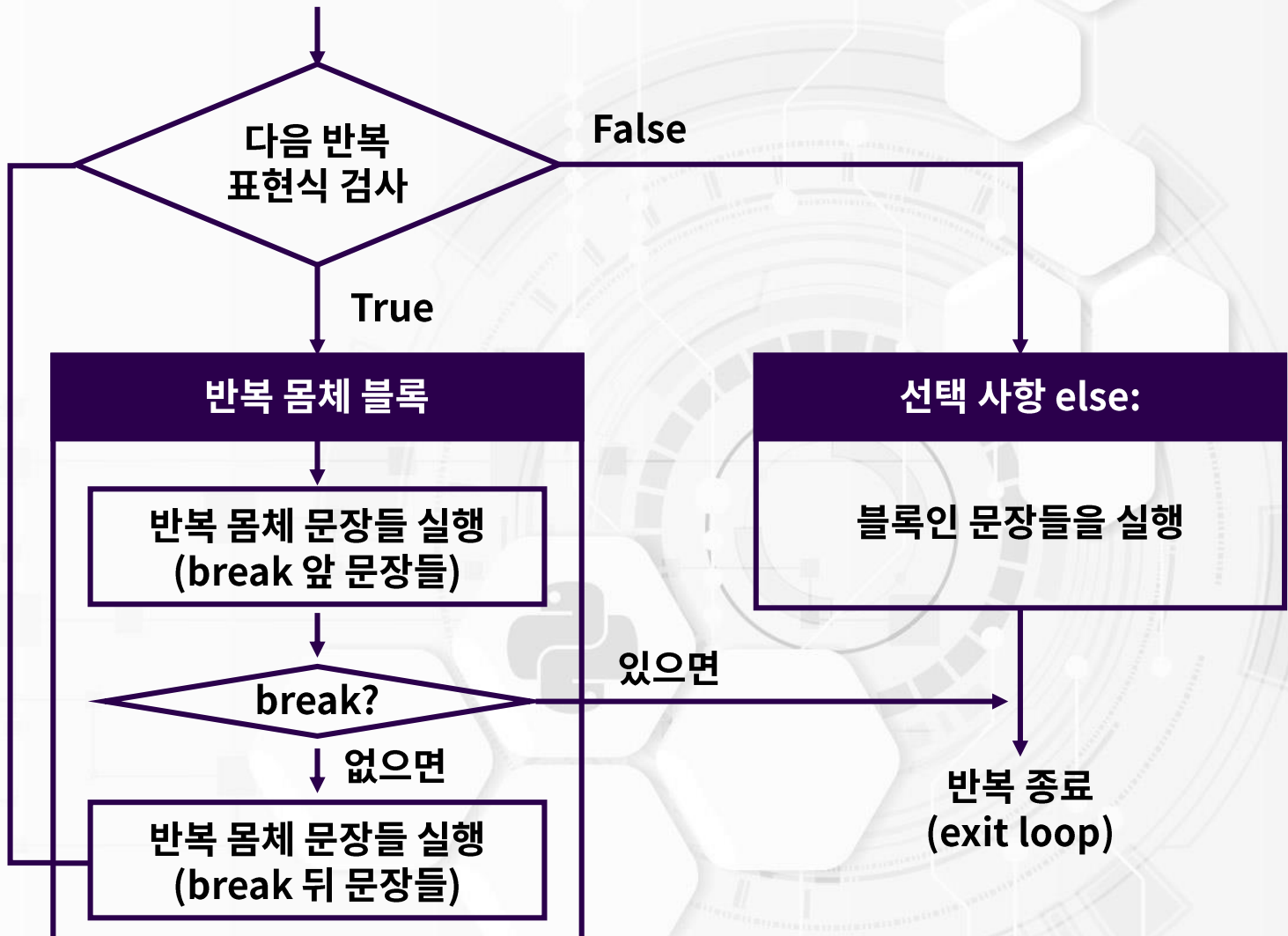
⚠ 반복을 종료하는 break문

무한반복

```
while True:
    반복 문장들
```

반복을 종료

```
while True:
    ...
    break
    ...
```



[그림14-1] 반복의 종료 break문

⚠ 0에서 9 사이의 난수 중에서 7이 나오면 반복 종료

[코딩실습] 0에서 9까지의 수 중에서 7이 나오면 반복 종료

난이도 응용

```
1. from random import randint
2. LUCKY = 7
3.
4. while True:
5.     n = randint(0, 9)
6.     if n == LUCKY:
7.         print('드디어 %d, 종료!' % n)
8.         break
9.     else:
10.        print('%d, %d 나올 때까지 계속!' % (n, LUCKY))
11.else:
12.    print('여기는 실행되지 않습니다.')
```

⚠ 0에서 9 사이의 난수 중에서 7이 나오면 반복 종료

주의	1번 줄은 문장으로, 5번 줄은 randint(0, 9)를 바로 사용할 수 있다.	
결과	2, 7 나올 때까지 계속! 8, 7 나올 때까지 계속! 3, 7 나올 때까지 계속! 5, 7 나올 때까지 계속! 드디어 7, 종료!	6, 7 나올 때까지 계속! 9, 7 나올 때까지 계속! 3, 7 나올 때까지 계속! 드디어 7, 종료!

⚠ continue 이후의 반복 몸체를 실행하지 않고 다음 반복을 실행

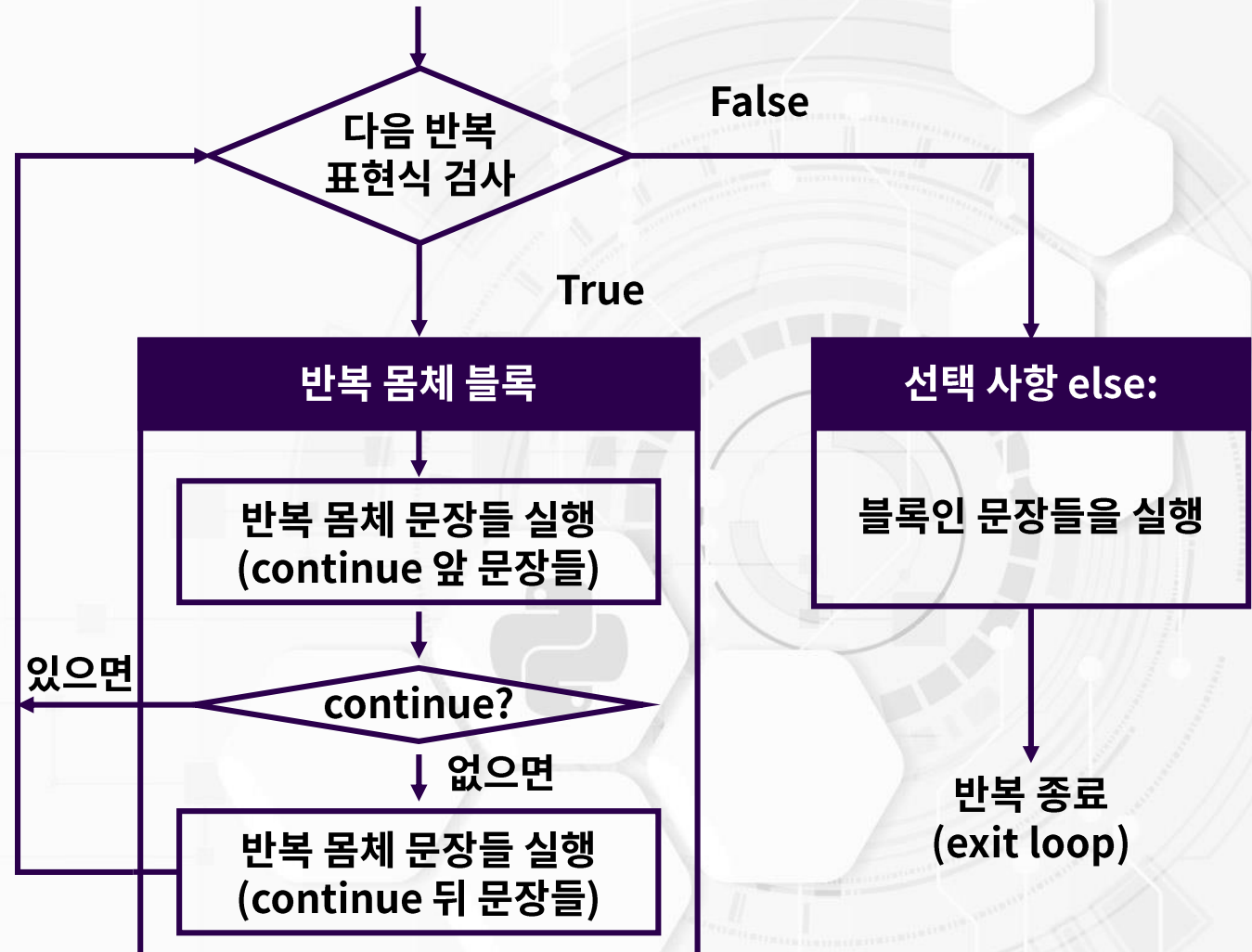
for에서 continue

```
for 변수 in 시퀀스:
    ...
    continue
    ...
```

while에서 continue

```
while 논리 표현식:
    ...
    continue
    ...
```

[그림14-2] 반복 몸체에서 continue문
이후의 문장을 제외하고 계속 반복



⚠ 일상 코딩: continue와 break로 여러 영어 단어 중에서 적어도 하나의 철자 맞추기

[코딩실습] 월, 화, 수 중 영어 철자 하나 검사

난이도 응용

```
1. days = ['monday', 'tuesday', 'wednesday']
2.
3. while True:
4.     user = input('월, 화, 수 중 하나 영어 단어 입력 >> ')
5.     if user not in days:
6.         print('잘못 입력했어요!')
7.         continue
8.     print('입력: %s, 철자가 맞습니다.' % user)
9.     break
10.
11. print(' 종료 '.center(15, '*'))
```

⚠ 일상 코딩: continue와 break로 여러 영어 단어 중에서 적어도 하나의 철자 맞추기

주의 7, 9번 줄은 들여쓰기를 잘 입력한다.

결과

월, 화, 수 중 하나 영어 단어 입력 >> monday
입력: monday, 철자가 맞습니다.
***** 종료 *****

월, 화, 수 중 하나 영어 단어 입력 >> tusday
잘못 입력했어요!
월, 화, 수 중 하나 영어 단어 입력 >> tuesday
입력: tuesday, 철자가 맞습니다.
***** 종료 *****

Upgrade coding

모든 요일에 대해 하나라도 맞는 문제로 향상시켜 보자.

힌트 days의 정답을 7개로 수정하고, 표준 입력 질의도 수정한다.

⚠ 내장 함수 range()를 사용한 for문

+ range(5)

- 정수 0에서 4까지 5개의 항목인 정수로 구성되는 시퀀스
 - 0 1 2 3 4

+ range(1, 10, 2)

```
for i in range(1, 10, 2)
...     print(i, end = ' ')
...
1 3 5 7 9
```

- start: 시퀀스의 시작
- stop: 시퀀스의 끝(미만), 실질적으로 stop-1까지
- step: 증가 값



좀 더 알아봅시다!

+ 범위라는 단어를 쓰는 함수 range()

- 내장 함수 range()는 일정 간격의 정숫값의 나열인 시퀀스를 생성해주는 함수다.

함수의 인자는 총 3개로, 적어도 1개의 인자를 지정해 줘야 한다. 만약 인자 중 하나만 지정하면 자동으로 stop 값이 지정되는데, stop의 이전 정숫값까지 포함된다. 시퀀스의 시작인 start와 인접한 숫자의 간격인 step은 자동으로 각각 0, 1로 지정된다. 인자를 2개 지정하면 자동으로 start와 stop 값을 설정하게 되며, 증가값은 자동으로 1로 지정된다.

- 모든 인자는 문자열 또는 부동소수점 숫자나 다른 유형이 올 수 없으며, 반드시 정수여야 한다

Python range(6)





좀 더 알아봅시다!

+ 범위라는 단어를 쓰는 함수 range()

- 다음 코드에서 볼 수 있듯이 list() 함수 내부에 range() 함수를 사용하면 바로 시퀀스 결과를 알 수 있다.

```
>>> list(range(4))  
[0, 1, 2, 3]  
>>> list(range(6, 10))  
[6, 7, 8, 9]  
>>> list(range(10, 20, 2))  
[10, 12, 14, 16, 18]
```

함수 호출	인자	결과
range(4)	stop	0, 1, 2, 3
range(6, 10)	start, stop	6, 7, 8, 9
range(10, 20, 2)	start, stop, step	10, 12, 14, 16, 18

SUMMARY



⚠ 난수를 위한 모듈 random

... 함수 randint(a, b)

⚠ 모듈 사용 방법 구문

... import random

... from random import randint

⚠ break와 continue문

- ... 반복을 종료하는 break문
- ... continue 이후의 반복 몸체를 실행하지 않고 다음 반복을 실행