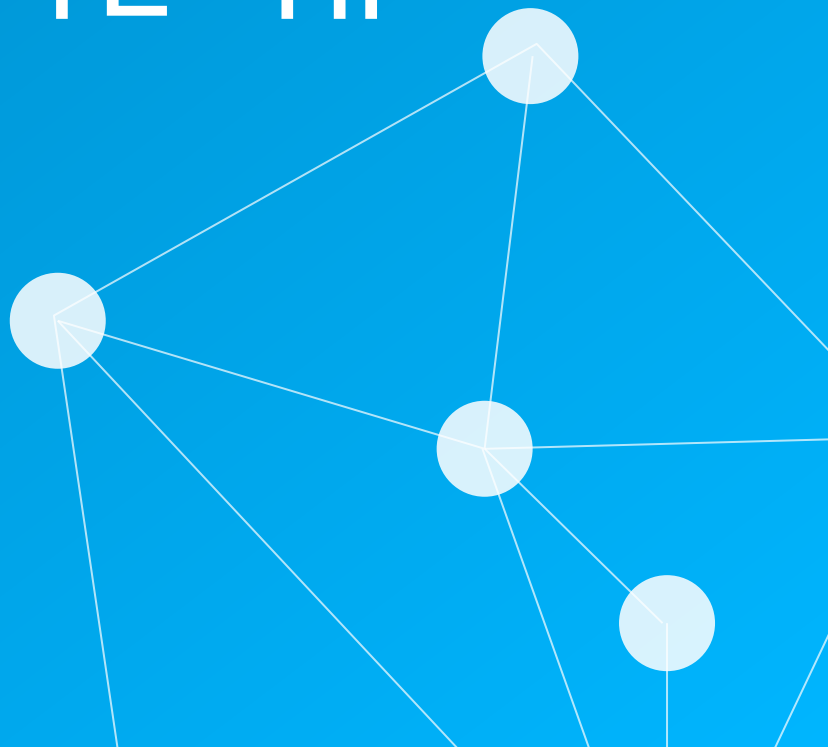
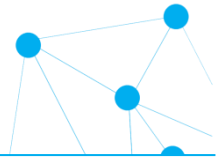
파이썬
자료구조

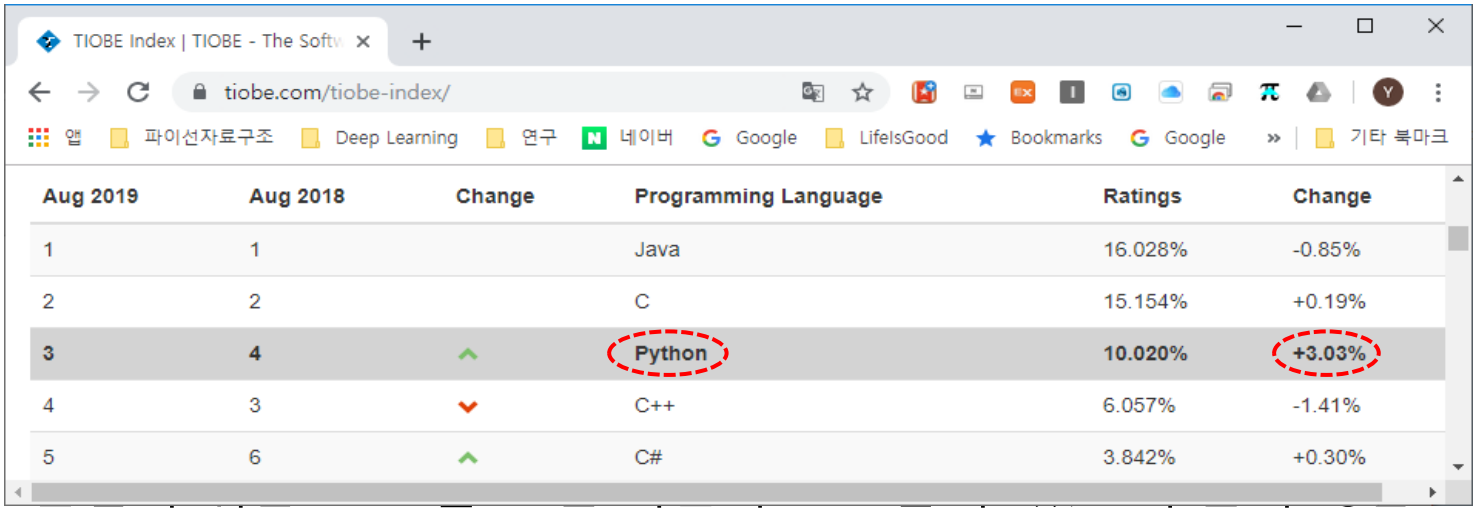
파이썬 리뷰



파이썬이란?



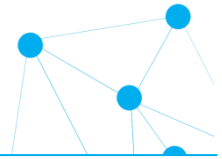
- AI와 빅데이터의 부상과 함께 최근 각광받고 있는 언어



Aug 2019	Aug 2018	Change	Programming Language	Ratings	Change
1	1		Java	16.028%	-0.85%
2	2		C	15.154%	+0.19%
3	4	▲	Python	10.020%	+3.03%
4	3	▼	C++	6.057%	-1.41%
5	6	▲	C#	3.842%	+0.30%

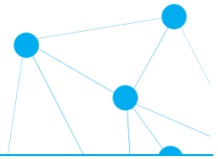
  - 인터프리터 방식 ←→ 컴파일 방식
  - 스크립트 모드 지원
  - 통합개발환경: IDLE, 주피터 노트북, 파이참, 비주얼 스튜디오
  - 파이썬 공식 홈페이지의 다운로드 페이지 (http://www.python.org/downloads)에서 윈도우용 파이썬 언어 패키지를 다운로드

파이썬 특징



- 파이썬은 문법이 쉬워 빠르게 배울 수 있다
- 파이썬은 간결하다
- 파이썬은 개발 속도가 빠르다
- 확장성이 좋아 여러 응용분야의 프로그래밍에 유리하다

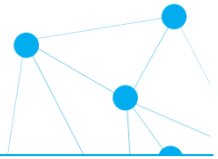
파이썬 참고용 교재



- 점트 투 파이썬, 박응용 (https://www.wikidocs.net/book/1)

- ....

자료형, 리터럴과 변수



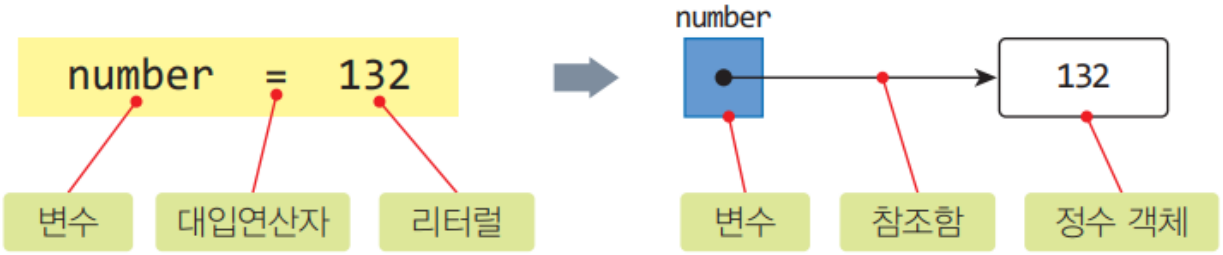
- 리터럴과 자료형

분류	내장 자료형	리터럴
수치	정수(int)	10 -30 0xfffe 073
	실수(float)	3.14 -0.45 123.032E-13
	복소수(complex)	complex(1,2) 1+2j 4+5j
	부울(bool)	True False
시퀀스	문자열(str)	'game' "over" "C"
	리스트(list)	[] [0, 1, 2, 3] [0, 'hello', 3.14]
	튜플(tuple)	(0, 1, 2, 3) ('hello', 'world', 'game')
매핑	딕셔너리(dict)	{ 3.14 : "phi", 4.5 : "score" }
집합	집합(set, frozenset)	{ 1, 2, 3 } {'one', 'two', 'three' }

- 변수(variable)

```
number = 132                    # 변수 생성 및 사용 문장
number = number + 8             # 변수 사용 문장

pi = 3.14                       # float 변수
comp = 1 + 2j                   # complex 변수
isValid = True                  # bool 변수
msg = 'game over !!!'           # str 변수
A = [0, 1, 1, 2, 3, 5, 8, 13 ]  # list 변수
```

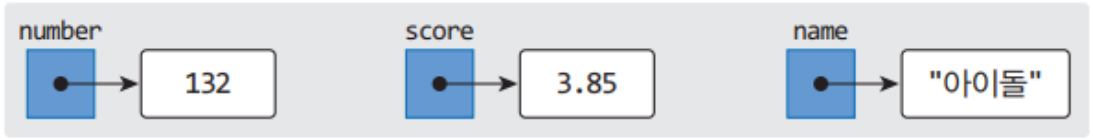
- 변수 이해하기



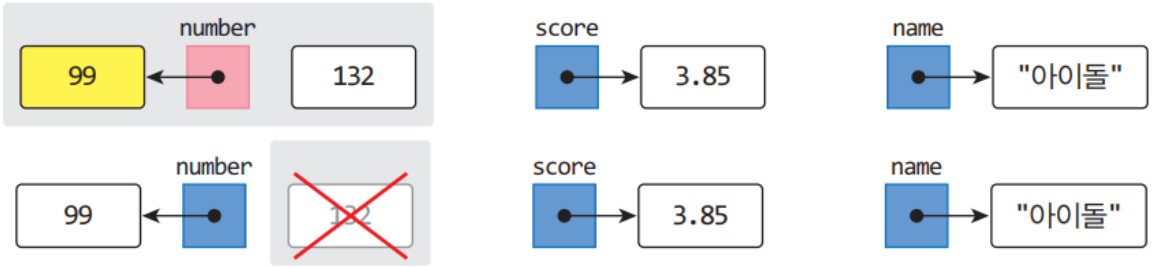
- 파이썬에서는 모든 자료가 클래스로부터 만들어진 객체이다.
- 변수는 다른 객체를 참조하는 참조자 또는 포인터의 역할을 한다.

- 변수의 동작 정확히 이해하기

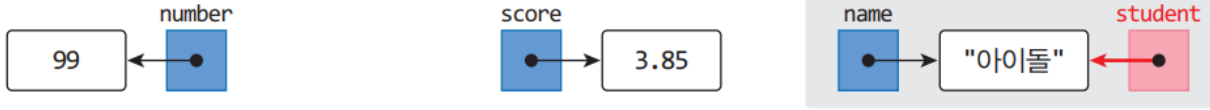
```
number = 132
score = 3.85
name = "아이돌"
```



number = 99

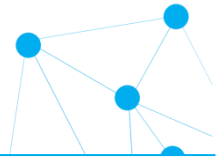


student = name



number = None

파이썬 자료형



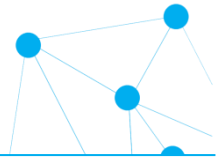
- 기본 자료형
  정수, 실수, 문자, 부울

- 정수: int
  10진수 (15, -30), 8진수(0o177), 16진수(0x8ff)
  x = 0o123

- 실수: float
  12.5, -50,2

- 부울: bool
  True, False

파이썬 자료형



- 문자열 : str

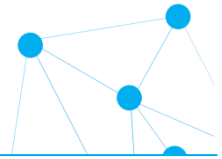
  "seoul", 'seoul', """seoul""", '''seoul'''

- 여러 줄인 문자열: ''' 혹은 """ 사용

```
'''
    Life is too short.
    You need python.
'''
```

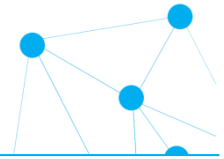
```
"""    Life is too short.
    You need python.
"""
```

파이썬 자료형 확인



- type 함수
  type(자료)

파이썬 주석

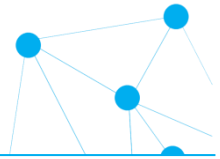


- 한 줄 주석: # 으로 시작

- 여러 줄 주석: """ 로 시작 """ 로 끝남

```
"""
    .....
    .....
"""
```

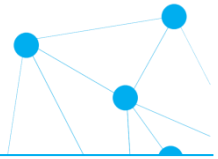
파이썬 출력 (기초)



- (화면) 출력: print() 함수
  정해진 양식으로 출력

```
print(a, b, c)
print(2, "3", sep= " ") // print(2, "3")
print(a, b, sep=":")
print(a, b, end=" ")

a = 10;
print(type(a))
```

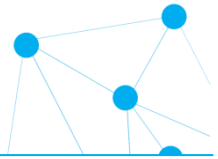
파이썬 출력 (기초)



- (화면) 출력: print() 함수
- 원하는 출력 양식에 맞게 출력: %
- 문자열 % (출력값1, 출력값2, ...) 형식
- 출력값에 대응하는 형식을 문자열에 포함
- %d, %3d, %f, %5.2f, %s, %+10s

```
price = 24
item = "banana"
print(" %s %d" % (item, price))
```

파이썬 입력 (기초)

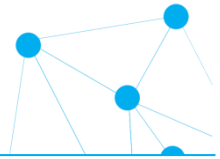


- (키보드) 입력: input() 함수
- input()는 입력되는 한 줄 자료를 문자열로 반환

```
n = input()              // 12  13  "12  13"
name = input("Input your name")
score = int(input())
print(n, name, score)
```

- 숫자를 입력해도 문자열로 반환하기 때문에, 변수에 숫자로 저장하려면 type 변환을 하는 것이 필요함

파이썬 입력 (기초)



- input().split() 함수: 한 줄로 입력되는 자료를 분리함
- input()는 입력되는 한 줄 자료를 문자열로 반환

```
a, b = input().split()  // 기본적으로 공백을 기준으로 분리
   12   13
```

```
x, y = input().split(":")
```

```
x, y = int(x), int(y)
```

입출력 연습

1 출력문 (print 문).

```
a = 20
b = 30
print(a, b)
print("a = ", a, "b= ", b)
```

2. 입력문 (input 문)

(1)

```
score = input("점수 입력: ")
print(score)  # 문자열
```

(2)

```
score = int(score)
print(score)  # 정수(2)
score = int(input())
print(score)
```

(3)

```
name = input()
print(name)
```

(4)

```
lastName, firstName = input().split()
print(lastName, firstName)
```

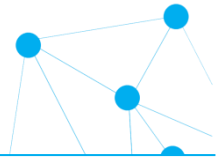
(5)

```
name = input()
score = int(input())
print(name,score)
```

(6)

```
name, score =input().split()
score = int(score)
print(name,score)
```

파이썬의 연산자

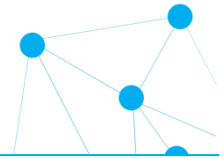


- 나눗셈 연산자 변경
  - 연산자 / : 실수의 나눗셈 (결과가 실수)
  - 연산자 //: 정수 연산(floor division)
- 이항 연산자 ** (예: 2**3) 2^3
- 단항 연산자 ++, -- 제공 않음
  - x += 1
- 관계 연산자 >, <, >=, <=, ==, !=
- 부울 연산자: or, and, not
- in과 not in 연산자

```
'a' in 'banana'          # True
'seed' in 'banana'       # False
```

```
A = [0, 1, 1, 2, 3, 5, 8, 13 ]
if 3 in A :
while 4 in A :
```

조건문



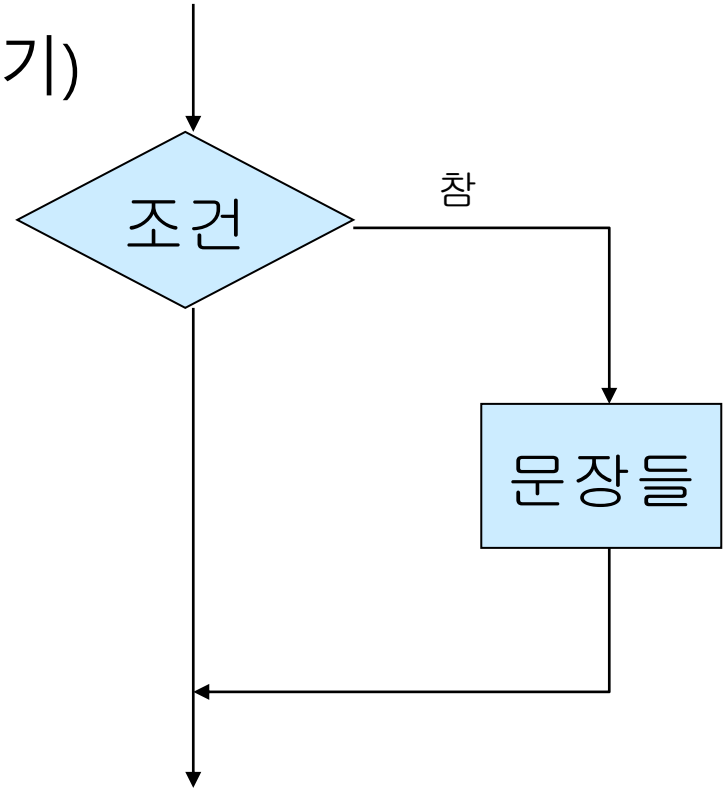
- if, if else, if elif

조건문 if

- 조건문: if, if else, if elif
- if 조건:
  문장들 (indentation 들여쓰기)

- 예:

```
if a > b:
    temp = a
    a = b
    b = temp
```



if 문 연습 – 세 수 정렬

1. 세 수 정렬하기

```
x,y,z = input().split()
# print(x+y+x) 출력결과 확인해보기
x = int(x)
y = int(y)
z = int(z)
# x, y, z = int(x), int(y), int(z)
if x>y:   # x가 y보다 크면
   temp = x  # x와 y 교환
   x = y
   y = temp
```

```
if y>z:   # y가 z보다 크면
   temp = y  # y와 z 교환
   y = z
   z = temp

if x>y:   # x가 y보다 크면
# x와 y 교환

print(x,y,z)
```

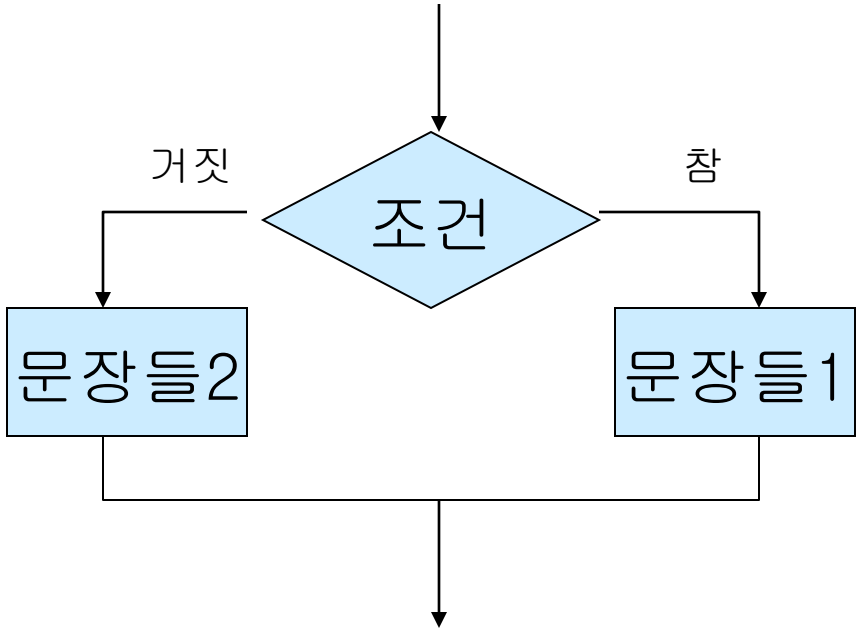
조건문 if else

- if else 문

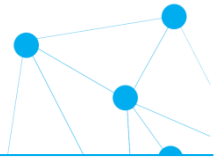
```
if 조건:
    문장들1
else:
    문장들2
```

- 예

```
if a > b:
    largest = a
else:
    largest = b
```

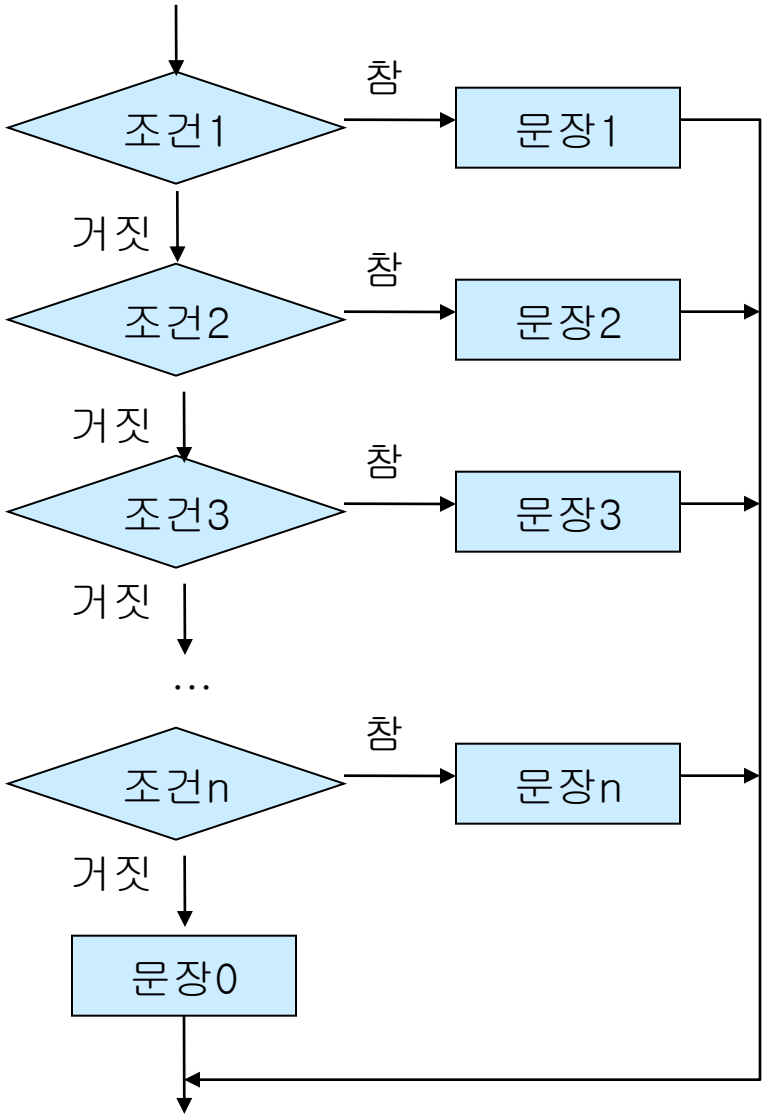


조건문 if elif

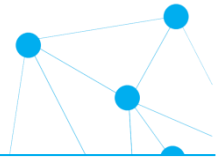


- if elif

```
if 조건1:
        문장1
elif 조건2:
        문장2
elif 조건3:
    ...
elif 조건n:
        문장n
else:
        문장0
```



if elif 문 연습

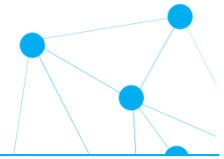


yy년도가 윤년인지 판별

yy가 400으로 나누어지면 윤년
그렇지않고 yy가 100으로 나누어지면 평년
그렇지않고 yy가 4로 나누어지면 윤년
그렇지않으면 평년

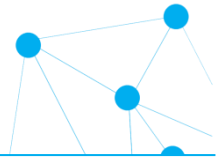
```
yy = int(input())
if yy % 400 == 0:
    isLeap = True
elif yy % 100 == 0:
    isLeap = False
elif yy % 4 == 0:
    isLeap = True
else:
    isLeap = False

if isLeap: # if isLeap == True
    print("The year %d is a leap year" % (yy))
else:
    print("The year %d is not leap a year" % (yy))
```

반복문



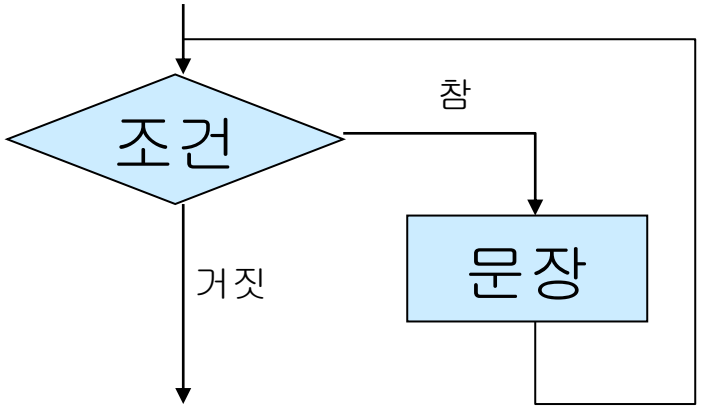
- while, for

반복문 while

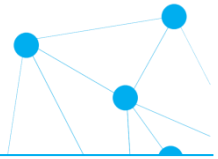


- while 문

```
while 조건:
    문장들
```



while 문 연습

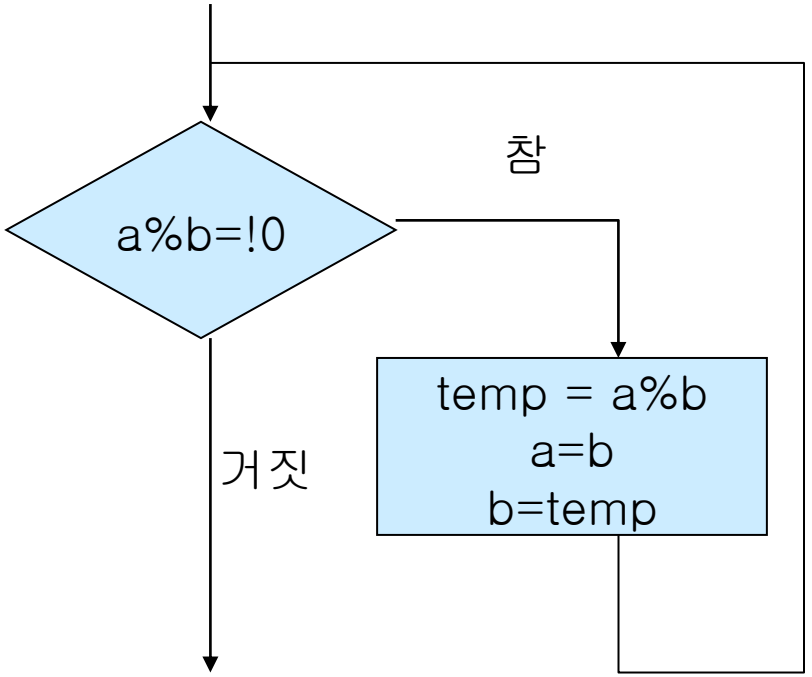


- 예:
  두 수 a,b의 최대공약수 출력

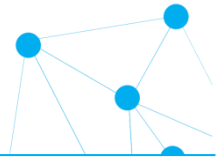
유클리드 알고리즘

gcd(a,b) = b if a%b = 0
gcd(b,a%b) otherwise

```
a,b = input().split()
a,b = int(a), int(b)
while(a%b != 0)
    temp = a%b
    a = b
    b = temp

print(b)
```



while 문 연습



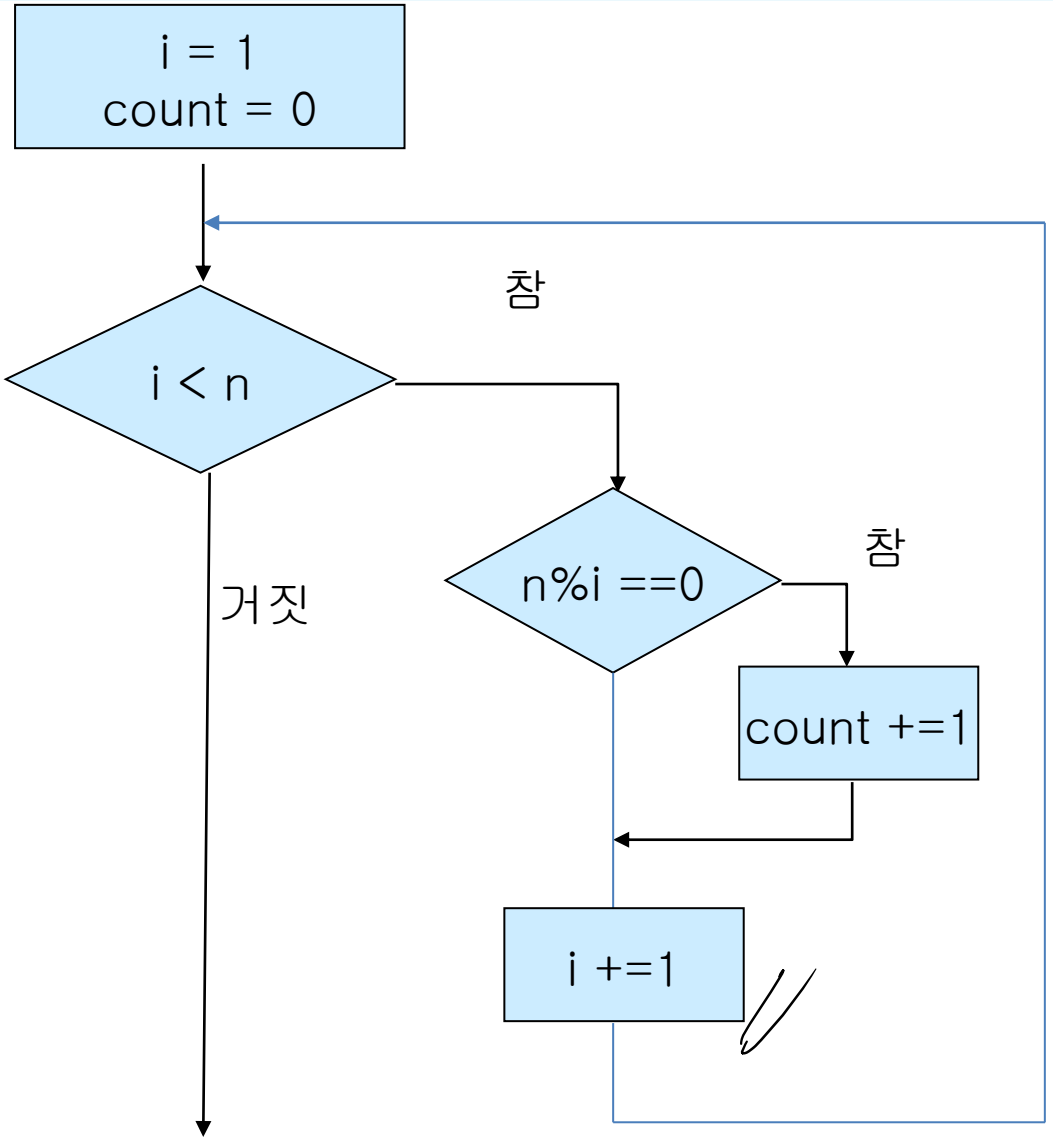
- 예: 양의 정수 n의 자릿수 개수와 자릿수 합 구하기

```
n = int(input())

sum = 0
count = 0

while n!= 0:
    count += 1
    sum = sum + n % 10
    n = n // 10

print(count, sum)
```

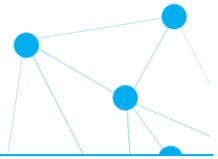
while 문 연습

- 예:
  양의 정수 n의
  약수 개수 출력

```
n = int(input())
i = 1
count = 0
while i <=n:
    if n%i == 0:
            count += 1
    i += 1
print(count)
```

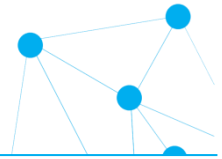


while 문 연습



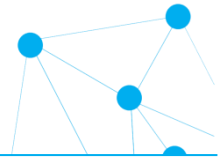
```
# 소수(prime number) 판별 1
   소수: 약수로 1과 자기 자신만을 가지는 양의 정수

n = int(input())
isPrime = True
i = 2
while i < n:
    if n % i == 0:
        isPrime = False
        break
    else:
        i += 1
if isPrime:
    print(n, " is a prime number")
else:
    print(n, " is not a prime number")
```

for 문



- for 변수 in range (또는 리스트, 튜플, 문자열):
  문장들

range 함수

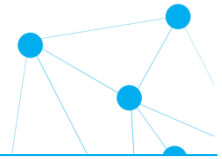


- range(2,10) # range(2,10,1)
  2부터 10 미만의 숫자를 포함하는 range 객체

- range(10) # range(0,10)
  0부터 10 미만의 숫자를 포함하는 range 객체

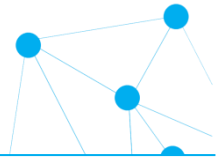
- range(2,10,3)
  2, 5, 8을 포함하는 range 객체

range 연습



```
for i in range(5):
    print(i)

for i in range(1,5):
    print(i)
```

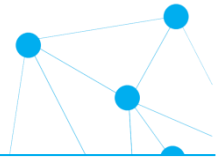
for 문 연습



- yy1 년도부터 yy2 년도 까지 윤년 수 구하기

```
yy1, yy2 = input().split()
yy1, yy2 = int(yy1), int(yy2)
count = 0
for yy in range(_____, _______):
    if yy % 400 == 0:
        isLeap = True # bool 형
    elif yy % 100 == 0:
        isLeap = False
    elif yy % 4 == 0:
        isLeap =True
    else:
        isLeap = False

    if isLeap:
        count += 1

print(count)
```

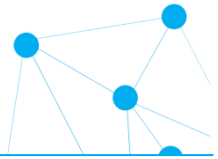
for 문 연습



- 약수 판별

```
n = int(input())
isPrime = True
for i in range(2,n):
    if n % i == 0:
        isPrime = False
        break

if isPrime:
    print(n, " is a prime number")
else:
    print(n, " is not a prime number")
```

소수 판별하는 효율적 방법



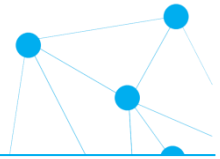
- 양의 정수 n이 2이상이고 root(n)보다 작은 약수가 없으면 소수이다

```
n = int(input())
isPrime = True
i = 2
while i*i <= n:  # while i <= root(n)
    if n % i == 0:
        isPrime = False
        break
    else:
        i += 1

if isPrime:
    print(n, " is a prime number")
else:
    print(n, " is not a prime number")
```

문자열, 리스트, 튜플, 집합, 딕셔너리

- 여러 개의 자료를 가질 수 있는 자료형

문자열(str)



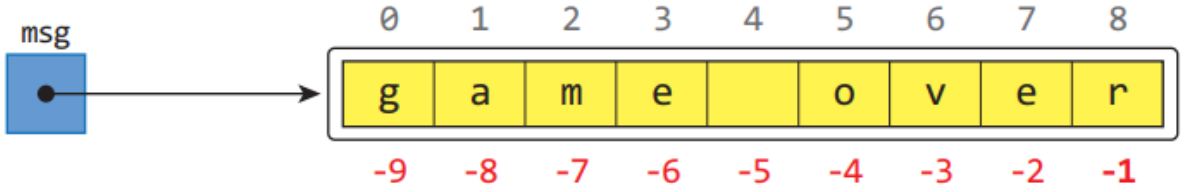
- 문자열

```
msg = 'game over'
hi = "hello world"
sum = "예전엔 " + hi + " 이제는 " + msg
print(sum)
```

```
C:₩WINDOWS₩system32₩cmd.exe
예전엔 hello world 이제는 game over
```

```
print(msg, '의 첫 글자는 ', msg[0])
print(msg, '의 끝 글자는 ', msg[-1])
```

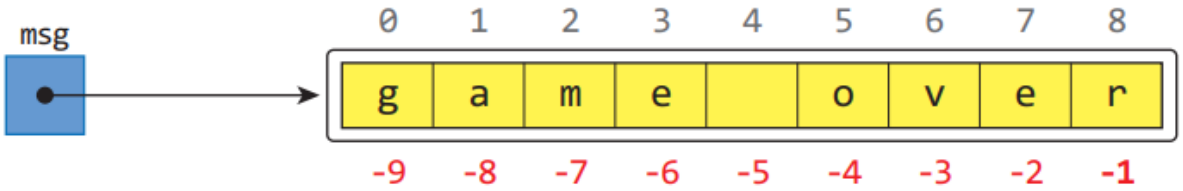
```
C:₩WINDOWS₩system3...
game over 의 첫 글자는 g
game over 의 끝 글자는 r
```




```
hobby = "테니스"
age = 21
score = 4.5
msg1 = " 당신의 학점은 %4.1f입니다" % score
msg2 = " 취미=%s, 나이=%d, 학점=%f" % (hobby, age, score)
```

문자열(str)

- 문자열
  msg = "game over"

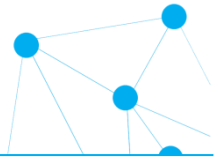


- 문자열 길이 함수
  len(msg)

- 문자열 특정 위치 문자 접근: 인덱스 이용
  문자열변수이름[인덱스] : 인덱스 0~문자열길이-1
  예: msg[i]

- 문자열은 변경 불가 (immutable 자료형)
  msg[0] = 'G' # 오류

리스트(list)

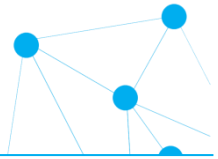


- 순서가 있는 원소(element, 요소, item:항목)들의 모임
- score = [80,90,85,95,70]
- print(score) # score 리스트 전체를 출력
- 리스트 원소를 접근할 때 인덱스를 사용: 리스트이름[인덱스]
  가장 앞에 있는 원소의 인덱스 0부터
  가장 마지막에 있는 원소 인덱스는 (리스트 길이-1)

  예: score[0] # 80
  score[4] # 70
  score[-1]도 가능 # score[4]

- 리스트는 원소들의 자료형이 달라도 된다
  L = [1, "momkey", 2, "hippo", 3, "tiger"]

리스트(list)



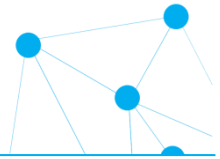
- 스마트한 배열

```
big3 = [ ]
lotto = [23, 34, 11, 42, 9]
big4 = [ '제이플라', '도티', '대도서관', '보람튜브' ]
print("lotto[1] = ", lotto[1])
big4[2] = '블랙핑크'
```

```
big3.append("알라딘")
big3.append("엘사")
big3.append("안나")
```

메소드	설명
s.append(item)	항목 item을 리스트 s의 맨 뒤에 추가한다.
s.extend(lst)	리스트 lst를 s에 추가한다.
s.count(item)	리스트에서 항목 item의 개수를 세고 그 개수를 반환한다.
s.index(item,[시작],[종료])	리스트에서 항목 item을 찾아 가장 작은 인덱스를 반환한다. 탐색의 시작 위치와 종료 위치를 지정할 수도 있다.
s.insert(pos, item)	pos 위치에 항목 item을 삽입한다.
s.pop(pos)	pos 위치의 항목을 s에서 꺼내고 반환한다.
s.remove(item)	항목 item을 s에서 제거한다.
s.reverse()	리스트 항목의 순서를 뒤집는다.
s.sort([key], [reverse])	항목을 정렬한다.

리스트 연산



- score = [80,90,85,95,70,80,99]

- 슬라이싱 (slicing)

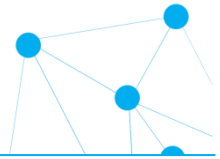
score[2:5] # score[2]부터 score[4]까지 원소들 리스트 [90,85]

score[0:2]
score[:2]
score[2:]

- 문자열 슬라이싱 (slicing)

string1 = “seoul”
String2 = string1[2:4]

리스트 연산

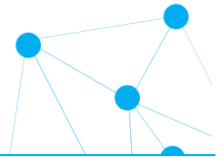


- score = [80,90,85,95,70]

- 리스트 원소 수정
  예: score[0] = 82

- 두 리스트 더하기(+)
  a = [10,30,20]
  b = [30,60,10]
  c = a+b # [10,30,20,30,60,10]

리스트 연산

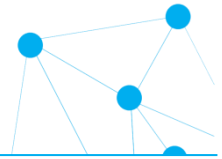


- score = [80,90,85,95,70]

- 리스트 반복하기 *
  예: score*2

- 리스트 길이 구하는 함수 len
  a = [10,30,20, 80, 25]
  l = len(a)

리스트 연산

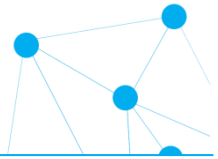


- score = [80,90,85,95,70]

- 리스트 수정

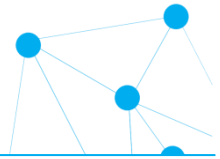
  score[2] = 87

리스트 관련 함수들



- score = [80,90,85,95,70]
- 리스트 변수 이름 뒤에 .을 붙여서 사용하는 함수
- 리스트의 마지막에 원소 추가: 함수 append
  score.append(20)
- a = []
  a.append(20)
  a.append(30)
  a.append(25)
- 리스트에서 원소 삽입 함수 insert
  score.insert(pos, element) # pos 번째에 element 삽입
  score.insert(0,50)

리스트 관련 함수들



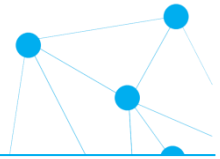
- score = [80,90,85,95,70]

- 리스트에서 원소 삭제 함수 remove
  score.remove(element) # 리스트에서 처음으로 나오는 원소 삭제
  score.remove(85)

- 리스트에서 원소 삭제 함수 pop
  score.pop() # 마지막 원소 삭제
  score.pop(2) # 2번째 원소 (index가 2인 원소)

- 리스트의 원소 삭제 함수 del
  del score[3] # index 3의 원소 삭제: [80,90,85,70]
  del score[:2] # slicing 사용 가능

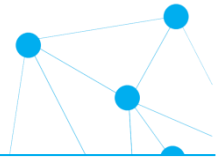
리스트 관련 함수들



- score = [80,90,85,95,70]
- 리스트 변수 이름 뒤에 .을 붙여서 사용하는 함수
- 리스트에서 원소 삭제 함수 pop
  score.pop() # 마지막 원소 삭제

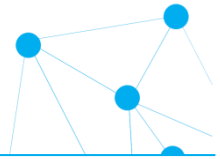
  score.pop(2) # 2번째 원소 (index가 2인 원소)

리스트 관련 함수들



- score = [80,90,85,95,70]
- 리스트에서 원소의 인덱스
  score.index(85)
- 리스트에서 원소 개수 세기 함수 count
  score.count(85)

리스트 관련 함수들

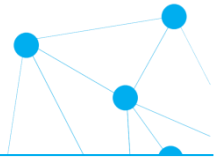


- score = [80,90,85,95,70]

- 리스트 확장 함수 extend

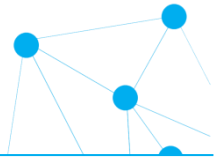
  score.extend([30,50]) # score = score +[30,50]

리스트 관련 함수들



- score = [80,90,85,95,70]
- 리스트 변수 이름 뒤에 .을 붙여서 사용하는 함수
- 리스트 원소들을 정렬하는 함수 sort
  score.sort()
- 리스트의 원소들을 역순으로 바꾸는 함수 reverse
  score.reverse()

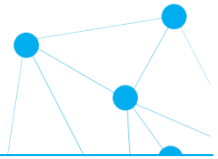
리스트에서 평균 구하기



리스트 scores에서 원소들 평균 구하기

```
for x in scores:
    sum = sum + x

avr = sum / n
print(avr)
```

리스트 이용 평균 관련 문제



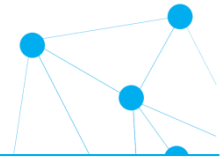
- 입력

```
5    # 점수 개수 n
60   # 점수1
70   # 점수2
80   # 점수3
90   # 점수4
85   # 점수5
```

평균 이상 점수 개수 구하기

```
n = int(input())
scores = []
sum = 0
for i in range(n):
    scores.append(int(input()))
    sum = sum + scores[i]

# print(scores)

avr = sum / n
print(avr)

cnt = 0
.....
```

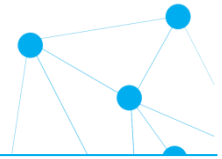
튜플



- 리스트와 유사하게 순서가 있는 원소들 모임
- t = (80,90,85,95,70)
- 인덱싱, 슬라이싱 가능
- 생성, 삭제 가능하지만 그 값을 바꿀 수 없다 (immutable 자료형)

  t[1] = 93 # 오류

튜플 연산



- t = (80,90,85,95,70)

- 두 튜플 더하기

  t1 = (20, 30)

  t2 = (40, 10, 50)

  t = t1 + t2 # (20,30,40,10,50)
- 튜플 곱하기

  t3 = t*2 # t3 = t1 + t1
- 튜플 길이 구하기 함수 len

  len(t1)