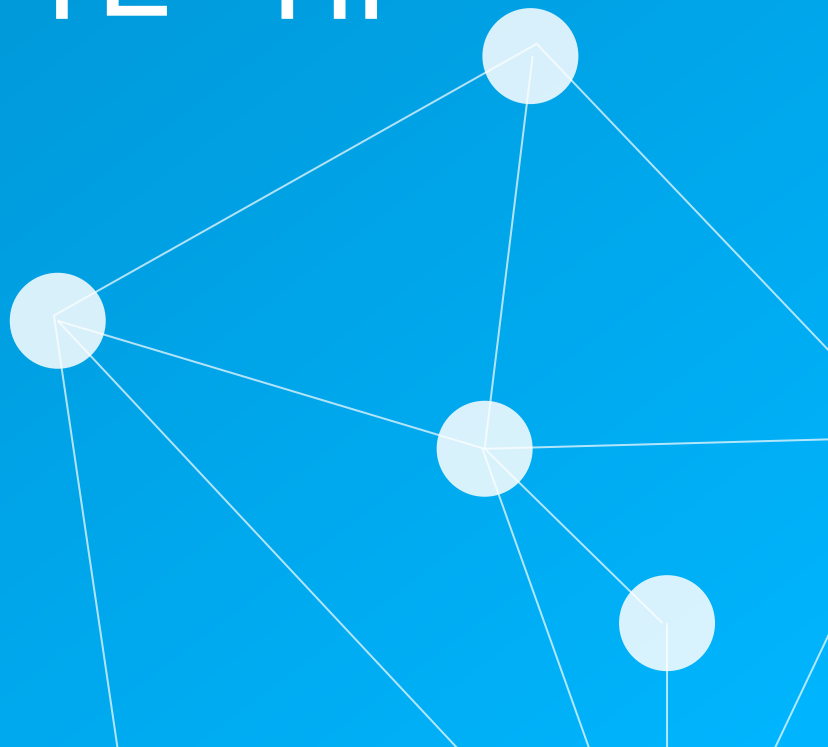
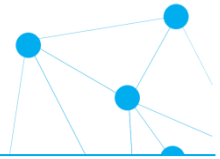
파이썬
자료구조

파이썬 리뷰



함수 매개변수 전달방법

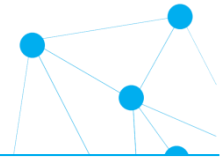


- 매개변수 전달방법
  call by object reference
  (call by assignment)

```
def increase1(a):
    a += 1


value = 10

increase1(x)
print(x)
```

```
def increase2(list):
    for x in list:
        x += 1


valueList = [10,20,30]

increase2(valueList)
```

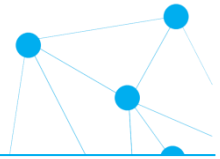
모듈



- 변수, 상수, 함수를 기능단위(모듈: module)로 분할하여 작성하고 필요할 때 현재 코드에 포함시켜 사용

- 모듈은 독립된 파이썬 코드로서 하나의 모듈에는 여러 개의 함수, 변수, 상수 등이 포함

모듈 사용 예



(1) 모듈 math에는 sin, cos, sqrt 등의 수학 함수를 포함

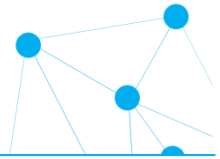
- import math
  print(math.sqrt(4.0)) # math 모듈의 sqrt 함수 사용
  print(math.pi) # math 모듈에 정의된 pi 값 상수

- from math import sqrt # math 모듈의 sqrt만 import
  print(sqrt(4.0)) # math 모듈의 sqrt 함수

(2) 파이썬 기본 모듈

```
>>> import sys
>>> sys.builtin_module_names
```

모듈 사용 예

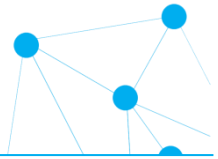


- 사용자 작성 모듈
  코드를 작성하여 my_module.py에 저장

```
e = 2.71828
def square(x):
    return x*x
```

```
import my_module
print(my_module.e, my_module.square(3.0))
```

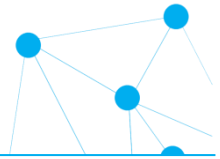
예외처리(exception handling)



- 예외: 문법(구문) 오류가 아닌 오류

```
>>> 4 / 0
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    4/0
ZeroDivisionError: division by zero

>>> a = [1, 2, 3]
>>> a[4]
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    a[4]
IndexError: list index out of range
```

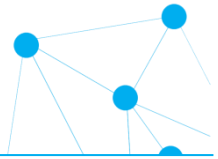
예외처리(exception handling)



- 문법(구문) 오류가 아닌 오류 발생할 때 처리
- try, except

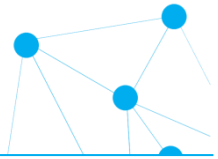
```
try:
    ....
except [발생오류 [as 오류메시지 변수]]:
    ....
```

예외처리(exception handling)



```
1) try:
        ....
   except:
# 오류가 발생하면 오류 종류에 관계없이 이 블록을 수행
# 오류가 발생하지 않으면 이 블록을 skip
        ....
2) try:
        ....
   except 발생오류:
# 오류가 발생하면 위의 오류 종류와 일치할 경우 이 블록을 수행
        ....

3) try:
        ....
   except 발생오류 as 오류메시지 변수:
# 오류가 발생하면 위의 오류 종류와 일치할 경우 오류메시지_변수에
# 오류 메시지 내용을 저장하고 이 블록을 수행
        ....
```

클래스



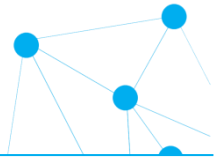
- OOP(Object Oriented Programming: 객체지향프로그래밍)의 핵심 개념
- 클래스 = 데이터 + 함수를 함께 정의
  함수를 클래스의 method라 부름

- 클래스: 물건을 만드는데 사용하는 틀
- 객체: 클래스 틀에 의하여 만들어진 인스턴스(instance)

- 클래스 정의

```
class 클래스이름[(상속 클래스명)]
    <클래스 변수 1>
    <클래스 변수 2>
        …
    def 클래스함수1(self, 매개변수1, …):
        함수 몸체
    def 클래스함수2(self, 매개변수1, …):
        함수 몸체
```

클래스 예



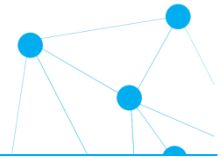
- OOP(Object Oriented Programming: 객체지향프로그래밍)의 핵심 개념
- 클래스 = 데이터 + 함수를 함께 정의
  함수를 클래스의 method라 부름

- 클래스: 물건을 만드는데 사용하는 틀
- 객체: 클래스 틀에 의하여 만들어진 인스턴스(instance)

- 클래스 정의

```
class 클래스이름[(상속 클래스명)]
    <클래스 변수 1>
    <클래스 변수 2>
        …
    def 클래스함수1(self, 매개변수1, …):
        함수 몸체
    def 클래스함수2(self, 매개변수1, …):
        함수 몸체
```

클래스 예



```
# 계산기 정의
class Calculator:
    numCalculators = 0 # 모든 클래스 객체가 공유
    def __init__(self):  # 초기화 함수 (생성자)
        self.value = 0
        self.memValue = 0
        Calculator.numCalculators += 1
    def __del__(self):  # 소멸자
        Calculator.numCalculators
    def clear(self):     # method
        self.value = 0
        return self.value

    def add(self, num): #  method
        self.value += num
        return self.value

    def currentVal(self):
        return self.value
```

```
# 계산기 사용 예

cal1 = Calculator()  # 클래스 객체 생성
                     # 자동으로 __init__ 호출

cal1.add(30) # add(a,30) => self:a, num: 30

print(cal1.currentValue())
```