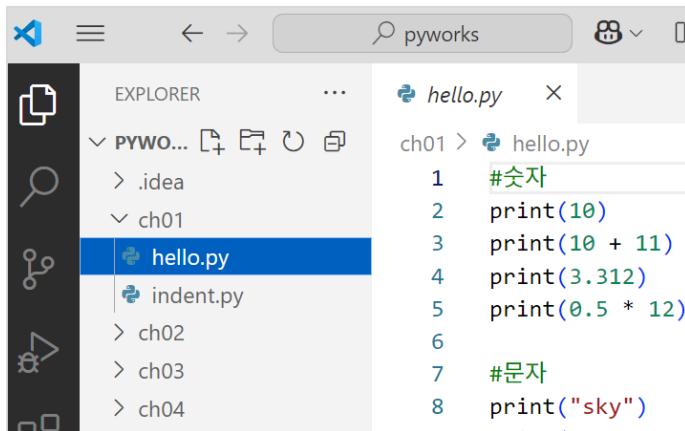


2장. 제어문(조건, 반복)



추천 IDLE(통합개발환경)

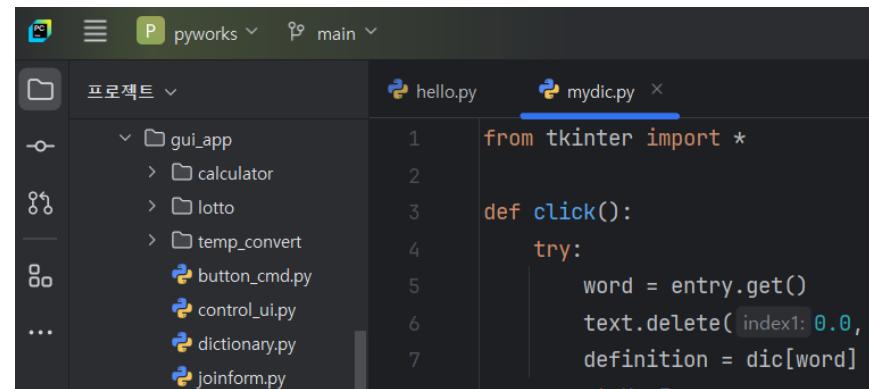
- IDLE – VS code, 파이참(Pycharm), 주피터 노트북 등



The screenshot shows the VS Code interface with a project named 'pyworks'. The Explorer sidebar shows a folder structure with '.idea', 'ch01', 'ch02', 'ch03', and 'ch04'. Inside 'ch01', there are files 'hello.py' (selected), 'indent.py', and 'main.py'. The 'hello.py' file contains the following code:

```
1 #숫자
2 print(10)
3 print(10 + 11)
4 print(3.312)
5 print(0.5 * 12)
6
7 #문자
8 print("sky")
```

비주얼 스튜디오 코드(VS code)



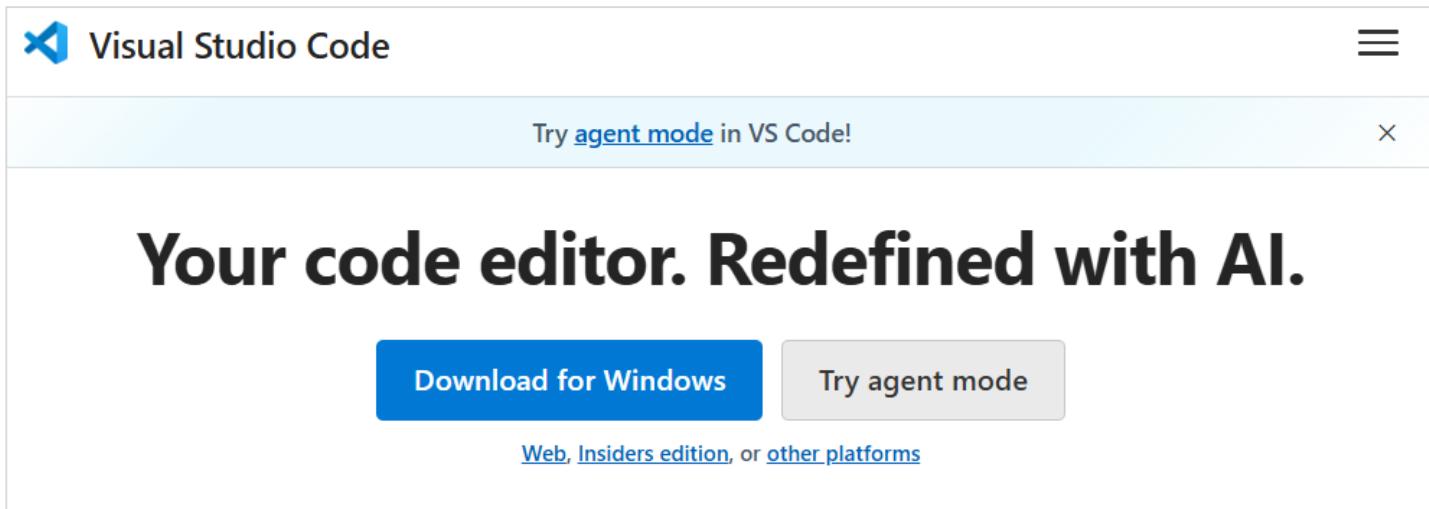
The screenshot shows the PyCharm interface with a project named 'pyworks'. The Project tool window shows a folder structure with 'gui_app', 'calculator', 'lotto', 'temp_convert', and several Python files like 'button_cmd.py', 'control_ui.py', 'dictionary.py', and 'joinform.py'. The 'mydic.py' file is open in the editor, showing the following code:

```
from tkinter import *
try:
    word = entry.get()
    text.delete( index1: 0.0,
    definition = dic[word]
```

파이참(pycharm)

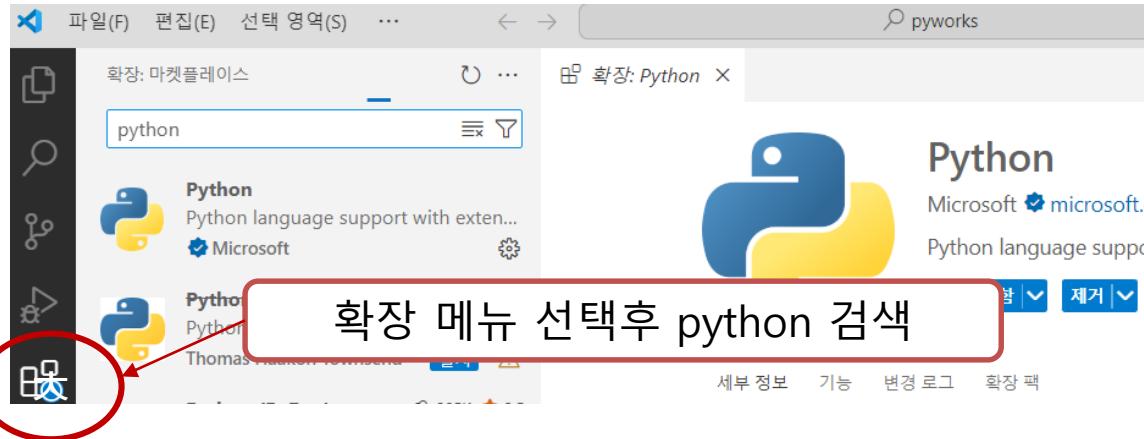
비주얼 스튜디오 코드 설치

◆ 비주얼 스튜디오 코드(VS code) 다운로드

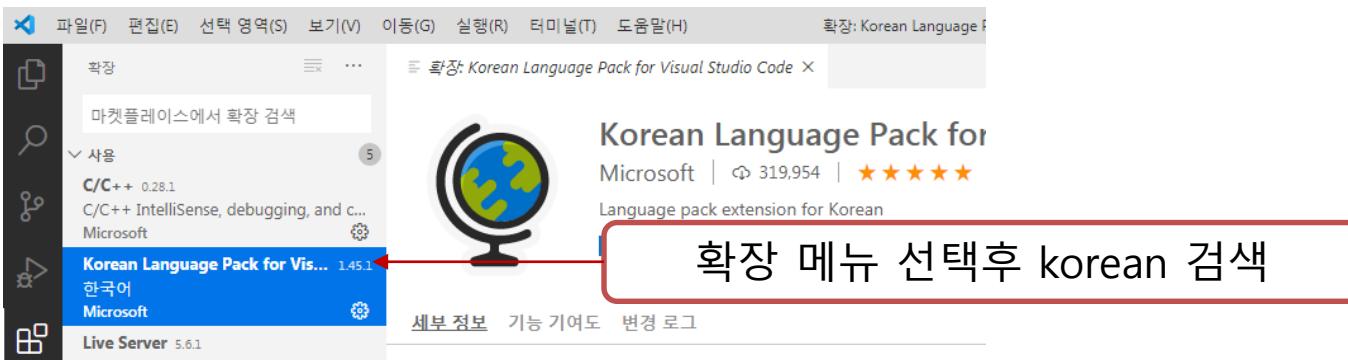


비주얼 스튜디오 코드 환경 설정

◆ Python 언어 지원 확장 팩 설치하기

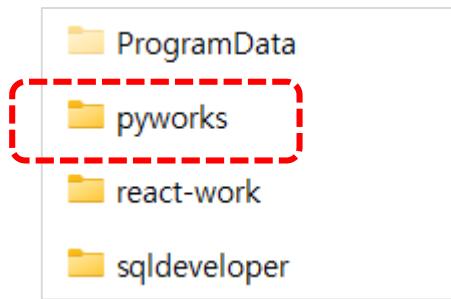


◆ 한국어 팩 설치

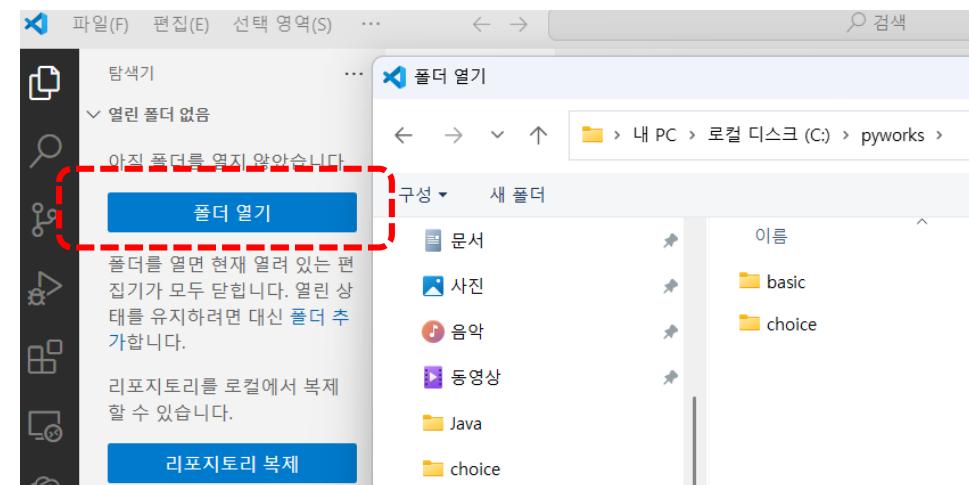


VS code – 작업 폴더 설정하기

◆ VS code – 작업 폴더 설정



① 작업영역 폴더 만들기



② 작업영역 폴더 설정

VS code – 설정 환경

◆ VS code – 관리 도구

The screenshot shows the VS Code interface. On the left, the sidebar has a gear icon highlighted with a red circle, and a red arrow points from it to a red box labeled "관리 > 설정". The main area shows the "General" tab of the settings editor. A red box highlights the "Files: Auto Save" section, specifically the "afterDelay" input field, which is also circled in red. Another red box highlights the "Editor: Font Size" input field, which contains the value "17" and is also circled in red. Red boxes also highlight the titles "일반적으로 사용되는 설정" and "자동 저장".

사용자 작업 영역

일반적으로 사용되는 설정

Files: Auto Save

작성되지 않은 변경 사항이 있는 편집기의 자동 저장을 제어합니다.

afterDelay

자동 저장

Editor: Font Size

글꼴 크기(픽셀)를 제어합니다.

17

글꼴 크기

Editor: Font Family

글꼴 패밀리를 제어합니다.

Consolas, 'Courier New', monospace

관리 > 설정

파이썬 파일 실행

◆ 파일 실행 및 터미널 출력

The screenshot shows the VS Code interface with a Python file named `hello.py` open. The code contains several `print()` statements. A context menu is open at the top right, with the "Python 파일 실행" option highlighted. A red circle and arrow point to this option, which is labeled "실행" (Run) in a red box. The terminal below shows the execution results.

```
# print() 흔히 쓰는 출력 방법
# 문자 출력
print("Hello")
print("안녕~")
print('010-1234-5678')

# 숫자 출력
print(12)
print(2.54)
print(10 + 20)
print(10 - 20)
```

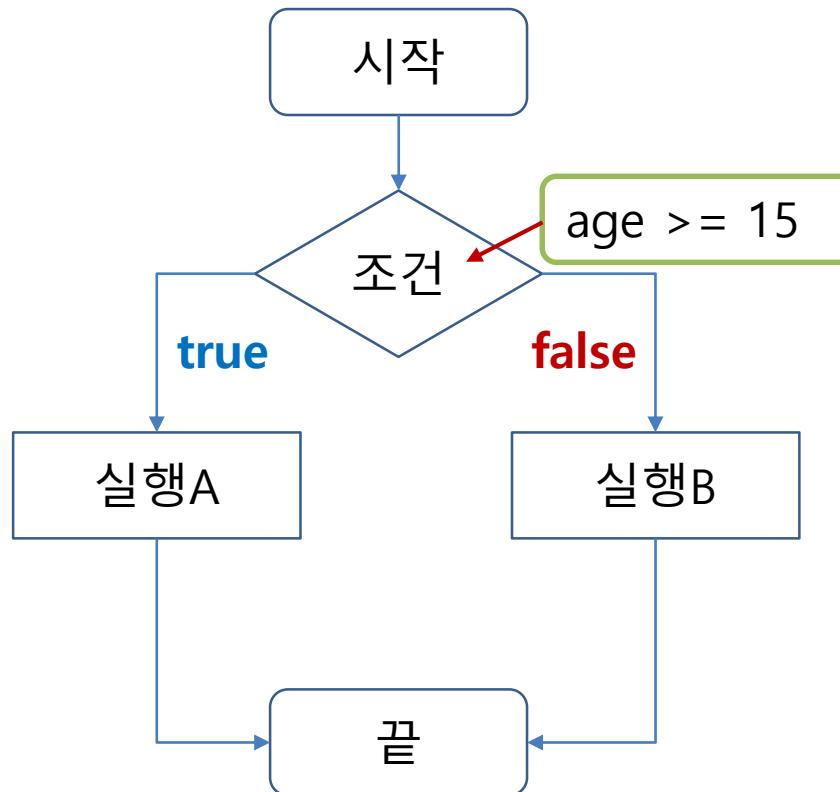
Python 파일 실행
전용 터미널에서 Python 파일 실행
대화형 창에서 현재 파일 실행
Python 디버거: Python 파일 디버그
Python 디버거: launch.json을 사용하여 디버그

문제	출력	디버그 콘솔	터미널	포트
PS C:\pyworks> & C:/Users/LG/AppData/Local/Programs/Python/Python38/python.exe hello.py	Hello~ World! 안녕~ 세계야! 010-1234-5678			
	12 2.54 30 -10			
	PS C:\pyworks>			

조건문(Choice Structure)

조건문

- 특정한 조건에 의해서 프로그램 진행이 분기되는 구문



조건문 - if

▪ if 문

if 조건식:

수행문

4칸 들여쓰기
인덴트(indent)

조건식이 참이면 수행문 실행

▪ if ~ else 구문

if 조건식:

수행문 1

else :

수행문 2

조건식이 참이면 수행문1 실행,
아니면 수행문2 실행

조건문 - if

- if 문

```
age = 16
if age < 19:
    print("미성년자입니다.")

print(f"나이는 {age}세입니다.")
```

- if ~ else 구문

```
age = 19
if age < 19:
    print("미성년자입니다.")
else:
    print("성인입니다.")

print(f"나이는 {age}세입니다.")
```

조건문 - if

▪ 좌석의 줄 수를 계산하는 프로그램

입장객 수: 20
좌석 열 수: 5
4개의 줄이 필요합니다.

입장객 수: 22
좌석 열 수: 5
5개의 줄이 필요합니다.

```
customer = int(input("입장객 수: "))
column = int(input("좌석 열 수: "));
row = 0

if customer % column == 0:
    #row = customer // column
    row = int(customer / column)
else:
    row = int(customer / column) + 1

print(str(row) + "개의 줄이 필요합니다.")
```

조건문 - if

- if ~ 내부 if문

```
num = int(input("정수를 입력하세요: "))

if num > 10:
    if num < 20:
        print("10보다 크고 20보다 작은 수입니다.")
    else:
        print("20 이상의 수입니다.")
else:
    print("10 이하의 수입니다.")
```

조건문 - if

▪ 윤년을 판정하는 프로그램

- 4의 배수(4년에 한 번 온다)
 - 100의 배수는 아니다(100년 단위는 윤년이 아님)
 - 400의 배수이다.(400년 단위는 윤년임)
- > 4의 배수이고, 100의 배수는 아니나(또는), 400의 배수이다.

연도를 입력하세요: 2024
2024년은 윤년입니다.

연도를 입력하세요: 1900
1900년은 윤년이 아닙니다.

```
year = int(input("연도를 입력하세요: "))

if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
    print(f"{year}년은 윤년입니다.")
else:
    print(f"{year}년은 윤년이 아닙니다.")
```

다중조건 – if ~elif ~ else

- if ~ elif ~ else구문

if 조건1:

수행문1

elif 조건2:

수행문2

else :

실행문 3

조건1이 참이면 수행문1 실행, 조건2가 참이면 수행
문2 실행, 조건1,2가 모두 거짓이면 수행문3 실행

다중조건 – if ~elif ~ else

◆ 놀이 공원 입장료 계산 프로그램

대상	입장료
어린이(14세 미만)	1,000원
청소년(19세 미만)	2,000원
일반인(65세 미만)	3,000원
경로우대(65세 이상)	1,500원

다중조건 – if ~elif ~ else

◆ 놀이 공원 입장료 계산하기

```
print("== 놀이 공원 입장료 계산 ==")
age = int(input("나이를 입력하세요: "))
fee = 0 # 입장료 초기화

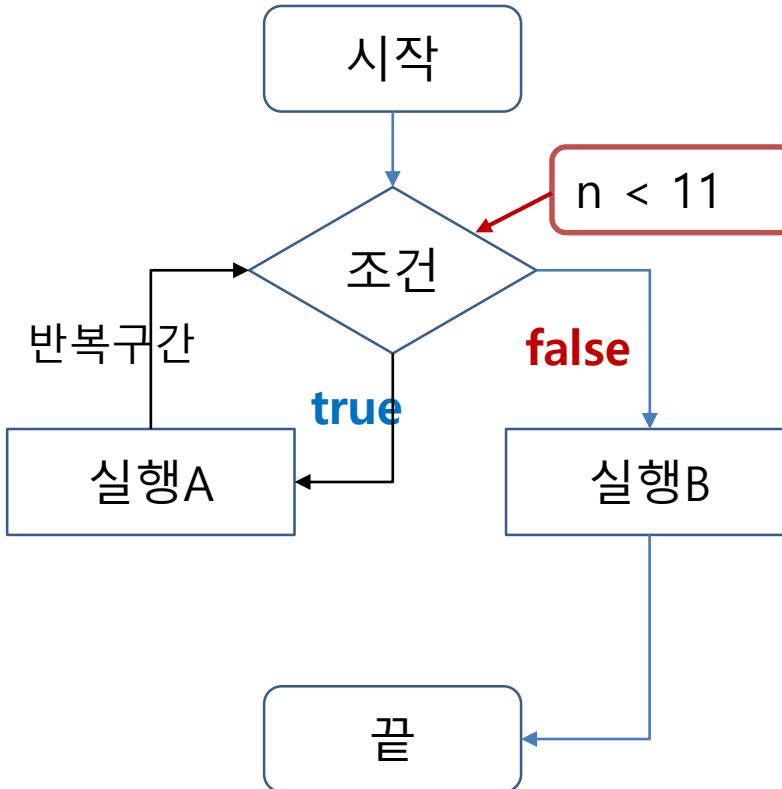
if age < 14:
    fee = 1000
    print("어린이 요금이 적용됩니다.")
elif age < 19 :
    fee = 2000
    print("청소년 요금이 적용됩니다.")
elif age < 65 :
    fee = 3000
    print("성인 요금이 적용됩니다.")
else:
    fee = 1500
    print("경로 우대 요금이 적용됩니다.")

print(f"입장료는 {fee}원입니다.")
```

반복문

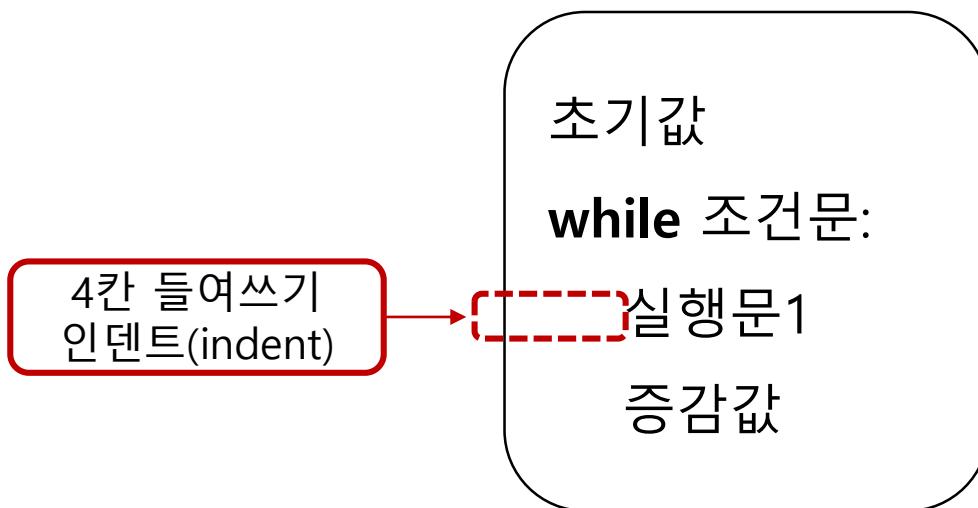
● 반복문

- 주어진 조건이 만족할 때까지 실행문을 반복적으로 수행
- while문과 for문이 대표적이다.



반복문

● while문



반복문 - while

● while문 예제

```
a = 1
print(a)

a += 1
print(a)

a += 1
print(a)
```

```
# 1부터 5까지 출력하는 while 반복문
print("1부터 5까지 출력")
n = 1
while n <= 5:
    print(n)
    n += 1

print("5부터 1까지 출력")
n = 5
while n >= 1:
    print(n)
    n -= 1
```

반복문 - while

● while문 예제

```
# 1부터 10까지의 합을 구하는 프로그램  
i = 1  
total = 0  
while i <= 10:  
    total += i  
    print("i=", i, ", total=", total)  
    i += 1  
  
print(f"1부터 10까지의 합은 {total}입니다.")
```

```
i= 1 , total= 1  
i= 2 , total= 3  
i= 3 , total= 6  
i= 4 , total= 10  
i= 5 , total= 15  
i= 6 , total= 21  
i= 7 , total= 28  
i= 8 , total= 36  
i= 9 , total= 45  
i= 10 , total= 55  
1부터 10까지의 합은 55입니다.
```

반복문 - while

● 반복 조건문

반복문에서 break 문을 만나면 더 이상 반복을 수행하지 않고,
반복문을 빠져 나옴

```
while True:
```

```
    수행문
```

```
    if 조건 :
```

```
        break
```

반복문 - while

● 반복 조건문

```
# 1부터 5까지 출력하는 프로그램 (break 사용)
n = 1
while True:
    print(n)
    n += 1
    if n > 5:
        break
print("반복문 종료")
```

반복문 - while

● 반복 조건문

```
# 1부터 10까지의 합을 구하는 프로그램 (break 사용)
i = 1
total = 0
while True:
    total += i
    i += 1
    if i > 10:
        break
print(f"1부터 10까지의 합은 {total}입니다.")
```

반복문 - while

● 반복 조건문

```
# 사용자에게 'y' 또는 'n'이 입력될 때까지 반복하는 프로그램
while True:
    answer = input("계속하려면 'y', 종료하려면 'n'을 입력하세요: ")
    if answer == 'y' or answer == 'Y':
        print("계속 진행합니다.")
    elif answer == 'n' or answer == 'N':
        print("프로그램을 종료합니다.")
        break
    else:
        print("올바른 입력이 아닙니다. 다시 시도하세요.")
```

반복문 - while

● in 명령어

```
# 문자열 내에 특정 문자나 문자열이 포함되어 있는지 확인
# in, not in
animals = "dog cat bird fish"
print("dog" in animals) # True
print("lion" in animals) # False
print("cat" not in animals) # False
print("tiger" not in animals) # True
```

반복문 - while

● 챗봇 프로그램

```
from datetime import datetime

while True:
    user_input = input("챗봇에게 질문하세요 (종료하려면 'exit' 입력): ")
    if user_input == "exit":
        print("챗봇을 종료합니다.")
        break
    elif "안녕" in user_input:
        print("챗봇: 안녕하세요! 무엇을 도와드릴까요?")
    elif "이름" in user_input:
        print("챗봇: 제 이름은 챗봇입니다.")
    elif "시간" in user_input:
        now = datetime.now()
        print(f"챗봇: 현재 시간은 {now.hour}시 {now.minute}분입니다.")
    else:
        print("챗봇: 죄송합니다, 잘 모르겠어요.")
```

반복문 - for

● for문

순서열의 각 원소를 처음부터 순회하면서 반복변수에 담아 낸다.

순서열은 **리스트**, **튜플**, **문자열** 등을 사용

for - in range() 와 for - in 문을 사용함.

for 반복변수 in range(시작값, 종료값, 증감값):

실행문

for 반복변수 in 순서열:

실행문

반복문 - for

- range() 함수 사용하기

range(시작값, 종료값, 증감값):

- ✓ 시작값을 생략하면 0부터 시작하고, 종료값은 (종료값-1) 이다.

```
print(range(5)) # range(0, 5)
print(list(range(0, 5))) # [0, 1, 2, 3, 4]
print(list(range(1, 5, 1))) # [1, 2, 3, 4]
```

```
# 1부터 10까지 출력
for n in range(1, 11):
    print(n)
```

반복문 - for

- range() 함수 사용하기

```
# 1부터 10까지의 합 구하기
total = 0
for n in range(1, 11):
    total += n
print(f"1부터 10까지의 합: {total}")

# 1부터 10까지의 짝수 출력
for n in range(2, 11, 2):
    print(n, end=" ")
print()

# 1부터 10까지의 홀수 출력
for n in range(1, 11):
    if n % 2 == 1:
        print(n, end=" ")
```

구구단

- 단을 입력받아 구구단 출력하기

```
단을 입력하세요: 7
```

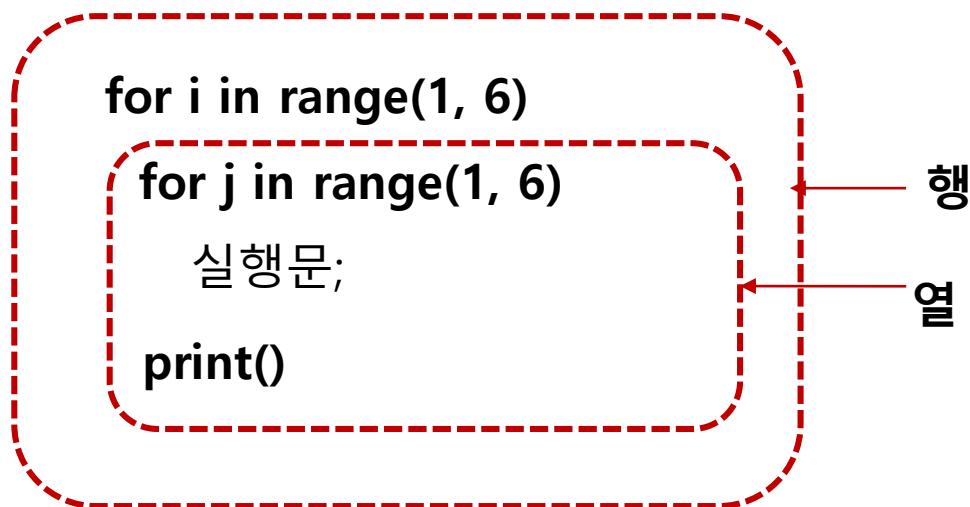
```
7 x 1 = 7  
7 x 2 = 14  
7 x 3 = 21  
7 x 4 = 28  
7 x 5 = 35  
7 x 6 = 42  
7 x 7 = 49  
7 x 8 = 56  
7 x 9 = 63
```

```
dan = int(input("단을 입력하세요: "))  
  
for n in range(1, 10):  
    result = dan * n  
    print(f"{dan} x {n} = {result}")
```

반복문

● 이중 for문

- 행, 열 구현하기



5행 5열

	열1	열2	열3	열4	열5
행1					
행2					
행3					
행4					
행5					

반복문

● 중첩 for문

```
가가가가  
가가가가  
가가가가
```

```
for i in range(3): # 바깥쪽 for문
    for j in range(4): # 안쪽 for문
        print("가", end="")
    print() # 줄바꿈
```

이중 for문 – 구구단 전체

▣ 구구단 전체 출력 프로그램

```
# 구구단 전체 출력하기
for dan in range(2, 10): # 2단부터 9단까지
    for n in range(1, 10): # 각 단의 1부터 9까지
        result = dan * n
        print(f"{dan} x {n} = {result}")
print() # 단이 바뀔 때마다 줄바꿈
```

2 x 1 = 2	4 x 1 = 4	6 x 1 = 6
2 x 2 = 4	4 x 2 = 8	6 x 2 = 12
2 x 3 = 6	4 x 3 = 12	6 x 3 = 18
2 x 4 = 8	4 x 4 = 16	6 x 4 = 24
2 x 5 = 10	4 x 5 = 20	6 x 5 = 30
2 x 6 = 12	4 x 6 = 24	6 x 6 = 36
2 x 7 = 14	4 x 7 = 28	6 x 7 = 42
2 x 8 = 16	4 x 8 = 32	6 x 8 = 48
2 x 9 = 18	4 x 9 = 36	6 x 9 = 54

3 x 1 = 3	5 x 1 = 5	7 x 1 = 7
3 x 2 = 6	5 x 2 = 10	7 x 2 = 14
3 x 3 = 9	5 x 3 = 15	7 x 3 = 21
3 x 4 = 12	5 x 4 = 20	7 x 4 = 28
3 x 5 = 15	5 x 5 = 25	7 x 5 = 35
3 x 6 = 18	5 x 6 = 30	7 x 6 = 42
3 x 7 = 21	5 x 7 = 35	7 x 7 = 49
3 x 8 = 24	5 x 8 = 40	7 x 8 = 56
3 x 9 = 27	5 x 9 = 45	7 x 9 = 63

이중 for문 – 별 찍기

▣ 별(*) 모양 출력하기

```
# 별 출력하기
for i in range(5): # 5행
    for j in range(5): # 5열
        print("*", end="")
    print() # 줄바꿈

# 별 출력하기
for i in range(1, 6): # 5행
    for j in range(1, 6): # 5열
        print("*", end="")
    print() # 줄바꿈

# 별(*) 삼각형 출력하기
rows = 5
for i in range(1, rows + 1): # 1부터 rows까지
    for j in range(i): # i개의 별 출력
        print("*", end="")
    print() # 줄바꿈
```

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

이중 for문 – 좌석번호 출력

▣ 좌석 번호 출력

```
# 좌석 번호 출력하기
for i in range(0, 5):
    for j in range(1, 6):
        seat_number = 5 * i + j
        print(seat_number, end=" ")
    print()
print()

# 23번 좌석까지만 출력하기
for i in range(0, 5):
    for j in range(1, 6):
        seat_number = 5 * i + j
        if seat_number > 23:
            break
        print(seat_number, end=" ")
    print()
```

```
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25
```

```
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23
```

자리 배치도

■ 자리 배치도(Seat Allocation)

좌석 배치도

입장객 수 입력(25명 정원): 23

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23

```
print("좌석 배치도")
customers = int(input("입장객 수 입력(25명 정원): "))
columns = 5 # 한 줄에 5개의 좌석

# 필요한 행 수 계산
if customers % columns == 0:
    rows = customers // columns
else:
    rows = customers // columns + 1

# 좌석 번호 출력
for i in range(0, rows):
    for j in range(1, columns + 1):
        seat_number = columns * i + j
        if seat_number > customers:
            break
        print(seat_number, end=" ")
print()
```

실습 문제 1 - 조건문

학점 계산 프로그램

아래의 표를 참고해서 프로그램을 구현해 보세요.

점수	학점
90 ~ 100	A
80 ~ 89	B
70 ~ 79	C
60 ~ 69	D
60 미만	F

☞ 실행 결과

점수를 입력하세요: 86
학점은 B입니다.

실습 문제2 – while문

1부터 입력한 정수 까지의 합을 계산하는 프로그램

☞ 실행 결과

```
정수를 입력하세요: 5  
1부터 5까지의 합은 15입니다.
```