Python Tutorial

따라하다보면 어느새 파이썬 중급 개발자!

조영일

2017.08.07~

print 사용법

- print(“hello”, “world”)
  - hello world

- print(“hello”)
- print(“world”)
  - hello
  - world

- print(“hello”, end=“”)
- print(“world”)
  - helloworld

기본 자료구조

- 배열 x
- 리스트
- 튜플
- 딕셔너리
- 집합

리스트

- 초기화
  - int_list = [1, 2, 3, 4, 5]
  - print(int_list)
- 인덱싱과 슬라이싱
  - print(int_list [0])
  - print(int_list [1])
  - print(int_list [2:4]

리스트

- 연산
  - 더하기
    - str_list = [“a”, “b”, “c”]
    - print(int_list + str_list)
  - 반복하기
    - print(int_list * 3)

튜플

- 수정불가
  - 그럼, 언제 사용하나?
    - 변경이 필요없는 static한 값을 담을 때
- 리스트와 유사한 수준의 연산

딕셔너리

- 연관 배열, 해시 맵
  - age_map = { ‘john’: 10, ‘rob’: 20 }
  - print(age_map)
  - print(age_map [‘john’])
  - print(age_map [‘rob’])
  - age_map[‘john’] = 31
  - age_map[‘mike’] = 45
  - print(age_map)

딕셔너리

- JSON과 유사한 문법
  - import json
  - str = json.dumps(age_map)
  - print(“str=“, str)
  - new_map = json.loads(str)
  - print(“new_map =“, new_map)

딕셔너리

- 연산
  - in
    - print(age_map[“phil”])
      - phil이라는 키가 존재하지 않으므로 exception 발생
    - if “phil” in age_map:
      - print(age_map[“phil”])

집합

- 초기화
  - 리스트로부터 생성
    - s1 = set([1, 2, 3])
    - print(s1)
    - s1.add(4)
    - print(s1)

집합

- 집합 연산들
  - 합집합, 교집합, 차집합, ...

조건문

- if 조건식:
  - …
- elif 조건식:
  - …
- else:
  - …

조건문

- int_list = [1, 2, 3]
- if int_list:
  - if int_list[0] == 1:
    - print("1")
  - elif int_list[0] == 2:
    - print("2")
  - else:
    - print("unknown")

반복문

- while 조건식:
  - …

- i = 0
- while i < len(int_list):
  - print(int_list[i])
  - i += 1

반복문

- for item in 리스트(or 튜플, 문자열):
  - ...

- for item in int_list:
  - print(item)

- for k in age_map:
  - print(k, age_map[k])

- for k, v in agep_map.items():
  - print(k, v)

반복문

- for i in range(len(자료구조)):
  - …
- for i in range(len(int_list):
  - print(i, int_list[i])

함수

- def 함수명(파라미터):
  - …
  - return

- def add(a, b):
  - return a + b
- print(add(3, 4))

- def mul(a, b)
  - return a * b
- print(mul(4, 5))
- print(mul('aaa', 3))

No actions

- if 조건문:
  - pass
- def 함수명():
  - return
- def 함수명():
  - None

IO

- from stdin
  - user_input = input()
- to stdout
  - print(“hello”)

IO

- old style
  - ~~f = open('data.txt', 'r')~~
  - ~~for line in f:~~
    - ~~print(line)~~
  - ~~f.close()~~

- new style
  - with open('data.txt', 'r') as infile:
    - for line in infile:
      - print(line)

IO

- to file
  - with open(‘newdata.txt’, ‘w’) as outfile:
    - outfile.write(...)

IO

- from stdin
  - input() 대신 sys.stdin으로부터 읽어들이는 방법
  - user_input = sys.stdin.read()

- to file
  - print() 대신 sys.stdout에 쓰는 방법
  - sys.stdout.write(message + “\n”)

OOP

- class ChildClass(ParentClass):
  - def method_name(self):
    - …

- instance = ChildClass ()
- instance.method_name()

OOP

- class Human():
  - def walk(self):
    - print(“I’m walking”)
  - def hear(self, message):
    - print(“I’m hearing”, message)
- foo = Human()
- man.walk()
- man.hear(”hey”)

OOP

- Tip!
  - format in print
    - def hear(self, message):
      - print(“I’m hearing ‘%s’” % (message)

OOP

- 생성자
  - def __init__(self):

- 소멸자
  - def __del__(self):

OOP

- class Human():
  - def __init__(self, name):
    - self.name = name
    - print(“I’m born now. My name is ‘%s’” % (self.name))
  - def __del__(self):
    - print(“I’m dead now”)

OOP

- @classmethod
  - 클래스에 스태틱한 메소드 지정
  - @classmethod
  - def touch(cls, obj):
    - print("I'm touching %s" % (obj))
  - Human.touch("my phone")

- @staticmethod
  - 유틸리티 함수를 클래스에 모아둘 때
  - @staticmethod
  - def read_config(config_file):
    - …
  - conf = Util.read_config("conf.xml")

Exception handling

- try:
  - …
- except IndexError as e:
  - …
- else:
  - …
- finally:
  - …

Module

- import json
  - data = json.loads(str)
- from my_class_file import MyClass
  - instance = MyClass()
- import json as j
  - j.loads(...)

Module

- 외부 모듈 설치
  - pip install 모듈명

  - https://pypi.python.org/pypi

Type hint

- def test_func(a : int, b : str) -> Dict[str, int]:
  - …

- a : str = “hello”
- b : List[int] = [1, 2, 3]

Type hint

- IntelliJ와 같은 IDE의 타입 힌트 시각화 지원
- mypy를 이용한 정적분석 검사