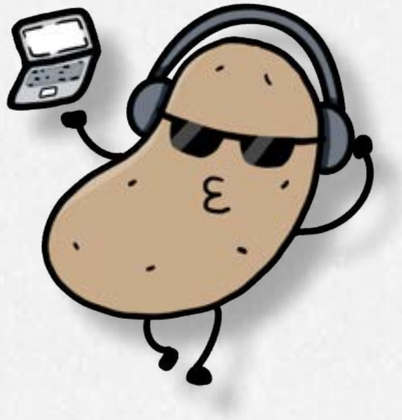
2023.06.15

코딩공감자

SELF INTRODUCTION



지난 시간 복습 Quiz

BIGDATA 특징

1 V

다양한 소스에서 방대한
데이터를 수집
- 1 tera~ 수십 peta

2 V

매우 빠른 속도로 데이터
가 유입되며 적시에 매우
빠르게 처리 되어야 한다

3 V

다양한 유형의 형식으로 제공
- 정형, 반정형, 비정형 데이터

4 V

회사 성장에 기여할 수
있는 사람일까?

지난 시간 복습 Quiz

BIGDATA 특징

1 VOLUME

다양한 소스에서 방대한
데이터를 수집
- 1 tera~ 수십 peta

2 VELOCITY

매우 빠른 속도로 데이터
가 유입되며 적시에 매우
빠르게 처리 되어야 한다

3 VARIETY

다양한 유형의 형식으로 제공
- 정형, 반정형, 비정형 데이터

4 VALUE/ VISUALIZATION

회사 성장에 기여할 수
있는 사람일까?

contents

CDG GAMZA

파이썬

- 기업의 실무에서 많이 사용되는 언어
- 구글에서 만든 소프트웨어의 50% 이상이 파이썬으로 작성됨.
  ex. 인스타그램, 드롭박스 등
- 공동 작업과 유지 보수가 매우 쉽고 편함.
- 파이썬을 사용해 프로그램을 개발하는 업체들이 늘어가는 추세

파이썬이란?

1. 파이썬은 인간다운 언어이다.

2. 파이썬은 문법이 쉬워 빠르게 배울 수 있다

3. 파이썬은 무료이지만 강력하다

4. 파이썬은 프로그래밍을 즐기게 해준다

5. 파이썬은 개발 속도가 빠르다

```
if 4 in [1,2,3,4]: print("4가 있습니다")
```

```
만약 4가 1, 2, 3, 4 중에 있으면 "4가 있습니다"를 출력한다.
```

파이썬으로 할 수 있는 일 VS. 할 수 없는 일

1. 시스템 유틸리티 제작

2. GUI 프로그래밍

3. C/C++와의 결합

4. 웹 프로그래밍

5. 수치 연산 프로그래밍

6. 데이터베이스 프로그래밍

7. 데이터 분석

8. 사물 인터넷

9. 인공지능과 머신러닝

1. 시스템과 밀접한 프로그래밍 영역

2. 모바일 프로그래밍

Colab

- Jupyter 기반으로 만들어진 웹용 서비스 by Google
- 특정 라이브러리를 별도로 설치할 필요없이 바로 적용하여 사용할 수 있기 때문에 편리함.
- 그래픽 카드(GPU)가 필요하지 않음.
- 협업하기 좋음.

자료형

프로그래밍을 할 때 쓰이는 숫자, 문자열 등 자료 형태로 사용하는 모든 것

1. 숫자형
2. 문자열 자료형
3. 리스트 자료형
4. 튜플 자료형
5. 집합 자료형
6. 불 자료형
7. 딕셔너리 자료형 Next time...

1. 숫자형

숫자 형태로 이루어진 자료형

항목	사용 예
정수	123, -345, 0
실수	123.45, -1234.5, 3.4e10
8진수	0o34, 0o25
16진수	0x2A, 0xFF

1. 숫자형

숫자 형태로 이루어진 자료형

정수형

```
>>> a = 123
>>> a = -178
>>> a = 0
```

= 은 '대입'을 의미

실수형

```
>>> a = 1.2
>>> a = -3.45
```

```
>>> a = 4.24E10
>>> a = 4.24e-10
```

E10(e10)은 10^10

e-10(E-10)은 10^(-10)

1. 숫자형

사칙연산

```
>>> a = 3
>>> b = 4
>>> a + b
7
>>> a - b
-1
>>> a * b
12
>>> a / b
0.75
```

덧셈 +

뺄셈 -

곱셈 *

나눗셈 /

```
>>> a = 3
>>> b = 4
>>> a ** b
81
```

a의 b제곱(승) **

```
>>> 7 % 3
1
>>> 3 % 7
3
```

나눗셈의 나머지 %

```
>>> 7 / 4
1.75
```

나눗셈 /

```
>>> 7 // 4
1
```

나눗셈의 몫 //

2. 문자열 자료형

문자, 단어 등으로 구성된 문자들의 집합

문자열 만들기

1. 큰 따옴표(")로 양쪽 둘러싸기 ex. "Hello World"

2. 작은 따옴표(')로 양쪽 둘러싸기 ex. 'Hello World'

3. 큰 따옴표 3개를 연속(""")으로 써서 양쪽 둘러싸기 ex. """Hello World"""

4. 작은 따옴표 3개를 연속(''')으로 써서 양쪽 둘러싸기 ex. '''Hello World'''

2. 문자열 자료형

문자열 안에 작은따옴표나 큰따옴표를 포함시키고 싶을 때

1. 문자열에 작은따옴표 (') 포함시키기

    ex. a = "Python's favorite food is perl"

2. 문자열에 큰따옴표 (") 포함시키기

    ex. b = '"Python is very easy." he says.'

2. 문자열 자료형

문자열 연산하기

문자열 더해서 연결하기

```
>>> head = "Python"
>>> tail = " is fun!"
>>> head + tail
'Python is fun!'
```

문자열 곱하기

```
>>> a = "python"
>>> a * 2
'pythonpython'
```

문자열 길이 구하기

```
>>> a = "Life is too short"
>>> len(a)
17
```

len(a) : a의 길이(length)

2. 문자열 자료형

문자열 슬라이싱

```
>>> a = "Life is too short, You need Python"
          0123456789 10
```

*파이썬은 숫자를 0부터 센다
*빈칸도 하나의 문자로 센다
*-1은 뒤에서부터 센다

```
>>> a = "Life is too short, You need Python"
>>> a[0]
'L'
>>> a[12]
's'
>>> a[-1]
'n'
```

2. 문자열 자료형

문자열 슬라이싱

```
>>> a = "Life is too short, You need Python"
>>> b = a[0] + a[1] + a[2] + a[3]
>>> b
'Life'
```

```
>>> a[0:2]
'Li'
>>> a[5:7]
'is'
>>> a[12:17]
'short'
```

문자열[시작번호:끝번호] : 시작번호부터 끝번호 전까지.
즉, 시작번호부터 (끝번호-1)번째까지

```
>>> a[19:]
'You need Python'
```

*시작번호를 생략하면 처음부터
*끝번호를 생략하면 끝까지 문자를 뽑아냄

2. 문자열 자료형

문자열 슬라이싱

```
>>> a = "20010331Rainy"
```

연도 (2001), 월과 일(0331), 날씨(Rainy)를 뽑아 각각 year, day, weather 변수에 저장해보기

2. 문자열 자료형

문자열 슬라이싱

```
>>> a = "20010331Rainy"
```

연도 (2001), 월과 일(0331), 날씨(Rainy)를 뽑아 각각 year, day, weather 변수에 저장해보기

```
>>> year = a[:4]
>>> day = a[4:8]
>>> weather = a[8:]
```

```
>>> year
'2001'
>>> day
'0331'
>>> weather
'Rainy'
```

2. 문자열 자료형

문자 개수 세기 (count)

```
>>> a = "hobby"
>>> a.count('b')
2
```

위치 알려주기 (find)

```
>>> a = "Python is the best choice"
>>> a.find('b')
14
>>> a.find('k')
-1
```

b가 처음으로 나온 위치 반환

찾는 문자나 문자열이 존재하지 않다면 -1을 반환

2. 문자열 자료형

위치 알려주기 (index)

```
>>> a = "Life is too short"
>>> a.index('t') t가 처음으로 나온 위치 반환
8
>>> a.index('k') 찾는 문자나 문자열이 존재하지 않다면 오류 발생
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ValueError: substring not found
```

문자열 삽입

```
>>> ",".join('abcd') 문자열의 각 문자 사이에 , 삽입
'a,b,c,d'
```

2. 문자열 자료형

소문자를 대문자로 바꾸기 (upper) / 대문자를 소문자로 바꾸기 (lower)

```
>>> a = "hi"
>>> a.upper()
'HI'
```

```
>>> a = "HI"
>>> a.lower()
'hi'
```

문자열 바꾸기 (replace)

```
>>> a = "Life is too short"
>>> a.replace("Life", "Your leg")
'Your leg is too short'
```

문자열 나누기 (split)

```
>>> a = "Life is too short"
>>> a.split()
['Life', 'is', 'too', 'short']
>>> b = "a:b:c:d"
>>> b.split(':')
['a', 'b', 'c', 'd']
```

3. 리스트 자료형

리스트명 = [요소1, 요소2, 요소3, ...]

```
>>> a = []    비어있는 리스트
>>> b = [1, 2, 3]
>>> c = ['Life', 'is', 'too', 'short']
>>> d = [1, 2, 'Life', 'is']
>>> e = [1, 2, ['Life', 'is']]
```

3. 리스트 자료형

리스트의 인덱싱

```
>>> a = [1, 2, 3]
>>> a
[1, 2, 3]
```

```
>>> a[0]
1
```

```
>>> a[-1]
3
```

```
>>> a[0] + a[2]
4
```

```
>>> a = [1, 2, 3, ['a', 'b', 'c']]
```

```
>>> a[0]
1
>>> a[-1]
['a', 'b', 'c']
>>> a[3]
['a', 'b', 'c']
```

3. 리스트 자료형

리스트의 슬라이싱

```
>>> a = [1, 2, 3, 4, 5]
>>> a[0:2]
[1, 2]
```

리스트 더하기

```
>>> a = [1, 2, 3]
>>> b = [4, 5, 6]
>>> a + b
[1, 2, 3, 4, 5, 6]
```

리스트 반복하기

```
>>> a = [1, 2, 3]
>>> a * 3
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```

리스트 길이 구하기

```
>>> a = [1, 2, 3]
>>> len(a)
3
```

3. 리스트 자료형

리스트에서 값 수정하기

```
>>> a = [1, 2, 3]
>>> a[2] = 4
>>> a
[1, 2, 4]
```

리스트 요소 삭제하기 (del)

```
>>> a = [1, 2, 3]
>>> del a[1]
>>> a
[1, 3]
```

리스트에 요소 추가(append)

```
>>> a = [1, 2, 3]
>>> a.append(4)
>>> a
[1, 2, 3, 4]
```

3. 리스트 자료형

리스트 정렬(sort)

```
>>> a = [1, 4, 3, 2]
>>> a.sort()
>>> a
[1, 2, 3, 4]
```

```
>>> a = ['a', 'c', 'b']
>>> a.sort()
>>> a
['a', 'b', 'c']
```

리스트 뒤집기(reverse)

```
>>> a = ['a', 'c', 'b']
>>> a.reverse()
>>> a
['b', 'c', 'a']
```

인덱스 반환(index)

```
>>> a = [1,2,3]
>>> a.index(3)
2
>>> a.index(1)
0
```

3. 리스트 자료형

리스트에 요소 삽입(insert)

```
>>> a = [1, 2, 3]
>>> a.insert(0, 4)
>>> a
[4, 1, 2, 3]
```

리스트 요소 제거(remove)

```
>>> a = [1, 2, 3, 1, 2, 3]
>>> a.remove(3)
>>> a
[1, 2, 1, 2, 3]
```

리스트 요소 끄집어내기(pop)

```
>>> a = [1,2,3]
>>> a.pop()
3
>>> a
[1, 2]
```

3. 리스트 자료형

리스트에 포함된 요소 x의 개수 세기(count)

```
>>> a = [1,2,3,1]
>>> a.count(1)
2
```

리스트 확장(extend)

```
>>> a = [1,2,3]
>>> a.extend([4,5])
>>> a
[1, 2, 3, 4, 5]
>>> b = [6, 7]
>>> a.extend(b)
>>> a
[1, 2, 3, 4, 5, 6, 7]
```

4. 튜플 자료형

- 리스트와 유사함.
- 리스트는 [ ]로 둘러싸지만 튜플은 ( )으로 둘러쌈.
- 리스트는 요소 값의 생성, 삭제, 수정이 가능하지만 튜플은 요소 값을 바꿀 수 없음.

```
>>> t1 = (1, 2, 'a', 'b')
>>> del t1[0]
```

요소를 삭제하려면 오류 발생 -> 변경불가

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object doesn't support item deletion
```

4. 튜플 자료형

인덱싱하기

```
>>> t1 = (1, 2, 'a', 'b')
>>> t1[0]
1
>>> t1[3]
'b'
```

슬라이싱하기

```
>>> t1 = (1, 2, 'a', 'b')
>>> t1[1:]
(2, 'a', 'b')
```

튜플 더하기

```
>>> t1 = (1, 2, 'a', 'b')
>>> t2 = (3, 4)
>>> t3 = t1 + t2
>>> t3
(1, 2, 'a', 'b', 3, 4)
```

튜플 곱하기

```
>>> t2 = (3, 4)
>>> t3 = t2 * 3
>>> t3
(3, 4, 3, 4, 3, 4)
```

튜플 길이 구하기

```
>>> t1 = (1, 2, 'a', 'b')
>>> len(t1)
4
```

5. 집합 자료형

- 중복을 허용하지 않음
- 순서가 없음 (Unordered)

```
>>> s1 = set([1,2,3])
>>> s1
{1, 2, 3}
```

```
>>> s2 = set("Hello")
>>> s2
{'e', 'H', 'l', 'o'}
```

5. 집합 자료형

```
>>> s1 = set([1,2,3,4,5,6])
>>> s2 = set([4,5,6,7,8,9])
```

교집합

```
>>> s1 & s2
{4,5,6}

>>> s1.intersection(s2)
{4,5,6}
```

합집합

```
>>> s1 | s2
{1,2,3,4,5,6,7,8,9}

>>> s1.union(s2)
{1,2,3,4,5,6,7,8,9}
```

차집합

```
>>> s1 - s2
{1,2,3}

>>> s1.difference(s2)
{1,2,3}
```

'&'는 'and'를 의미하고, '|'는 'or'을 의미한다

6. 불 자료형

참(True)과 거짓(False)을 나타내는 자료형

```
>>> 1 == 1
True
```

```
>>> 2 > 1
True
```

```
>>> 2 < 1
False
```

'='은 대입을 의미

'=='은 같은지 여부를 의미

1은 참, 0은 거짓을 의미

Quiz1

홍길동 씨의 주민등록번호는 881120-1068234이다. 홍길동 씨의 주민등록번호를 연도, 월, 일(YYYYMMDD)과 그 뒤의 숫자 부분으로 나누어 출력해 보자.

pin = "881120-1068234"

출력결과)

print (year)
>> 88

print (month)
>> 11

print (date)
>>20

print (num)
>> 1068234

Quiz1

홍길동 씨의 주민등록번호는 881120-1068234이다. 홍길동 씨의 주민등록번호를 연도, 월, 일(YYYYMMDD)과 그 뒤의 숫자 부분으로 나누어 출력해 보자.

pin = "881120-1068234"

출력결과)

print (year)
>> 88

year = pin[:2]

print (month)
>> 11

month = pin[2:4]

print (date)
>>20

date= pin[4:6]

print (num)
>> 1068234

num = pin[7:]

Quiz2

다음과 같은 문자열 a:b:c:d가 있다. a#b#c#d로 바꿔서 출력해 보자.
(힌트: replace 함수)

a = "a:b:c:d"

출력결과)

print (a)
>> 'a#b#c#d'

Quiz2

다음과 같은 문자열 a:b:c:d가 있다. a#b#c#d로 바꿔서 출력해 보자.
(힌트: replace 함수)

a = "a:b:c:d"

출력결과)

print (a)
>> 'a#b#c#d'

a = a.replace(":", "#")

Quiz3

[1, 3, 5, 4, 2] 리스트를 [5, 4, 3, 2, 1]로 만들어 보자.

(힌트: 리스트 내장 함수 이용)

a = [1,3,5,4,2]

출력결과)

print (a)
>> [5, 4, 3, 2, 1]

Quiz3

[1, 3, 5, 4, 2] 리스트를 [5, 4, 3, 2, 1]로 만들어 보자.

(힌트: 리스트 내장 함수 이용)

a = [1,3,5,4,2]

출력결과)

print (a)
>> [5, 4, 3, 2, 1]

a.sort()
a.reverse()

Q&A

코딩공감자

We wish you **success.**

(**당신의 성공을** 기원합니다.)

CDG GAMZA