



Python Tutoring #3

School of Computing, KAIST
& 대덕고등학교 빛나리



목차

- python 설치하기
- Review
- Sequence 자료형
 - String(문자열) 자료형
 - List 자료형
- 문제 풀이



Python 설치하기



Python 설치 여부 확인하기

1. (Windows 기준) 명령 프롬프트 켜기
2. `python --version`

명령창에 위와 같이 입력하면 아래와 비슷하게 출력됨

```
>>Python 3.7.3
```



Python 설치하기

- Python 설치하기 (3.7.? 설치를 권장)

<https://wikidocs.net/20339>

- IDLE 사용하기

<https://wikidocs.net/20341>

- Pycharm 설치하기 : 권장 (가상환경 구성이 쉽다.)

<https://wikidocs.net/20343>



설치 없이 Python 사용하기

- 구글 Colab 사용하기

<https://theorydb.github.io/dev/2019/08/23/dev-ml-colab/>

머신 러닝에 관심이 많은 학생들에게 추천

.py : Python 파일 형식

.ipynb : 쥬피터 노트북 형식



Review



Conditional Statement (조건문)

- if에 주어진 조건을 보고

특정 조건을 만족할 때만 실행되는 코드

- Indentation (들여 쓰기)
파이썬 vs. c 언어

```
score = int( input() )  
  
if score >= 60 :  
    print("Pass")  
else :  
    print("Fail")
```




2가지 이상의 상황에 대한 if문 작성하기

- if / else를 if / else 안에 추가할 것

```
score = input()

if score >= 60 :
    if score >= 90:
        print("Awesome")
    else:
        print("Pass")
else :
    print("Fail")
```



2가지 이상의 상황에 대한 if문 작성하기

- elif를 활용할 것

```
grade = input()

if score >= 90 :
    print("Awesome")
elif score >= 60:
    print("Pass")
else :
    print("Fail")
```

예제 1

- 이차방정식의 해가 몇 개인지 판별하는 프로그램을 짜보자.

Step 1 : A, B, C의 정수를 입력 받자

Step 2 : A가 0이면 "2차방정식이 아니다."라고 print한다.

Step 3 : 판별식의 값에 따라 해가 몇 개인지 판별하자.

힌트: 판별식은 $D = B * B - 4 * A * C$ 와 같다.



For Loop

- 매번 Loop를 돌 때마다
i의 값이 0 -> n-1까지 순차적으로 변화
- : (콜론)과 들여쓰기에 유의할 것

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

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



While Loop

- while + 조건문 + : (콜론)
- : (콜론)과 들여쓰기에 유의할 것
- Loop를 빠져나갈 수 있는지 확인 할 것

```
i = 0
while i < 10:
    print(i)
    i = i + 1
```



예제 2

- 구구단 전체를 출력하는 프로그램을 작성하라.

힌트 :

2개의 for Loop를 활용.

print 함수 내에서 ','로 이어주면 띄어쓰기를 하고 이어짐.

e.g.

```
print(3, 'x', 4, '=', 12)
```

```
>> 3 x 4 = 12
```



지난 시간 예제 풀이

- <https://www.acmicpc.net/problem/8958>

"OOXXOXXOOO"와 같은 OX퀴즈의 결과가 있다.

O는 문제를 맞은 것이고, X는 문제를 틀린 것이다.

문제를 맞은 경우 그 문제의 점수는 그 문제까지

연속된 O의 개수가 된다. 예를 들어, 10번 문제의 점수는 3이 된다.

"OOXXOXXOOO"의 점수는 $1+2+0+0+1+0+0+1+2+3 = 10$ 점이다.

OX퀴즈의 결과가 주어졌을 때, 점수를 구하는 프로그램을 작성하라.



```
T = int(input())
for i in range(T):
    result = input()
    point = 0
    count = 0
    for s in result:
        if(s == 'O'):
            count+=1
        else:
            count=0
    point+=count
print(point)
```




Sequence 자료형



시퀀스(Sequence) 자료형

- 시퀀스형

변수에 index(순서)로 접근이 가능한 자료형

e.g.

String, List, Tuple 등등



Index(순서)

e.g.

string1 = "ABCDEF"

A	B	C	D	E	F
0	1	2	3	4	5

Index(순서)로 접근할 수 있다. (Index는 0부터 시작)

Indexing 예시 : `print(string1[1])` >> B



Slicing(자르기)

e.g.

```
string1 = "ABCDEF"
```

A	B	C	D	E	F
0	1	2	3	4	5

Slicing(자르기)를 하면 원래 자료형의 일부가 나온다.

Slicing 예시 : `print(string1[1:3])` >> BC (index 1부터 3이전까지)



Slicing(자르기)

e.g.

```
string1 = "ABCDEF"
```

```
print(type(string1)) >> <class 'str'>
```

```
print(type(string1[1:2])) >> <class 'str'>
```

```
list1 = [1, 2, 3, 4, 5]
```

```
print(type(list1)) >> <class 'list'>
```

```
print(type(list1[1:2])) >> <class 'list'>
```



Slicing(자르기)

sequence[n, m]에서

n이 생략된 경우 : 처음부터 시작

m이 생략된 경우 : 끝까지

e.g.

```
string1 = "ABCDEF"
```

```
string1[:] >> "ABCDEF"
```

```
string1[1:4] >> "BCD"
```

시퀀스형 자료에 대한 Loop Control

- 형식 :

```
for s in sequence1:  
    명령문1
```

```
for i in range(len(sequence1)):  
    명령문2
```

```
string1 = "I LOVE PYTHON"
```

```
for s in string1:  
    print(s)
```

```
for i in range(len(string1)):  
    print(string1[i])
```



String 자료형

참고 : <https://wikidocs.net/13>



String(문자열) 자료형

- String이란?

문자들로 구성된 문자열, Sequence 자료형에 해당

"" , ""와 같이 따옴표로 둘러싸여 있다면 String

string들의 연산은 숫자와 다르게 정의



String의 연산

연산	결과	예시
string1 + string2	두 개의 문자열을 이어준 것	"A" + "B" >> "AB"
string * n	하나의 문자열을 n번 반복한 것	"A" * 3 >> "AAA"
string1 == string2	두 개의 문자열이 같은가	"A" == "A" >> True
len(string)	문자열의 길이 반환	len("ABC") >> 3



String Indexing

A	B	C	D	E	F
0	1	2	3	4	5

- `string[n]`은 $n+1$ 번째 문자를 의미
(`string[0]`이 1 번째 문자인 것을 고려하자.)



String Slicing

A	B	C	D	E	F
0	1	2	3	4	5

- `string[n : m]`은 n 번째부터 $m-1$ 번째까지의 문자열을 의미

`string[0:3] >> "ABC "`



Review: Loop Control

- 형식 :

```
for s in sequence1:  
    명령문1
```

```
for i in range(len(sequence1)):  
    명령문2
```

```
string1 = "I LOVE PYTHON"
```

```
for s in string1:  
    print(s)
```

```
for i in range(len(string1)):  
    print(string1[i])
```



예제 3

문자열을 입력 받아 "A"의 개수를 세는 프로그램을 작성하라

Step1: input함수를 활용하여 문자열을 하나 입력 받자.

Step2: for 문과 if 문을 활용하여 "A"의 개수를 세자.



String 함수

함수	설명	예시
<code>string1.count(string2)</code>	string1 안의 string2의 개수를 센다.	<code>"AA".count("A") >> 2</code>
<code>string1.find(string2)</code>	string1에서 string2가 처음 등장하는 인덱스를 찾는다.	<code>"ABCAB".find("B") >> 1</code>

예제 4

예제 3을 for문 없이 다시 풀어보자.



List 자료형

참고 : <https://wikidocs.net/14>



List 자료형

- List는 여러 개의 값을 순서대로 저장하는 자료형
- [] (대괄호)로 감싸 표현한다.
- List의 원소는 무엇이든지 가능

e.g.

[1, 2, "String", [0, 1]]

List의 연산

연산	결과	예시
<code>list1 + list2</code>	두 개의 list을 이어준 것	<code>[1, 2] + [3] >> [1, 2, 3]</code>
<code>list1 * n</code>	하나의 list을 n번 반복한 것	<code>[1] * 3 >> [1, 1, 1]</code>
<code>list1 == list2</code>	두 개의 list의 각각의 원소가 같은가	<code>[1, 2] == [1, 2] >> True</code>
<code>len(list1)</code>	list의 길이 반환	<code>len([1, 2, 3]) >> 3</code>



List Indexing

list1	3	2	5	"AB"	7	[0]
	0	1	2	3	4	5

- list1[n]은 n+1 번째 원소를 의미
(list1[0]이 1 번째인 것을 고려하자.)

List Indexing

list1	3	2	5	"AB"	7	[0]
	0	1	2	3	4	5

Index로 접근하여 값 바꾸기

> list1 [1] = "New"

3	"New"	5	"AB"	7	[0]
0	1	2	3	4	5

List Slicing

list1	3	2	5	"AB"	7	[0]
	0	1	2	3	4	5

- list1[n : m]은 n번째부터 m-1번째까지의 새로운 list를 의미

list1[0:3] >> [3, 2, 5]



Review: Loop Control

- 형식 :

```
for s in list1:  
    명령문1
```

```
for i in range(len(list1)):  
    명령문2
```

```
list1 = [1, 2, 3]
```

```
for e in list1:  
    print(e)
```

```
for i in range(len(list1)):  
    print(list1[i])
```

List 함수

함수	설명	예시
<code>element in list1</code>	list1에 element가 있는지 확인	<code>1 in [1, 2, 3] >> True</code>
<code>list1.index(element)</code>	list1에서 element가 처음 등장하는 인덱스를 찾는다.	<code>[1, 3, 1, 4].index(1)>> 0</code>
<code>list1.append(element)</code>	list1에 원소를 추가한다.	<code>list1 = [1, 2]</code> <code>list1.append(3) >> [1, 2, 3]</code>
<code>list1.remove(element)</code>	list1에 처음 등장하는 element를 삭제한다. 유의점 : element가 list1에 없다면 에러가 발생할 수 있다.	<code>list1 = ["B", "C", "B"]</code> <code>list1.remove("B") >> ["C", "B"]</code>



예제 5

- 1) Input 함수를 이용하여 숫자 n 을 입력 받는다.
- 2) n 개의 문자열을 input으로 입력 받아 list에 추가한다.
- 3) 해당 list 안에 같은 문자열이 있는지 검사한다.



더 나아가기 : List 만들기

- List를 for 문과 비슷하게 만들 수 있다.

```
> [ n*n for n in range(4) ]
```

```
>> [ 0, 1, 4, 9 ]
```



더 나아가기 : Dynamic Programming

동적계획법 :

<https://ko.wikipedia.org/wiki/%EB%8F%99%EC%A0%81%EA%B3%84%ED%9A%8D%EB%B2%95>



Dynamic Programming

- 복잡한 경우의 문제를 보다 단순한 경우의 문제로 나누어서 해결하는 방법
- 추가적인 공간을 사용하여 간단한 문제들에 대한 풀이를 저장하여 활용할 수 있다.



피보나치 수

$$A_0 = 1, A_1 = 1$$

$$A_n = A_{n-1} + A_{n-2}$$

$$1 \rightarrow 1 \rightarrow 2 = 1+1 \rightarrow 3 = 1+2 \rightarrow 5 = 2 + 3$$

$$\begin{aligned} A_n &= A_{n-1} + A_{n-2} \\ &= (A_{n-2} + A_{n-3}) + A_{n-2} \\ &= \dots \end{aligned}$$



피보나치 수

- N 번째 피보나치 수를 구하는 방법?

```
N = int(input())
numbers = [0] * (N+2) # N개의 수를 저장할 공간 + 여유공간
numbers[0] = 1
numbers[1] = 1
for i in range(2, len(numbers)):
    numbers[i] = numbers[i-1] + numbers[i-2]
print(numbers[N-1])
```



추가 예제

- 가장 긴 증가하는 부분 수열
<https://www.acmicpc.net/problem/11053>
- 연속합
<https://www.acmicpc.net/problem/1912>
- 1, 2, 3 더하기
<https://www.acmicpc.net/problem/9095>