❝파이썬3 기초 요약(1) – 문법❞

★ 주석

print("안녕") **# 이 부분은 메모**
한 줄 내에서 '#' 기호 이후에 적는 모든 문장은
단순 메모입니다. 프로그램 실행에 영향이 없습니다.

★ 변수

fruit = 1 *# 'fruit' 변수에 저장*
print (fruit) *# 'fruit' 변수를 사용*
과일 = 3 *# 한글변수도 사용 가능. '과일' 변수에 저장*
숫자나 문장 등의 데이터를 영어, 한글 등으로 이름지어 저장하고 사용할 수 있습니다.

★ 산술연산식

a * 2 + b
변수, 숫자를 대상으로 사칙연산 외 다양한 연산을
할 수 있습니다. 연산식 결과는 변수에 저장할 수 있습니다.
*+ : 더하기, - : 뺄셈, * : 곱셈, / : 나눗셈, ** : 제곱*
// : 나눗셈에서 몫만 산출, % : 나눗셈에서 나머지만 산출

★ 문자열 연산

사용자이름="크레이"
문구="안녕하세요, "
print(문구 + 사용자이름) # '안녕하세요, 크레이' 출력
문자열끼리 연산을 합니다. 보통 문자열을 이어 붙일 때 덧셈 기호를 사용합니다.

★ 조건연산식

사용자연령<=30
특정 조건이 만족하는지 여부를 판단합니다.
주로 변수나 연산식으로 비교하며 제어문에서 사용됩니다.
왼쪽 == 오른쪽 : 왼쪽과 오른쪽이 동일한가?
왼쪽 != 오른쪽 : 왼쪽과 오른쪽이 다른가?
왼쪽> 오른쪽 : 왼쪽이 큰가?
왼쪽 < 오른쪽 : 왼쪽이 작은가?
왼쪽 >= 오른쪽 : 왼쪽이 크거나 같은가?,
왼쪽 <= 오른쪽 : 왼쪽이 작거나 같은가?

★ 복합조건연산식

사용자연령<=30 and 사용자키>=70
2가지 이상의 조건연산식이 만족하는지 여부를 판단합니다.
조건1 and 조건2 : 조건1,2가 모두 만족하는지 판단
조건1 or 조건2 : 조건1,2중 하나만이라도 만족하는지 판단

★ if 비교문

```
사용자연령=36
if(사용자연령<=30):
  print("30세 미만입니다")
else:
  print("30세 이상입니다") # 결과: "30세 이상입니다" 출력
```
조건에 따라 다른 처리를 할 때 사용합니다.

★ for 반복문

```
sum=0
for i in range(1, 11): # 1 ~ 10을 반복 (11 바로 전)
  sum += i
print(sum) # 1~10의 합계 55 출력
```
특정 횟수를 반복합니다.

★ while 반복문

```
sum=0
i=0
while sum<=10000: # 합계가 10000 이하인 경우 반복
  i = i + 1
  sum += I
# 합계가 10000 이 넘으면 반복을 마치고,
print(i)  # I값과 sum 값을 출력
print(sum) #
```
특정 조건이 만족할 때까지 반복합니다.

★ 함수

변수=함수명()
변수=함수명(변수/연산식)
변수=함수명(변수/연산식, 변수/연산식, ...)
특정기능을 갖는 함수를 사용합니다.
괄호 내 '변수/연산식'을 넘겨주는 것이 일반적이며, 함수유형에 따라 '변수/연산식'은 없을 수도, 1개나 2개 이상을 넘겨줘야 할 수도 있습니다.
함수는 연산결과값을 반환하며, 그 값을 변수에 담거나 바로 연산식에 사용할 수 있습니다.

❝파이썬3 기초 요약(2) – 데이터형❞

★ 정수형 (integer)

3, 45, -12 등의 값

소숫점이 붙지 않은 양수와 음수, 0을 정수라 부릅니다.
파이썬은 다른 언어와 달리 정수의 범위 제한이 없습니다.

```
a=12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345
b=a*2
print(a+b)
# 출력값
37037036703703703670370370367037037036703703703670370370367037037036703703703670370370367037037036703703703670370370367037037036703703703670370370367037037035
```

★ 실수형 (float)

3.14, 0.53, 4e-5 등의 값

소숫점이 붙은 숫자를 실수라고 합니다.
4e-5은 지수표현이라고 하여 4 x 10의 -3승을 의미하며 그 값은 0.00004입니다.

★ 논리형 (bool)

True, False

참 거짓의 결과값을 갖는 논리형입니다.

```
비가온다 = True
print(비가온다) # True
if(비가온다):  # 논리형 변수는 if문에서 사용가능
  print("우산을 챙기시오!") # True일 때 내부 실행
```

★ 문자열형(string)

"안녕하세요", 'Hello' 등

단어나 문장을 따옴표 또는 겹따옴표 기호로 묶어 표기합니다.

★ 허수형 (complex)

2 + 4j, 7 + 11j

복소수, x의 제곱 = -1과 같이 제곱한 값이 음수가 되는 수를 의미합니다. 복소수를 이해하려면 수학적 지식을 필요로 합니다.

```
x = 1j          #  허수
x2 = x ** 2   # x의 제곱을 구함, -1
print ( x2 )    # (-1+0j) 출력
print ( x2.real ) # -1.0 출력
```

★ 튜플 (tuple)

t = (값1, 값2, 값3, ...)

1개의 변수에 여러개의 값을 담습니다.
각 값들은 대괄호[순번]을 이용해 참조할 수 있으며
순번은 0부터 시작되며 첨자라고 합니다.
튜플은 한번 선언되면 그 값을 변경할 수 없습니다.

```
t[0], t[1]. t[2], ,...
튜플변수 = ('철수', 89, 79, 34)
print(튜플변수[0] + '의 점수 합계는 ' + str(튜플변수[1] + 튜플변수[2] + 튜플변수[3]) + '입니다.' )
# 결과 : 철수의 점수 합계는 202입니다.
튜플변수[0]='영수' # 오류 발생
```

★ 리스트 (list)

a = [값1, 값2, 값3, ...]

1개의 변수에 여러개의 값을 담습니다.
튜플과 동일한 구조이나 그 값을 변경할 수 있습니다.

```
리스트 = ['철수', 89, 79, 34]
print(리스트[0] + '의 점수 합계는 ' + str(리스트[1] + 리스트[2] + 리스트[3]) + '입니다.' )
# 결과 : 철수의 점수 합계는 202입니다.
리스트[0]='영수' # 첫번째 값을 변경
print(리스트[0] + '의 점수 합계는 ' + str(리스트[1] + 리스트[2] + 리스트[3]) + '입니다.' )
# 결과 : 영수의 점수 합계는 202입니다.
```

★ 딕셔너리 (dictionary)

딕셔너리 = {'이름':값, '이름':값, ...}

1개의 변수에 여러개의 값을 담되 이름을 지어서 담습니다.
리스트와 비슷하나 첨자 대신 이름을 사용합니다.

```
딕셔너리 = {
  '이름':'철수', '국어':90, '영어':75, '수학':100 }
print(딕셔너리['이름']) # 철수
print(딕셔너리['수학']) # 100
딕셔너리['컴퓨터']=95 # 새 항목 추가
print(딕셔너리) # {'이름': '철수', '국어': 90, '영어': 75, '수학': 100, '컴퓨터': 95}
```

★ 집합 (set)

집합변수 = set([값1, 값2, ...])

합집합, 교집합 등을 연산할 집합형태

```
집합변수1 = set([3,4,5,6])
집합변수2 = set([5,6,7,8])
print(집합변수1 & 집합변수2) # {5, 6} (교집합 결과)
```

❝파이썬3 기초 요약(3) - 리스트 가공❞

★ 리스트값 추출

변수 = 리스트[첨자]
리스트의 특정위치값을 추출합니다.
변수=리스트[4] # 리스트의 5번째 값을 꺼냅니다

★ 리스트값 변경

리스트[첨자]=값
리스트의 특정위치값의 내용을 변경합니다.
리스트[4]=3 # 리스트의 5번째 값을 3으로 변경

★ 리스트 뒤에 추가

리스트.append(변수/연산식)
리스트 맨 뒤에 값을 추가합니다.
리스트 = ['철수', 89, 79, 34]
리스트.append(55) # ['철수', 89, 79, 34, 55]

★ 리스트 중간에 추가

리스트.insert(첨자, 변수/연산식)
리스트 중간에 값을 삽입합니다.
리스트 = ['철수', 89, 79, 34]
리스트.insert(2, 100) # ['철수', 89, 100, 79, 34]

★ 리스트 값을 하나 받아오며 삭제

변수 = 리스트.pop()
리스트 맨 뒤의 값을 삭제하며 그 값을 받아옵니다.
리스트 = ['철수', 89, 79, 34, 89]
점수 = 리스트.pop() # 맨 뒤 89값을 받아오면서 삭제
print(점수) # 89 출력
print(리스트) # ['철수', 89, 79, 34]

★ 리스트 값 삭제

리스트.remove(변수/연산식)
리스트 값을 삭제. 동일값이 2개면 앞의값만 삭제됩니다.
리스트 = ['철수', 89, 79, 34, 89]
리스트.remove(89) # ['철수', 79, 34, 89]

★ 리스트 값 모두 삭제

리스트.clear()
리스트 내의 값을 모두 삭제합니다.
리스트 = ['철수', 89, 79, 34, 89]
리스트.clear() # []

★ 리스트 슬라이스

리스트[첨자1:첨자2]
리스트 첨자1~첨자2 전까지 추려 리스트로 반환합니다.
첨자2가 3이면 3번째까지 추립니다. 첨자1 생략시 맨 처음부터, 첨자2 생략시 맨 끝까지 반환
리스트1 = [1, 2, 3, 4, 5]
리스트2 = 리스트1[1:3] # [2, 3]
리스트3 = 리스트1[:4] # [1, 2, 3, 4]
리스트4 = 리스트1[3:] # [4, 5]
리스트5 = 리스트1[:] # [1, 2, 3, 4, 5]
리스트6 = 리스트1[-3:] # [1, 2, 3, 4, 5]
** -3: 끝에서 3번째부터*

★ 리스트 값을 슬라이스로 삭제

del 리스트[첨자], del 리스트[첨자:첨자]
리스트 중간값을 첨자나 슬라이스를 이용해 삭제합니다.
리스트 = ['철수', 89, 79, 34, 89]
del 리스트[2] # ['철수', 89, 34, 89]
리스트 = ['철수', 89, 79, 34, 89]
del 리스트[2:4] # ['철수', 89, 89]

★ 리스트 합침

2개 이상의 리스트를 나열하여 합칩니다.
리스트1 = [1, 2]
리스트2 = [3, 4]
print(리스트1+리스트2) # [1, 2, 3, 4]

★ 리스트 반복

리스트를 횟수만큼 반복합니다.
*리스트 = [1, 2] * 3 # [1, 2, 1, 2, 1, 2]*

★ 그 외 리스트 기능

리스트.count() : 리스트의 들어있는 값의 갯수 반환
[1, 1, 2, 3].count(1) # 2
리스트.sort() : 리스트를 숫자/ABC/가나다순으로 정렬합니다.
a = ['바나나', '파인애플', '사과']
a.sort() # ['바나나', '사과', '파인애플']
리스트.reverse() : 리스트를 뒤집습니다.
a = ['바나나', '파인애플', '사과']
a.reverse() # ['사과', '파인애플', '바나나']
max(리스트) : 리스트중 제일 큰 최대값 반환
min(리스트) : 리스트중 제일 작은 최소값 반환
리스트1.extend(리스트2) : 리스트 뒤에 이어 붙입니다.
리스트.index(값) : 리스트의 값이 들어있는 첨자 리턴