PYTHON 프로그래밍 실습

파이썬 기초
(조건문, 반복문, 리스트 II)

2019-12-23

조건문

bool 자료형

- bool 자료형
  - bool은 **True**와 **False**의 두 가지 값을 나타내는 자료

```
>>> result = 3 > 2
>>> print(result)
True
>>> type(result)
<type 'bool'>
```

- 조건문이 다음과 같을 때 거짓(False)으로 평가
  - False, None, 숫자 0 예) 0, 0.0
  - 비어있는 순서열 예) '', (), [] 등
  - 비어있는 딕셔너리 : 예) {}

- 어떤 객체가 참인지 거짓인지 알고 싶을 때 bool() 함수 이용

```
>>> bool(0.0)
False
>>> bool([1,2,3])
True
```

관계 연산자와 논리 연산자

- 연산의 결과는 항상 참(True) 또는 거짓(False)
- **관계 연산자**
  - **<, >, ==, <=, >=, !=**
    - **'=='와 '='는 서로 다른 연산자**
- **논리 연산자**
  - **not, and, or**

```
>>> a = 99
>>> (a > 100) and (a < 200)
False
>>> (a > 100) or (a < 200)
True
>>> not(a == 100)
True
```

if 문

- 조건문의 결과가 참(True)이면 A를 수행

```
if 조건문 :
    수행할 문장1
    수행할 문장2
    ...
    수행할 문장n
```

A

- 블록(Block)
  - 여러 코드가 이루는 일정한 구역
  - **파이썬의 경우, 들여쓰기로 구역을 나눔**
  - 들여쓰기는 스페이스(Space)나 탭(Tab) 둘 다 사용 가능

    - 스페이스 4 칸 사용 권장

if-else

- 조건의 결과가 참(True)이면 if문 바로 다음의 문장(if 블록)들을 수행하고, 거짓(False)이면 else문 다음의 문장(else 블록)들을 수행

```
if 조건문 :
    수행할 문장1
    수행할 문장2
        …
else :
    수행할 문장A
    수행할 문장B
        …
```

```
a = 200

if a < 100 :
    print("참")
    print("100보다 작다")
else :
    print("거짓")
    print("100보다 크거나 같다")

print("프로그램 끝")
```

if-elif-else

- 다중 조건 판단

```
if 조건A :
    수행할 문장 A
elif 조건B :
    수행할 문장 B
...

else :
    수행할 문장 C
```

```
num = int(input("정수를 입력하시오: ")

if num > 0 :
    print("양수입니다.")
elif num == 0 :
    print("0입니다.")
else :
    print("음수입니다.")
```

중첩 if

```
if 조건A :
    if 조건1 :
        수행할 문장 A1
    else :
        수행할 문장 A2
else :
    수행할 문장 B
```

if 문 응용

- 리스트와 함께 사용

if 항목 in 리스트 :

리스트 안에 항목이 있으면 True를 반환

```
>>> 1 in [1, 2, 3]
True
```

if 항목 not in 리스트 :

리스트 안에 항목이 없으면 True를 반환

```
>>> 1 not in [1, 2, 3]
False
```

조건문 - 실습1

- 학생수준평가 시험에서 영어 점수와 수학 점수가 합해서 110점 이상이면 합격이다. 단, 각 점수가 40점 미만이면 불합격이다. 영어(eng), 수학(math)점수를 입력 받아 합격여부를 출력하는 프로그램을 작성하시오.

```
영어 점수 입력: 80
수학 점수 입력: 20
불합격: 총합 점수 부족
```

```
영어 점수 입력: 90
수학 점수 입력: 30
불합격: 수학 점수 부족
```

```
영어 점수 입력: 70
수학 점수 입력: 80
합격
```

```
영어 점수 입력: 35
수학 점수 입력: 30
불합격: 총합 점수 부족
```

```
영어 점수 입력: 35
수학 점수 입력: 95
불합격: 영어 점수 부족
```

조건문 - 실습2

- 세 개의 정수를 입력 받아, 가장 큰 수만 출력하는 프로그램을 작성하시오.

```
세 개의 수를 입력하시오: 30 22 50
가장 큰 수는 50입니다.
```

반복문

while문

- 조건이 참(**True**)인 동안 **A**를 반복 수행

```
while 조건식 :
    반복할 문장들 A
```

- 예제) 1부터 5까지 반복해서 숫자를 출력하는 프로그램

```
number = 1
while number <= 5 :
    print (number)
    number = number + 1
```

실행결과)

```
1
2
3
4
5
```

- 실행결과를 한 줄에 이어서 출력하려면?

```
print number,
```

Python 2.x 버전

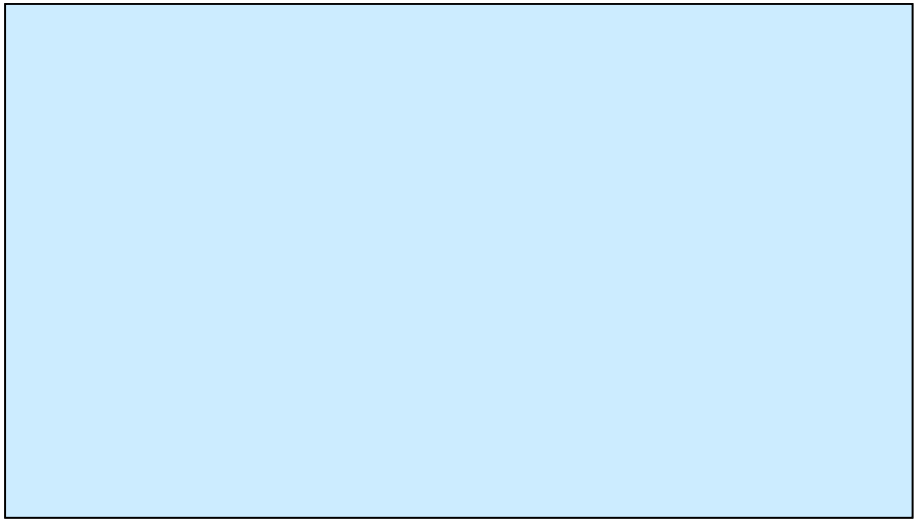
```
print (number, end=' ')
```

Python 3.x 버전

while문

- 1부터 10까지의 합(while 문 이용해서 작성)

1부터 10까지의 합: 55



range() 함수

- 형식

range(start, stop, step)

※ Python 2.x 버전

start부터 step만큼씩 (증가/감소)하여 stop전까지의 모든 정수의 값을 리스트 형태로 반환

```
>>> range(1, 7, 2)
[1, 3, 5]
```

- start와 step은 생략 될 수 있음
  - 생략된 경우 start의 값은 0, step은 1이라고 간주

```
>>> range(5)
[0, 1, 2, 3, 4]
>>> range(5, 10)
[5, 6, 7, 8, 9]
```

range(stop)

range(start, stop)

for문

- 순서열을 순회하다가 순서열의 끝에 도달하면 반복을 종료 함

```
for 반복변수 in 순서열 :
    반복할 문장
```

- 실습

```
for x in range(5) :
    print ("Hello World!")
```

```
for x in range(5) :
    print (x, end=' ')
```

```
for x in "python" :
    print (x, end=' ')
```

반복문 - 실습1(구구단 출력)

- 입력한 숫자의 구구단을 출력하는 프로그램 작성하기
- 실행예제 (밑줄은 사용자 입력을 말함)

```
출력하고 싶은 단을 입력하세요: 7
7 * 1 =  7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 6 = 42
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
```

중첩 반복문

- 반복문 내부의 반복문

```
while 조건1 :
    반복할 문장1
    반복할 문장2
    while 조건2 :
        반복할 문장A
        반복할 문장B
    반복할 문장3
```

반복문 - 실습2

- ★출력 프로그램
- 실행화면

```
Input an integer: 3159
★★★
★
★★★★★
★★★★★★★★★
```

break, continue

- break
  - 반복문 순회 도중 break 문을 만나면 내부 블록 벗어남

```
for i in range(1, 11) :
    if i > 5:
        break
    print (i)
```

```
1
2
3
4
5
```

- continue
  - 반복문 순회 도중 continue 문을 만나면 그 아래의 문장은 해당 반복에 한 해서 건너뛴다.

```
for i in range(1, 11) :
    if i % 2 == 0 :
        continue
    print (i)
```

```
1
3
5
7
9
```

리스트 II

리스트 관련 함수

- **생성**

```
>>> colors = list()  # 또는 colors = []
```

- **리스트이름.append(x)**

  : 기존 리스트 끝에 x 삽입

```
>>> colors
[]
>>> colors.append("red")
>>> colors.append("green")
>>> colors.append("blue")
>>> print (colors)
['red', 'green', 'blue']
```

- **리스트이름.remove(x)**

: 리스트에서 첫 번째로 나오는 x를 제거

```
>>> colors.append("red") # 항목 추가
>>> print (colors)
['red', 'green', 'blue', 'red']
>>> colors.remove('red')
>>> print (colors)
???
```

리스트 관련 함수

- **리스트이름.sort()**

: 리스트의 항목을 오름차순으로 정렬

```
>>> a = [1, 7, 9, 2, 5]
>>> a.sort()
>>> a
[1, 2, 5, 7, 9]
>>> f = ['banana', 'apple', 'orange']
>>> f.sort()
>>> f
['apple', 'banana', 'orange']

>>> a.sort(reverse=True) # 내림차순
```

- **sorted(리스트이름)**
  - sort()는 리턴값이 없고,
    sorted()함수는 리턴값이 있다.

```
>>> f = ['banana', 'apple', 'orange']
>>> f1 = f.sort()
>>> print (f1)
None
>>> f = ['banana', 'apple', 'orange']
>>> f2 = sorted(f)
>>> print (f2)
['apple', 'banana', 'orange']
```

리스트 관련 함수

- **리스트이름.reverse()**

: 리스트의 항목의 순서를 역순으로 뒤집어 줌

```
>>> a = [1, 7, 9, 2, 5]
>>> a.reverse()
>>> a
[5, 2, 9, 7, 1]
```

- **리스트의 최대값, 최소값, 총합**

```
>>> a = [45, 87, 99, 21, 55]
>>> max(a)
99
>>> min(a)
21
>> sum(a)
307
```

리스트 - 실습1

- "done"을 입력할 때까지 사용자로부터 숫자를 입력 받아 리스트에 저장하고, "done"을 입력하면, 리스트의 평균, 최대값과 최소값을 출력하는 프로그램을 작성하시오. (힌트. sum(), max(), min() 함수를 사용)
- 실행예시(밑줄은 사용자 입력)

```
Enter a number: 7
Enter a number: 2
Enter a number: 9
Enter a number: 3
Enter a number: 5
Enter a number: done

[7.0, 2.0, 9.0, 3.0, 5.0]
Average: 5.2
Maximum: 9.0
Minimum: 2.0
```