13. 반복문 for

최희윤 강사

반복문 개요

✓ print('Hello, World!')를 1000번 출력하기

- 복사, 붙여넣기를 활용하면 간단하게 해결 가능
- 하지만, 코드의 내용도 너무 많아지고, 유지보수도 어려움
- 만약 복사 및 붙여넣기로 1000줄의 코드를 작성했는데, 'Hello, World!'가 아니라 'Hi, Wrold!'로 수정해야 한다면?
- 이를 해결하기 위해 for문 및 while문과 같은 반복문을 사용함

range() 함수

✓ range()

- 연속되는 숫자 요소들을 만들 때 활용
- 슬라이싱과 구조가 비슷하지만, 함수 형태로 쓰기 때문에 콤마를 통해 요소가 구분됨
- 시작 인덱스는 포함(이상), 끝 인덱스는 불포함(미만)
- 시작 인덱스(default=0)와 증감크기(default=1)는 생략 가능
- 숫자들의 시퀀스로 이터레이트 할 때 사용하면 편리

변수 = range(시작, 끝, 증감크기)

시퀀스자료형변수[시작 : 끝 : 증감크기]

헷갈리지 않기!

range() 함수

✓ range() 함수로 list 생성하기

- range()로 반환 받은 결과 자료형은 range임
- 이는 list() 내장함수를 활용하여 리스트로 변경 가능

✓ 100 미만의 2의 배수를 담은 리스트를 생성하는 방법

```
# 집합 자료형 접근
even_num = list(range(2, 100, 2))
print(even_num)

>> [2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, … , 96, 98]
```

range() 함수 특징

✓ range() 함수로 특징

- range()가 돌려준 객체는 마치 리스트인 것처럼 동작하지만, 사실 리스트와는 다름
- range 객체는 iterate할 때 원하는 시퀀스 항목들을 순서대로 요소를 돌려주는 객체지만,
- 메모리에 공간을 할당하여 미리 모든 요소들을 만들어 두는 것이 아니라
- 필요할 때 그 때 그 때 다음 요소를 만들어 반환하기에 메모리 공간 활용에 효율적

range()함수 실습 01

✓ [문제] 아래의 실행 결과처럼 나오도록 list()와 range()를 활용해서 구현해보기

➢ 실행 결과:

```
[5, 6, 7, 8, 9]
```

➢ 실행 결과:

```
[0, 3, 6, 9]
```

➢ 실행 결과:

```
[-10, -40, -70]
```

중간 용어 정리

✓ iterable

- 사전적 의미: '반복 가능한'
- 파이썬: 여러 개의 요소들로 구성 돼있으면서, 요소를 한 번에 하나씩 돌려줄 수 있는 객체
- 예) list, str, tuple, range와 같은 시퀀스 자료형이나비시퀀스 자료형인 disctionary

✓ 객체

- 메모리에 할당되어 존재하고 있는 데이터
- 클래스를 배울 때 다시 배울 예정! ☺

for문

✓ for문: 파이썬의 for문은 시퀀스 자료형의 항목들을 순서대로 요소를 하나씩 꺼냄

- 반복되는 부분을 만들 때 활용하면 좋음
- iterable객체를 루핑할 때 좋음

  (looping루핑: 반복문을 이용하여 iterable객체의 요소를 하나씩 처리하는 것을 의미)
- 반복 횟수가 정해져 있을 때 주로 사용

for 요소를 담을 변수 in 반복 가능한 객체:

반복할 코드(띄어쓰기 4칸)

for문과 range() 내장함수

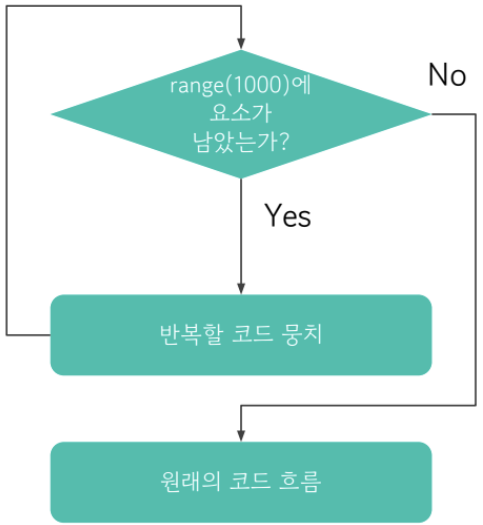
✓ for문과 range() 내장함수를 활용하여 'Hello, World!' 1000개 생성하는 원리 이해해보기

① 정수 0 ~ 999까지 1000개 생성
-> 0, 1, 2, 3, ..., 998, 999

② 생성된 요소들을
순서대로 하나씩 i 변수에 담음

```
for i in range(1000):
    print('Hello, World!')
```

③
- 요소를 하나 담을 때 마다
  들여쓰기로 구획된 코드 뭉치를 실행
- 이 때 i는 코드 뭉치에서 활용 가능



for문과 다양한 자료형 1

✓ list()

```
a = [1, 2, 3, 4, 5, 6, 7]

for i in a:
    print(i)
>> 1
   2
   3
   …
   7
```

for문과 다양한 자료형 2

✓ 문자열

```
for i in 'Orange':
    print(i, end='')
>> Orange
```

for문과 다양한 자료형 3

✓ dictionary()

```
a = {'name': 'Tom', 'math': 80, 'english': 70}

for i in a:
    print(i)

# key가 출력 됨
>> name
   math
   english
```

```
for i in a:
    print(i, end=' : ')
    print(a[i])
>> name : Tom
   math : 80
   english : 70
```

for문 응용

✓ 입력한 횟수만큼 반복하기

```
count = int(input('반복할 횟수: ')) # 3 넣어보기

for i in range(1, count+1):
    print(f'{i}번째 hello, world')

>> 1번째 hello, world
   2번째 hello, world
   3번째 hello, world
```

for문 응용

✓ enumerate 사용하여 index 접근

- iterable(반복 가능한) 객체를 순서와 함께 반복할 수 있도록 도와주는 내장 함수

```
a = [1, 2, 3]

for idx, val in enumerate(a):
    print(idx, val, sep=‘, ’)

>> 0, 1
   1, 2
   2, 3
```

for문 중첩

✓ for문 중첩

- for문 중첩은 반복문 안에 또 다른 반복문을 작성하여 이중으로 반복을 수행

```
for 변수1 in 반복 가능한 객체1:
    for 변수2 in 반복 가능한 객체2:
        실행문
```

for문 중첩

✓ for문 중첩

- for문 중첩은 반복문 안에 또 다른 반복문을 작성하여 이중으로 반복을 수행

```
for i in range(3): # 외부 루프: 0, 1, 2
    for j in range(2): # 내부 루프: 0, 1
        print(f"i={i}, j={j}")
>> i=0, j=0
   i=0, j=1
   i=1, j=0
   i=1, j=1
   i=2, j=0
   i=2, j=0
```

for문 실습 01

✓ [문제] Hello World를 100번 출력해주세요.

- 1부터 100까지의 숫자를 함께 출력해주세요.
- 아래의 실행 결과를 참고하세요.

➢ 실행 결과:

```
Hello World 1
Hello World 2
Hello World 3
…
Hello World 100
```

for문 실습 02

✓ [문제] 사용자로부터 자연수를 입력 받고 입력 받은 횟수만큼 'Hello World'를 출력해보세요.

➢ 실행 결과:

```
인사 몇 번 해드릴까요? 3
Hello World 1
Hello World 2
Hello World 3
```

for문 실습 03

✓ [문제] 아래의 실행 결과처럼 나오도록 for문과 range()를 활용해서 구현해봅시다.

- 숫자와 숫자 사이에 \t를 사용 (print문에 end='₩t' 옵션을 추가해주세요)

➢ 실행 결과:

```
자연수 하나 입력해주세요: 6
6 6 6 6 6 6
5 5 5 5 5
4 4 4 4
3 3 3
2 2
1
```

for문 실습 04

✓ [문제] 아래의 실행 결과처럼 나오도록 for문과 if문, range()를 활용해서 구현해봅시다.

- 첫 줄의 *의 개수는 10개 입니다.

➢ 실행 결과:

```
* * * * * * * * * *
*                 *
*                 *
*                 *
*                 *
*                 *
*                 *
*                 *
*                 *
* * * * * * * * * *
```

for문 실습 05

✓ [문제] 1부터 사용자로부터 입렵 받은 자연수까지의 홀수의 합을 구합니다.

- 1부터 입력받은 수까지 홀수 전체를 각각 더하는 산식을 표현하고
- 그 계산식의 결과를 아래 실행 결과와 같이 출력되도록 구해보세요

➢ 실행 결과:

```
1부터 입력한 자연수까지의 모든 홀수의 합을 구합니다: 100
1 + 3 + 5 + 7 + 9 + … + 99 = 2500
```

for문 실습 06

✓ [문제] 계승(Factorial)을 계산해보려 합니다.

- 사용자로부터 정수 하나를 입력 받습니다.
- 입력 받은 숫자까지의 계승을 for문을 활용하여 구한 뒤 출력해주세요.
- 계승이란 1부터 n까지의 자연수를 모두 곱하는 것을 의미합니다.
- n! = 1 x 2 x 3 x … x (n-1) x n

➢ 실행 결과:

```
계승을 구할 숫자를 입력하세요: 4
24
```

➢ 실행 결과:

```
계승을 구할 숫자를 입력하세요: 7
5040
```

14. 반복문 while

최희윤 강사

while문

✓ while문: 조건식으로만 동작

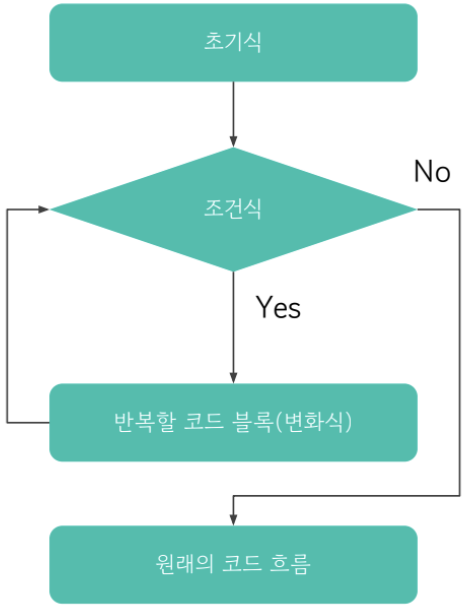
초기식

조건식

```
i = 0
while i < 100:
    print('Hello, World')
    i += 1
```

반복할 코드

변화식



while문

✓ while문

- 조건식으로만 동작

✓ while문 실행 순서

1. 조건식을 판별
2. 조건식이 True인 동안 내내 while문의 바디를 실행
3. 조건식이 False가 되면 while문을 빠져나옴

while문

```
i = 0                           # 초기식
while i < 100:                  # while 조건식
    print('Hello, Wolrd')       # 반복할 코드
    i += 1                      # 변화식
```

while문 실습 01

✓ [문제] while문을 사용하여 1이상 100이하의 짝수를 '\t'를 구분자로 하여 출력해보세요.

➢ 실행 결과:

```
2  4  6  8  10  12  14  16  18  20  … 98  100
```

while문 실습 02

✓ [문제] 자연수를 입력받아 구구단을 출력하는 프로그램을 구현해보세요.

- 아래의 실행 결과 참고하세요.

➢ 실행 결과:

```
구구단을 알려드립니다. 숫자를 입력하세요: 9
9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
…
9 * 8 = 72
9 * 9 = 81
```

while문 실습 03

✓ [문제] 학생의 수와 학생들의 성적을 입력 받아 성적 평균을 구하는 프로그램을 구현해보세요.

➢ 실행 결과:

```
학생 수: 3
1번 학생의 점수: 77
2번 학생의 점수: 88
3번 학생의 점수: 99
평균: 88.0점
```

while문 실습 04

✓ [문제] 학급의 수학 평균 점수를 구하는 프로그램을 구현해보세요.

- A학급의 수학점수가 다음과 같을 때 while문 조건식에 len()함수를 사용하여 평균 점수를 구해보세요
- class_a = (70, 60, 55, 75, 95, 90, 80, 80, 85, 100)

➢ 실행 결과:

```
79.0
```

for문과 while문 비교

✓ for문과 while문의 시퀀스 객체의 인덱스 루프(loop) 테크닉 비교

```
scores = (97, 100, 77)

# while문
i = 0
while i < len(scores):
    print(scores[i])
    i += 1

# for문
for i in range(len(scores)):
    print(score[i])
```

break

✓ for문과 while문에서 break는 **해당 반복문을 완전히 빠져나감**

- 보통 반복문 안에서 특정 조건에 부합할 때 해당 반복문을 빠져나가기 위해 쓰임
- 그래서 break는 보통 if문과 함께 쓰임

```
# 10000회 반복
for i in range(10000):
      print(i, end=‘ ’)
      if i == 100:
           break
```

while문 무한루프의 활용

while문 실습 05

✓ [문제] 0부터 100까지 자연수의 총합을 구하는 프로그램을 구현해 보세요

- while문과 break를 활용해서 작성해보세요

➢ 실행 결과:

```
5050
```

continue

✓ for문에서의 continue

- continue는 제어흐름을 유지하고, continue가 실행된 반복 횟수의 코드 실행만 건너뜀

```
for i in range(100): # 0부터 99까지 증가하면서 100번 반복
      if i % 2 == 0: # i를 2로 나누었을 때 나머지가 0이면 짝수
           continue  # 아래 코드를 실행하지 않고 건너뜀
      print(i, end='\t') # 즉, 홀수만 출력 됨
```

continue

✓ while문에서의 continue

- while문에서도 continue동작은 동일

```
i = 0
while i < 100: # 0부터 99까지 증가하면서 100번 반복
      i += 1
      if i % 2 == 0: # i를 2로 나누었을 때 나머지가 0이면 짝수
           continue # 아래 코드를 실행하지 않고 건너뜀
      print(i, end='\t') # 즉, 홀수만 출력 됨
```

while문 실습 05

✓ [문제] 두 자연수 n, m을 입력받고 n부터 m까지의 합을 구하는 프로그램을 작성하세요.

- (for, while 모두 작성해보기)

➢ 실행 결과:

```
시작하는 정수 입력: 1
끝나는 정수 입력: 10
1부터 10까지의 합은 55입니다.
```