

## 4. Python 기초와 실습 1

# CONTENTS

01

## 기초와 실습 1

01-1 출력(Print), 주석  
(Comment), 연산

01-2 변수(Variable)

01-3 자료형(type)

02

## 기초와 실습 2

02-1 입력(input)

02-2 조건문(if, else,  
elif)



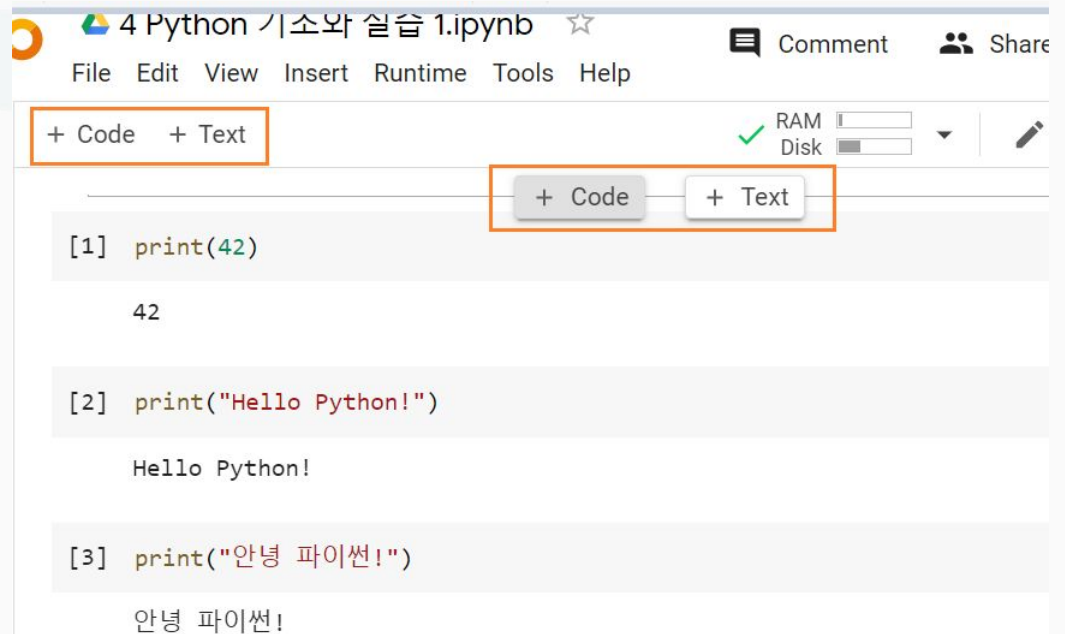
# 01 파이썬(Python) 기초와 실습 1

## 실습 준비

1. <https://colab.research.google.com/> 로 이동합니다.
2. 구글 계정으로 로그인합니다.
3. File(파일) >> New notebook을 선택합니다.
4. Python basic exercise 1.ipynb 로 파일이름을 변경합니다.
5. 시작하기전에 Save를 해봅니다. File >> Save
6. File >> Locate in Drive로 저장되 있는 경로를 확인합니다. (보통 My Drive > Colab Notebooks로 저장됩니다.)

## 실습 준비

- 중간지점에 마우스를 가지고 가면  
CODE와 TEXT의 옵션이 나타남
- 새로운 코드박스를 만들기
- Ctrl+Enter의 단축키로 실행 가능



## 출력하기(print)

- 숫자출력
- 문자출력

```
[1] print(42)
```

42

```
[2] print("Hello Python!")
```

Hello Python!

```
[3] print("안녕 파이썬!")
```

안녕 파이썬!

## 문제풀기

### 출력하기

- 다음 출력값은 무엇일까요?



```
print(10)  
print(20)
```

1. 1020
2. 10  
20
3. 10  
  
20



## 문제풀기

### 출력하기

- 다음 출력값은 무엇일까요?



```
print(10)  
print(20)
```

1. 1020

2. 10  
20

# print() 함수는 출력 후 새로운 줄을 추가

3. 10

20



## 출력하기

- 작은따옴표('')
- 큰따옴표("")
- print() 함수로 출력할 문장(문자열)은 '' 또는 "" 로 감쌈
- 쉼표/кома(,) - 문자열을 나열할 경우 공백(기본값)이 자동으로 추가
- 연산자 더하기(+) - 문자열을 공백없이 연결

\*쉼표(,) 사용할때 파라미터(매개변수)가 둘(혹이 둘 이상)이 됨



```
print('Hello Python!')  
  
print("Nice to meet you.")  
  
print('Hello "Python"')  
  
print("Hello 'Python'")  
  
print('Hello', 'Python!')  
  
print('Hello' + 'Python!')
```

```
Hello Python!  
Nice to meet you.  
Hello "Python"  
Hello 'Python'  
Hello Python!  
HelloPython!
```

## 문제풀기

### 출력하기

- 파이썬은 출력전에 연산을 할 수 있을까요? 다음 출력값은 무엇일까요?



```
print(10 + 7)
```

1. 107
2. 17
3. 10

## 문제풀기

### 출력하기

- 파이썬은 출력전에 연산을 할 수 있을까요? 다음 출력값은 무엇일까요?



```
print(10 - 1)  
print(8 - 5)  
print(3 + 4)
```

## 주석달기(Comment)

코드는 컴퓨터에게 전달되는 내용이지만, 코드의 일부는 작성자나 다른 사람을 위한 설명으로 채워집니다. 이부분을 주석(comment)이라고 하고 앞에 #을 붙입니다. 연습과정에서는 코드를 연습한 날짜라던가 주의할 부분을 간단하게 기록하는 것부터 시작하면 됩니다.

주석은 중요한 기능중에 하나입니다.

- 내가 작성한 코드를 남들이 보았을때 코드가 긴경우나 복잡한 경우 주석으로 설명
- 보는사람이 이해하기 쉽게
- 주석은 컴퓨터가 읽지 않음

또한, 코딩을 하다보면 오래된 소스인 경우 기억이 잘 나지 않는경우가 있기 때문에, 주석을 설정합니다.

소스에 주석을 다는 습관을 길들이면 정말 유용합니다. 하지만 너무 복잡하게 쓰면 오히려 더 혼란이 올 수도 있습니다.

주석은 **한줄주석**과 **여러줄 주석**으로 나뉩니다.

## 주석달기(Comment) - 한줄주석

한줄주석 에는 샵(#)을 사용

1. 공백에 공간에 사용가능
2. 코드 뒷부분에 사용가능



```
# a는 apple를 의미함  
print("a를 출력")
```

```
print("a를 출력") # a는 apple를 의미함
```

```
a를 출력  
a를 출력
```

## 주석달기(Comment) - 여러줄 주석

- 큰따옴표 `"""` 3개 연달아
- 작은따옴표 `'''` 3개를 연달아

시작과 끝부분에 사용



#큰 따옴표 `"""` 주석방법

```
print("퇴근하고 싶습니다 Start")
"""
오늘은 날씨가 참 좋아요.
시계를 보니 아직 퇴근시간은
아직멀었어요. 일하기가 싫네요.
"""
print("퇴근하고 싶습니다 End ")
```

#작은 따옴표 `'''` 주석방법

```
print("오늘은 연차 사용날 Start")
'''
오늘은 날씨가 참 좋아요.
시계를 보니 아직 퇴근시간은
아직멀었어요. 일하기가 싫네요.
'''
print("벌써 하루가 저물었네요 End ")
```

```
퇴근하고 싶습니다 Start
퇴근하고 싶습니다 End
오늘은 연차 사용날 Start
벌써 하루가 저물었네요 End
```

## 주석달기(Comment) - 주석 에러

‘들여쓰기’ -

주석은 소스와 동일하게 들여쓰기를  
해야합니다.



```
print("오늘은 연차 사용날 Start")
```

```
'''
```

```
    오늘은 날씨가 참 좋아요.  
    시계를 보니 아직 퇴근시간은  
    아직 멀었어요. 일하기가 싫네요.
```

```
'''
```

```
print("벌써 하루가 저물었네요 End ")
```

```
File "<ipython-input-3-f6c964ceb6ec>", line 2
```

```
'''
```

```
^
```

```
IndentationError: unexpected indent
```

SEARCH STACK OVERFLOW



## 문제풀기

### 주석달기(Comment)

한줄 주석을 달아봅니다.  $3 + 5$  에 대한 설명을 주석을 달아봅니다.



```
print(3 + 5)
```

## 문제풀기

### 연산

다음 코드를 실행하면 어떤 결과가 나올까요?



```
print(6 % 4)
```

## 연산

modulo(mod) %

나머지를 구하는 연산



```
print(6 % 4)
```

## 문제풀기

### 연산

다음 코드를 실행하면 어떤 결과가 나올까요?



```
print(5 % 2)
```

1. 2.5
2. 2
3. 1

## 연산

거듭제곱 - \*\* 을 사용



```
print(5 ** 2)
```

25

## 문제풀기

## 연산

물음표(?) 안에 들어갈 것은?

Complete the code to return the output

```
print( ? ? 2)
```

25

FILL IN THE BLANKS

4

\*\*\*

\*

\*\*

5

3



# 01-2 변수(Variable)



## 변수 (Variable)

지금까지 파이썬에서 숫자와 연산자를 사용하여 직접 계산을 해보았습니다. 그런데 계산 결과를 바로 출력할 수는 있었지만 결과를 계속 가지고 있을 수는 없었습니다.

이때는 변수를 사용해서 결과를 저장해야 합니다.

이번 유닛에서는 변수(variable)를 만들어 결과를 저장하는 방법을 알아보겠습니다.

## 변수 (Variable)

변수 만들기

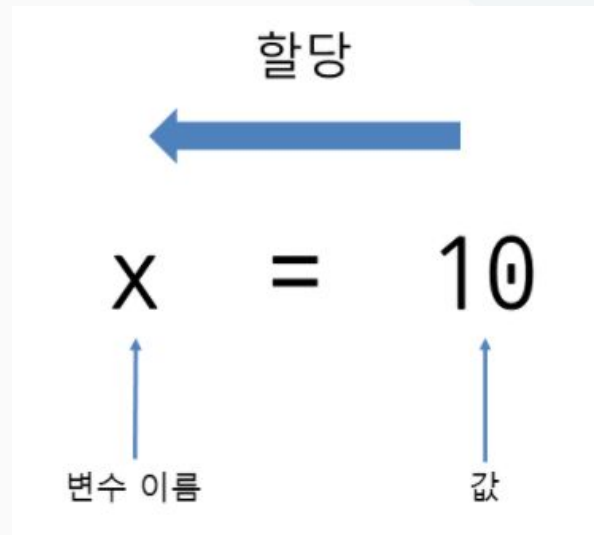
\*강조: (=) 기호는 같다는 뜻 아닌가요?

수학에서는 =(등호) 기호는 양 변이 같다는 뜻이죠? 하지만 프로그래밍 언어에서 =는 변수에 값을 할당 (assignment)한다는 의미입니다. 수학의 등호와 같은 역할을 하는 연산자는 ==입니다.

## 변수 (Variable)

### 변수 만들기

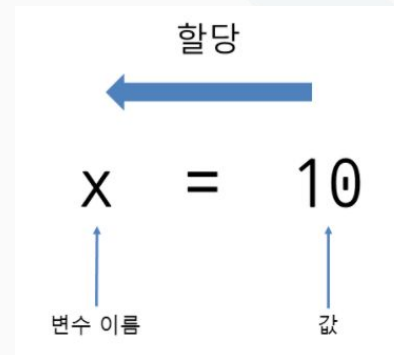
파이썬에서는 다음 그림과 같은 형식으로 코드를 입력하여 변수를 만듭니다.



## 변수 (Variable)

### 변수 만들기

`x = 10`이라고 입력하면 10이 들어있는 변수 `x`가 만들어집니다. 즉, 변수이름 = 값 형식이죠. 이렇게 하면 변수가 생성되는 동시에 값이 할당(저장)됩니다.



## 변수 (Variable)

### 변수 만들기

변수 이름은 원하는 대로 지으면 되지만 다음과 같은 규칙을 지켜야 합니다.

- 영문 문자와 숫자를 사용할 수 있습니다.
- 대소문자를 구분합니다.
- 문자부터 시작해야 하며 숫자부터 시작하면 안 됩니다.
- \_(밑줄 문자)로 시작할 수 있습니다.
- 특수 문자(+, -, \*, /, \$, @, &, % 등)는 사용할 수 없습니다.
- 파이썬의 키워드(if, for, while, and, or 등)는 사용할 수 없습니다.

## 변수 (Variable)

변수 만들기

\*변수의 경우 `print()` 함수 없이 출력 가능



```
x = 10
x
```

10

The image shows a code execution environment. The top part is a light gray box containing the code `x = 10` and `x` on separate lines. The bottom part is a white box containing the output `10`. A play button icon is visible on the left side of the top box.

## 변수 (Variable)

변수 만들기

\*문자열도 가능



```
y = 'Hello world!'  
y
```

```
'Hello world!'
```



## 변수 (Variable)

변수 여러 개를 한 번에 만들기

\*심표(.)를 사용해서 여러 개의 변수만들기가  
한 줄에 가능

```
[15] x, y, z = 10, 20, 30  
x
```

10

```
[16] y
```

20

```
▶ z
```

30

## 문제풀기

### 변수 (Variable)

변수 여러 개를 한 번에 만들기

할당 값이 동일하지 않을 경우는?



```
x, y, z = 10, 20
```

## 변수 (Variable)

변수 여러 개를 한 번에 만들기

- 변수 여러 개를 만들 때 값이 모두 같아도 된다면 다음과 같은 방식도 사용가능
- 변수 3개를 만들면서 모두 같은 값을 할당.

\*이렇게 변수1 = 변수2 = 변수3 = 값 형식으로 변수 여러 개를 =로 연결하고 마지막에 값을 할당해주면 같은 값을 가진 변수 3개가 만들어집니다.



```
x = y = z = 10  
print(x)  
print(y)  
print(z)
```

```
10  
10  
10
```

## 변수 (Variable)

변수 삭제하기

\*del을 사용

```
x = 10  
del x  
x
```

```
-----  
NameError                                Traceback (most recent call last)  
<ipython-input-22-3dde88e24a37> in <module>()  
      1 x = 10  
      2 del x  
----> 3 x  
  
NameError: name 'x' is not defined
```

SEARCH STACK OVERFLOW

## 변수 (Variable)

빈 변수 만들기

\*None값을 사용

```
[23] x = None  
      print(x)
```

None

```
[24] x # 아무것도 출력되지 않음
```

## 문제풀기

### 변수 (Variable)

1. 변수 `student_age` 만들고, 값 13를 할당
2. `student_age` 출력
3. 변수 `adult_age` 만들고, 값 18를 할당
4. `adult_age` 출력
5. `adult_age`를 `print()` 함수를 사용해 출력

## 문제풀기

물음표(?)에 들어가는 것은?

Complete the code to return the output

```
revenue = 100  
cost = 30  
profit = ? - cost  
print(profit)
```

70

FILL IN THE BLANKS

30

profit

cost

revenue



## 문제풀기

### 변수 (Variable)

1. 변수 david\_height 만들고, 값 180 를 할당
2. 변수 David\_height 만들고, 값 170 를 할당
3. 변수 david\_Height 만들고, 값 175 를 할당
4. 3개 모두를 출력

## 변수 (Variable)

“Variables are the dynamic heart of programming. Variables make code more than a static set of instructions. They allow logic to occur, enabling developers to measure time, analyze data, and customize the program to the user.”

“변수는 프로그래밍의 역동적인 핵심입니다. 변수는 코드를 정적 명령어 세트 이상으로 만듭니다. 이를 통해 로직이 발생하여 개발자가 시간을 측정하고 데이터를 분석하고 사용자에게 맞게 프로그램을 사용자 정의 할 수 있습니다.” (google translate)

## 변수 (Variable)

1. Be Clear (명확하게)
  - N\_FLT
  - dRange vs. dateRange
  - tmp, x, i (for loop에서는 쓰임)
2. Be Concise (간결하게)
  - ThisPersonsFavoriteColor
3. Be Consistent (일관성있게)
  - PascalCase, camelCase, snake\_case, lowercase, or hyphen-case
  - Hungarian Notation(헝가리안 노테이션) - 예: iAge, sName

## 문제풀기

### 변수 (Variable)

1. '당신을 위하여'란 의미의 변수가 꼭 필요합니다.
2. 변수 4you 만들고, "I Love You"를 저장
3. 출력
4. 4you를 꼭 쓰고 싶다면 어떤 방법이 좋을까요?

## 문제풀기

### 변수 (Variable)

1. 개발팀장님한테 +\$App을 변수로 만들라고 지시를 받았습니다.
2. 변수 +\$App 만들고, “돈 왕창버는 앱, 플러스 달러”를 저장
3. 출력
4. 위의 경우는 어떤 방법이 좋을까요?

## 문제풀기

### 변수 (Variable)

1. 개발팀장님이 특별히 부탁한 변수 '만약에'를 만들어야 합니다.
2. 변수 if 만들고, "만약에"를 저장
3. 출력
4. ㅋㅋ 어떡할까요?

## 문제풀기

1. 다음과 같은 음료 메뉴를 저장하고 출력해 봅시다.

Americano 1500

Latte 2000

Espresso 1700

Mocha 2500

식혜 2000

수정과 1900

2. Latte와 식혜 가격이 200원이 올랐습니다. 값을 바꾸고 다시 출력해 봅시다.
3. americano 3잔, mocha 2잔, Latte 8잔 주문 후 총 가격을 물어봅니다. 얼마인가요?
  - 총 가격을 변수로 만든후, 이 총가격 변수를 출력합니다.
4. americano 4잔, Latte 4잔, 수정과 1잔 주문 후, 20% 할인쿠폰을 적용, 총 가격을 물어봅니다. 얼마인가요?
  - 할인쿠폰을 변수로 만든후, 이 변수를 계산에 적용합니다.



# 01-3 자료형(Type)



## 정수와 실수 (int & float)

### 정수(int)

- 영어로 *integer*, 줄여서 파이썬에서는 *int*라고 표현
- 정수끼리 더하거나 곱하거나 빼면 정수
- 정수끼리 나누면 실수가 나올 수 있으나, 나눗셈의 몫만을 구하려면 *//*연산자를 이용

```
a = 5//3
```

- 실수를 정수로 바꾸려면 *int*를 이용

`a=int(5.4)`라고 하면 `a`는 5를 값으로 가지게 된다.

```
[2] int_div = 5 / 3  
int_div
```

```
1.6666666666666667
```

```
[4] int_div_int = 5 // 3  
int_div_int
```

```
1
```

## 정수와 실수 (int & float)

### 실수

- 부동소수점(floating point)이라는 표현법을 이용해 소수점을 표시할 수 있는 숫자
- 어느정도의 계산 정확도는 가지지만, 계산에 있어서 완벽한 정확성은 가지지 않는다.

```
0.1+0.1+0.1 == 0.3
```

- 정수를 실수로 바꾸려면 float를 사용

a=float(5)라고 하면 a는 5.0을 값으로 가지게 된다.

## 문제풀기

## int 혹은 float?

1. int와 float중?
  - 2
  - 26.0
  - 3
  - 10.5
2. 변수 a에 계산  $2 + 5.2$  후, 출력
3. 변수 b에 계산  $1 * 3$  후, 출력
4. 변수 c에 계산  $1.0 * 3$  후, 출력

## 문자열 (string)

1. 텍스트 두개를 더하면 문자열이 이어붙여짐
2. 텍스트는 더하기만 가능하고, 빼기(-) 등 다른 계산은 불가능



```
text = '2015' + '1991'  
text
```

```
'20151991'
```



```
text = '2015' - '1991'  
text
```

```
-----  
TypeError                                Traceback (most recent call last)  
<ipython-input-8-57cadd6c2ad1> in <module>()  
----> 1 text = '2015' - '1991'  
      2 text
```

```
TypeError: unsupported operand type(s) for -: 'str' and 'str'
```

[SEARCH STACK OVERFLOW](#)

## 문제풀기

## 문자열 (string)

string과 string이 아닌것은?

1. 5
2. 'c'
3. 'Ireland'
4. 7.0
5. 'Colin'

## type() 함수

데이터타입을 확인 하기 위해서는 **type()** 을 사용하여 확인



```
print(type(123))  
print(type(12.3))  
print(type('123'))
```

```
<class 'int'>  
<class 'float'>  
<class 'str'>
```

## 문제풀기

**type() 함수**

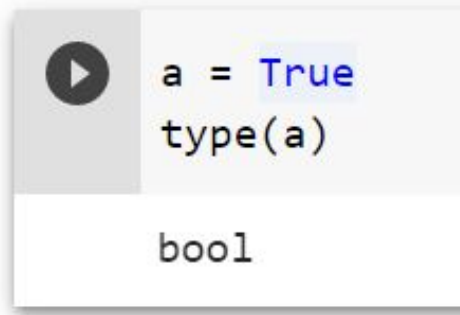
각 값을 변수에 넣고 (혹은 바로), **type**를 맞춰봅시다.

1. 5
2. 'c'
3. 'Ireland'
4. 7.0
5. 'Colin'

## 불(bool)

불리언(boolean)의 줄임말로 불(bool) 자료형이란 참(True)과 거짓(False)을 나타내는 자료형. 불 자료형은 다음 2가지 값만을 가질 수 있음:

- True - 참
- False - 거짓



```
a = True
type(a)

bool
```

※ True나 False는 파이썬의 예약어로 true, false와 같이 사용하지 말고 첫 문자를 항상 대문자로 사용



## 불(bool)

불 자료형은 조건문의 반환 값으로도 사용. 조건문에 대해서는 if문에서 자세히 배우겠지만 잠시 살펴보고 넘어가자.



`1 == 1` 은 "1과 1이 같은가?"를 묻는 조건문. 이런 조건문은 결과로 **True** 또는 **False**에 해당되는 불 자료형을 돌려줌. 1과 1은 같으므로 **True**를 돌려줌.

## 불(bool)

2는 1보다 크기 때문에  $2 > 1$  조건문은 **True**를 돌려준다.



2는 1보다 작지 않기 때문에  $2 < 1$  조건문은 **False**를 돌려준다.



## 문제풀기

### 불(bool)

다음 중 Boolean 값으로 표현할 수 있는 것은?

- 이름
- 나이
- 나이가 18세 이상
- 기혼? 미혼?

## 문제풀기

### 불(bool)

다른점은?

1. true vs. True
2. false vs. 'False'
3. False vs. 'False'
4. True vs. False

## 문제풀기

### 불(bool)

<class 'bool'> 의 출력이 나오는것은?

1. `print(type('True'))`
2. `print(type(5))`
3. `print(type('Sam'))`
4. `print(type(True))`

## None

python에서 NoneType은 값이 존재하지 않음(None Object)을 나타내는 Type입니다



```
a = None  
type(a)
```

```
NoneType
```

## 실습

**None**

a에 아무것도 지정하지 않은후, **type**을 리턴하면 어떤 결과가 나올까요?



```
a = None  
type(a)
```

```
NoneType
```



# 02 파이썬(Python) 기초와 실습 2



## 입력(input)

- 파이썬(Python)에서 키보드 입력을 받을 때는 input() 함수를 사용
- 프로그래밍 공부, 알고리즘 학습에 반드시 필요한 함수



```
age = input("Enter your age: ")  
print(age)
```

Enter your age:

## 입력(input)



```
name = input('이름을 입력하세요: ')\n\nprint('당신의 이름은', name, '이군요')
```

이름을 입력하세요: Tom  
당신의 이름은 Tom 이군요

## 문제풀기

### 입력(input)

1. `what_is_your_type` 이름의 변수를 만듭니다.
2. `input`으로 “넌 타입이 뭐니?”을 유저한테 물어봅니다.
3. `what_is_your_type`를 출력합니다.
4. `what_is_your_type`의 `type()`을 출력합니다.

## 입력(input)

input() 함수의 리턴값은 string(문자열)이기 때문에 계산을 하려면, 정수(int), 실수(float)로 바꿔야 합니다.



```
s1 = input('정수1 입력하세요: ')\ns2 = input('정수2 입력하세요: ')\ni1 = int(s1)\ni2 = int(s2)\nprint('두 수의 합은 :', i1 + i2)
```

```
정수1 입력하세요: 11\n정수2 입력하세요: 24\n두 수의 합은 : 35
```

## 입력(input) - split()

split() 함수를 사용하면, 두 개의 입력을 받을 수 있습니다. (공백으로 분리하는 것이 기본값)



```
s1, s2 = input('두 수를 입력하세요: ').split()
i1 = int(s1)
i2 = int(s2)
print('두 수의 합은 :', i1 + i2)
```

```
두 수를 입력하세요: 22 9
두 수의 합은 : 31
```

## 문제풀기

### 입력(input)

1. 2개의 실수(float) 값의 입력을 받아봅니다. `split()`를 사용해도 됩니다.
2. 변수이름은 자유롭게 만들어 봅니다.
3. 실수 값을 계산해야 함으로, `float()` 함수를 사용해서 `string`를 `float`로 바꿉니다.
4. 만든 2 변수를 더한 후, 새로운 변수에 저장합니다.
5. 이 합한 변수를 '두 수의 합은 .' 을 출력합니다.



# 02-2 조건문 (if)

## 조건문 - if

특정 조건에 따라 다른 동작을 할 수 있도록 해 주는 구문



```
people = 12
apple = 9
if people > apple:
    print('사람이 너무 많아! 몇 명은 못먹겠네')
```

사람이 너무 많아! 몇 명은 못먹겠네

구조

1. if 예약어 : 조건문의 시작을 알림
2. 조건: `people > apple`와 같이 참/거짓을 판단할 수 있는 조건
3. : 조건이 끝났다는걸 표현한하는 명령
4. 실행하고자 하는 코드. 코드는 **탭(tab)**키를 이용해서 들여서 쓴다.
  - 예. `print('사람이 너무 많아! 몇 명은 못먹겠네')`



## 조건문 - if

조건식에 대소비교가 쓰인 경우



```
if 3<5:  
    print("조건식이 True이므로 실행됩니다.")  
if 3>5:  
    print("조건식이 False이므로 실행되지 않습니다.")
```

위의 예처럼 3은 5보다 작으므로 3<5가 참인 if문만 실행됩니다.

## 조건문 - if

조건식에 True/False가 쓰인 경우



```
if True:  
    print("조건식이 True이므로 실행됩니다.")  
if False:  
    print("조건식이 False이므로 실행되지 않습니다.")
```

조건식이 True이므로 실행됩니다.

조건문이 참인 True의 if문만 실행됩니다.

## 조건문 - 들여쓰기

‘들여쓰기’는 파이썬 코더한테 실수(혹은 에러)가 잘 일어나는 부분입니다.



# 들여쓰기가 잘못된 예제

```
a, b = 4, 6
```

```
if a>b:
```

```
print("a가 b보다 크다")
```

File "<ipython-input-9-13097351e3c6>", line 4

```
print("a가 b보다 크다")
```

^

IndentationError: expected an indented block

SEARCH STACK OVERFLOW

## 실습

### 조건문 - 들여쓰기

아래에 똑같은 예제에서, 유저 Lady는 들여쓰기 잘못해서 **tab**을 두번을 눌렀습니다. 어떤 결과가 나왔을까요?



# 들여쓰기가 잘못된 예제

a, b = 4, 6

if a>b:

print("a가 b보다 크다")

File "<ipython-input-9-13097351e3c6>", line 4

print("a가 b보다 크다")

^

IndentationError: expected an indented block

SEARCH STACK OVERFLOW

## 조건식 - 숫자 비교

- 크다
  - $0 < 10$
  - $10 > 11$
- 크거나 같다, 작거나 같다
  - $3 \leq 10$
  - $15 \geq 10$
- 같다
  - $5 == 5$
- 같지 않다
  - $5 != 10$


비교의 결과는 True 또는 False

## 조건식 - **boolean(bool)**연산

- **and**
  - 두 조건이 모두 참인지를 체크
- **or**
  - 두 조건 중 하나라도 참이다
- **not**
  - **true/false**를 뒤집기 위해 사용

## 조건식 - boolean(bool)연산

- 예. a는 20대이다.



```
20 <= a and a < 30
```

- 예. a는 18세 미만 또는 60세 이상이다.



```
a < 18 or 60 <= a
```

## 문제풀기

### 조건식

1. 유저에게 현재 시간을 물어봅니다.
2. 변수 `hour`는 `int type`의 값을 저장합니다.
3. `hour`에 저장된 현재 시간이 12보다 작을때, '오전입니다.'를 출력
4. `hour`에 저장된 현재 시간이 12보다 클때, '오후입니다.'를 출력



## 문제풀기

### 조건식

어떤 숫자가 3의 배수인지 알아보기 위해서는 그 숫자를 3으로 나눈 나머지가 0인지 알아보면 됩니다.

1. 유저에게 원하는 숫자를 물어봅니다.
2. 입력한 숫자가 3의 배수일때, '3의 배수'를 출력
3. 입력한 숫자가 3의 배수 아닐때, '3의 배수가 아님'를 출력

% remainder - mod - %는 어떠한 값을 나누고 난 나머지를 구해주는 연산자.

```
print('33 % 4 = ', 33 % 4)  
print('5 % 3 = ', 5 % 3)  
print('10 % 5 = ', 10 % 5)
```

```
33 % 4 = 1  
5 % 3 = 2  
10 % 5 = 0
```

## 블럭 (block)

1. 함께 실행 되는 하나의 코드 덩어리
2. 들여쓰기로 블럭을 구분한다.
3. 들여쓰기가 어긋나면 오류가 발생한다.
4. 블럭 안에 다른 블럭이 들어갈 수 있다.
5. 내부의 블럭은 외부의 블럭에 종속적
6. 파이썬 코드 전체를 하나의 블럭으로 볼 수 있다.



```
a = 10
b = 7
if a > 5:
    print(a)
    if b < 8:
        print('b < 8')
    if b > 7:
        print(a)
        print('b > 7')
```

## 조건문 - if else

else

- if의 조건이 맞지 않는 경우 항상 실행
- 반드시 if뒤에 나와야 한다.



```
mine = "가위"  
yours = "바위"  
DRAW = "비김"  
if mine == yours:  
    result = DRAW  
else:  
    result = '이기거나 지거나'  
result
```

## 조건문 - elif

elif - (else if)의 줄임

- else 와 if의 결합으로 조건이 맞지 않는 경우 다른 경우를 검사



```
mine = 1
SCISSOR = 1
ROCK = 2
if mine == SCISSOR:
    result = '가위'    # 조건이 참일 때 실행
elif mine == ROCK:
    result = '바위'    # 다른 조건이 참일 때 실행
else:
    result = '나머지'  # 조건이 거짓일 때 실행
```

## 문제풀기

### 조건문 - if, else, elif

gender의 값을 검사해서 “남자”라면 “남자입니다.”라고 출력하고 “여자”라면 “여자입니다.”라고 출력하고, 둘 다 아닐 경우 “젠더 뉴트럴입니다.”라고 출력하도록 만들어 보세요.

\*if, elif, else를 이용해야 합니다.

\*gender의 기본값은 ‘남자’로 설정하고, 다른 값으로 바꾸면서 테스트해 봅니다.



## Reference

<https://withcoding.com/64>

<https://Insideout.tistory.com/entry/Python-%ED%8C%8C%EC%9D%B4%EC%84%A0-%EC%A3%BC%EC%84%9D-%EC%82%AC%EC%9A%A9%EB%B2%95%ED%95%9C%EC%A4%84%EC%97%AC%EB%9F%AC%EC%A4%84%EB%8B%A8%EC%B6%95%ED%82%A4>

<https://www.datacamp.com/>

<https://dojang.io/mod/page/view.php?id=2176>

<https://www.freshconsulting.com/development-principle-1-choose-appropriate-variable-names/>

<https://wikidocs.net/18507>

<https://programmers.co.kr/learn/courses/2>

[https://zetawiki.com/wiki/%ED%8C%8C%EC%9D%B4%EC%8D%AC\\_%EC%9E%90%EB%A3%8C%ED%98%95\\_%ED%99%95%EC%9D%B8\\_type\(\)](https://zetawiki.com/wiki/%ED%8C%8C%EC%9D%B4%EC%8D%AC_%EC%9E%90%EB%A3%8C%ED%98%95_%ED%99%95%EC%9D%B8_type())

<https://wikidocs.net/17>

<https://withcoding.com/65>