

공학적 문제해결 기법

6주차 2차시

여러 개의 조건걸기





6주차 2차시 : 여러 개의 조건걸기

I. 개요

여러 개의 조건걸기와 문자열과 숫자

II. 학습 개요

1) 학습 목표

여러 개의 조건을 걸어서 작업할 수 있는 방법과 int와 str, float를 사용하여 문자열을 숫자로 숫자를 문자열로 변경하는 방법을 알아보고, 그 외의 수치 연산 함수들에 대해 학습한다.

2) 학습 목차(세부 목차)

- F문과 E,F문
- 숫자와 문자열 변경
- 수치 연산 함수



6주차 2차시 : 여러 개의 조건걸기

III. 학습 1 - 학습 1 - IF 문과 ELIF 문

elif(else-if의 약자)문

- ☒ if문은 elif(else-if의 약자) 문으로 확장할 수 있다.
- ☒ 예를 들어 어떤 사람이 10살인지 11살인지 또는 12살인지를 검사할 수 있으며
- ☒ 각각에 따라서 다른 작업을 하는 프로그램을 만들 수 있다.
- ☒ if-then-else문과 다르며, 하나 이상의 elif문을 둘 수 있다.
- ☒ 다음의 예를 살펴봅시다.
- ☒ 다음 코드를 IDLE에 입력하면 들여쓰기가 자동으로 됨
- ☒ print문의 입력이 끝나고 if문과 elif문, else문을 입력할 때
- ☒ 백스페이스(Back Space)나 DELETE키를 눌러 가장 왼쪽 끝에서 입력을 시작함



6주차 2차시 : 여러 개의 조건걸기

III. 학습 1 - IF 문과 elif 문

elif (else-if의 약자) 문

```
>>> age = 12
>>> if age == 10:
    print("What do you call an unhappy cranberry?")
    print("A blueberry!")
elif age == 11:
    print("What did the green grape say to the blue grape?")
    print("Breathe! Breathe!")
elif age == 12:
    print("What did 0 say to 8?")
    print("Hi guys!")
else:
    print("Huh?")
```

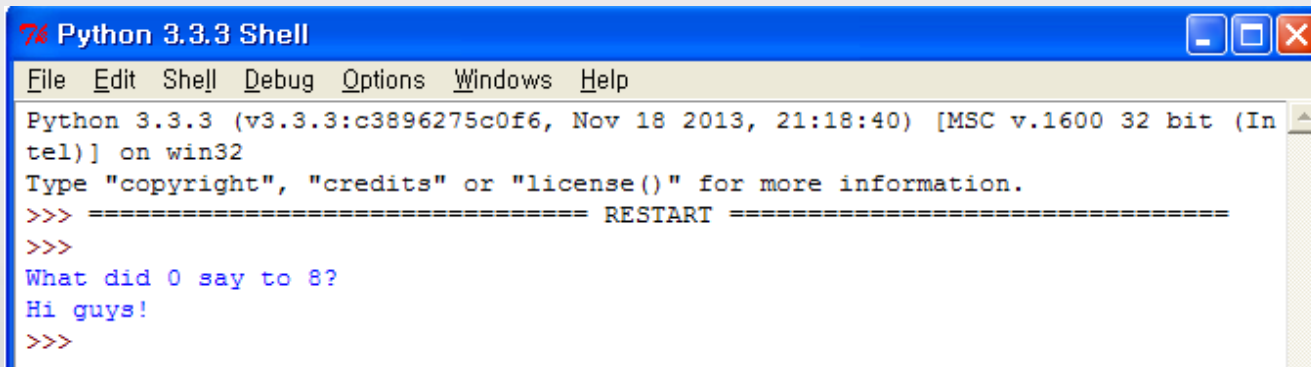


6주차 2차시 : 여러 개의 조건걸기

III. 학습 1 - IF 문과 ELIF 문

elif (else-if의 약자) 문

- ☒ 코드가 길어지는 앞서 배웠던 File > New File 을 선택하여 쉘 창에서 코드를 입력
- ☒ File > Save As 를 선택하여 적당한 파일명 (elif_statements.py) 으로 저장
- ☒ Run > Run Module 을 클릭하여 실행



```
Python 3.3.3 Shell
File Edit Shell Debug Options Windows Help
Python 3.3.3 (v3.3.3:c3896275c0f6, Nov 18 2013, 21:18:40) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
What did 0 say to 8?
Hi guys!
>>>
```



6주차 2차시 : 여러 개의 조건걸기

III. 학습 1 - IF문과 ELIF문

 실행결과가 나올 수 있도록 완성시켜 보세요

```
→ order = 'spagetti'
   if order == 'spam' :
       price = 500
   elif order == 'ham' :
       price = 700
   elif order == 'egg' :
       price = 300
   elif order == 'spagetti' :
       price = 900
```




6주차 2차시 : 여러 개의 조건걸기

III. 학습 1 - IF 문과 ELIF 문

조건문 조합하기

- ☒ 코드를 더 짧고 간단하게 해주는 and와 or 키워드를 사용



age = 12

```
if age == 10 or age == 11 or age == 12 or age == 13:
```

```
    print('What is 13 + 49 + 84 + 155 + 97? A headache!')
```

```
else:
```

```
    print('Huh?')
```

- ☒ 이 코드를 더 줄여 볼까요?



```
if age >= 10 and age <= 13:
```

```
    print('What is 13 + 49 + 84 + 155 + 97? A headache!')
```

```
else:
```

```
    print('Huh?')
```



6주차 2차시 : 여러 개의 조건걸기

III. 학습 1 - IF 문과 ELIF 문

아무런 값이 없는 변수 - NONE

- ✓ 변수에 아무것도 할당하지 않거나 빈 값을 할당할 경우
- ✓ 빈 값은 None이라고 하며, 값이 없는 것
- ✓ None이라는 것은 0과 다름 · 0도 아닌 값이 없는 것을 의미
- ✓ None이라는 값을 변수에 할당하는 것은 변수 초기의 빈 상태로 리셋
- ✓ 프로그램에서 나중에 변수가 필요할 예정이지만 초기에 변수를 설정하고 싶을 때



```
>>> myval = None
```

```
>>> if myval == None :
```

```
    print("The variable myval doesn't have a value")
```




6주차 2차시 : 여러 개의 조건걸기

III. 학습 2 - 숫자와 문자열 변경

문자열을 숫자로 또는 숫자를 문자열로

- ✓ 사용자입력(User input)은 사람이키보드로 입력하는것
- ✓ 입력되는것은문자나방향키, ENTER키,그외의다른것들도가능
- ✓ 파이썬에서사용자입력은문자열로간주
- ✓ 키보드로 100이라는숫자를 입력할 경우 숫자가아닌 문자열 변수에저장됨
- ✓ 변수age에문자열 '10'을 설정



```
>>> age = '10'
```

```
>>> if age == 10:
```

```
    print("What's the best way to speak to a monster? ")
```

```
    print("From as far away as possible!")
```

- ✓ print문이실행되지않음
- ✓ 문자열을숫자로 변환해야함



6주차 2차시 : 여러 개의 조건걸기

III. 학습 2 - 숫자와 문자열 변경

문자열을 숫자로 또는 숫자를 문자열로

- ☑ 문자열 '10'을 숫자로 변환

➡

```
>>> age = '10'
>>> converted_age = int(age)
```

- ☑ 숫자를 문자열로 변환

➡

```
>>> age = 10
>>> converted_age = str(age)
```

- ☑ 앞의 예제에서 int 함수를 사용하여 결과가 출력될 수 있도록 코드를 수정

➡

```
>>> age = '10'
>>> converted_age = int(age)
>>> if converted_age == 10:
    print("What's the best way to speak to a monster?")
    print("From as far away as possible!")
```



6주차 2차시 : 여러 개의 조건걸기

III. 학습 2 - 숫자와 문자열 변경

소수점이 있는 숫자를 변환

☒ int 함수는 정수형만 받음

➡

```
>>> age = '10.5'
>>> converted_age = int(age)
```

☒ 위 예는 ValueError로 사용하려고 하는 값이 올바르지 않다는 에러를 냄

☒ int 대신 float 함수를 사용

☒ float 함수는 정수가 아닌 실수들을 처리

☒ 다음 예는 어떤 값이 나올까요?

➡

```
>>> age = '10'
>>> if age == 10:
    print("What's the best way to speak to a monster?")
    print("From as far away as possible!")
```



6주차 2차시 : 여러 개의 조건걸기

III. 학습 3 - 수치 연산 함수

연산(operation)

- ☒ 덧셈과 뺄셈, 새로운 값의 대입 그리고 크기 비교 등의 행위를 통하여 값을 변경하거나 새로운 값을 산출하는 과정
- ☒ 연산자(operator) : 연산 과정을 표현하려고 사용하는 기호
- ☒ 관계 연산자: 크기 비교
- ☒ 논리 연산자: and, or, not 연산자가 있음
- ☒ 산술 연산: 덧셈, 뺄셈, 곱셈, 나눗셈, 지수승 및 나머지 연산자



6주차 2차시 : 여러 개의 조건걸기

III. 학습 3 - 수치 연산 함수

산술 연산자

➡ `>>> 2 ** 3` # $2 * 2 * 2$ 와 같다.

8

`>>> 2 ** 3 ** 2` # $2 ** (3 ** 2)$ 와 같다.

512

`>>> (2 ** 3) ** 2` # $2 * 2 * 2$ 와 같다.

64

`>>> 5 % 2` # 5를 2로 나눈 나머지

1

`>>> 3 + 5.1` # 정수 + 실수의 결과는 실수

8.1

`>>> 5/3` # 몫

1.6666666666666667



6주차 1차시 : 조건걸기

III. 학습 3 - 수치 연산 함수

내장된 수치 연산자

 int, str 및 float 함수처럼 내장된 수치 연산 함수들이 있음

함	의
abs(x)	x의 절대값
int(x)	x를 int로 변환
float(x)	x를 float으로 변환
divmod(x, y)	(x/y, x%y) 쌍
pow(x, y)	x의 y승
complex(re, im)	실수부 re와 허수부 im를 갖는 복소수



6주차 2차시 : 여러 개의 조건걸기

III. 학습 3 - 수치 연산 함수

내장된 수치 연산자

수치내장함수의사용예

➡ `>>> abs(-3)` # 양수의 값을 돌려준다.
3
`>>> int(-3.14)`
-3
`>>> divmod(5, 3)` # 5를 3으로 나눈 몫과 나머지
(1, 2)
`>>> pow(2, 3)` # 2 ** 3과 같다.
8
`>>> complex(3, 5)` # 실수부 3과 허수부 5를 갖는 복소수
(3+5j)



『 이 콘텐츠는 2014학년도 학부교육 선도대학 육성사업(ACE)에 의하여 개발한 것임 』

