스터디 복습 자료

5월 7일 배운 내용 복습 (슬라이드 #1, #2)

변수란? 값을 담는 상자 (문자 또는 숫자)
변수_이름 = 변수_값 (순서 바뀌면 안됨)
여기서의 등호는 '같다'가 아닌 '할당'의 의미

Shell에서 사용하는 기능

- python # python 자체 (REPL) 실행
- python test.py # python으로 test.py를 실행, test.py는 NOTEPAD++에서 작성

python 파일은 .py 형식으로 저장해야 함
파일명이나 폴더명에 띄어쓰기 없이 영어로 작성하는게 편리함

주석 달 때는 한 줄 (#), 여러 줄 (‘’’) 사용

python에서 사용하는 다양한 사칙연산 기호 (+, -, *, /, **, %)
python에서 사용하는 다양한 조건식 기호 (>, <, >=, <=, ==, !=, and, or)
True, False 자체를 사용한 조건식

AND	True	False
True	True	False
False	False	False

OR	True	False
True	True	True
False	True	False

NOT	
True	False
False	True

if문의 기능

- 특정 조건이 참일 때, 블록 내의 코드를 실행
- 블록 바깥의 코드는 if문이 참이든 거짓이든 상관 없음

if문만으로 구성해도 상관없으나, 필요할 경우 elif 또는 else를 넣을 수 있음

if-elif-else의 관계

- if문의 조건이 참이면, if 블록 내의 코드를 실행
- if문의 조건이 거짓이고 elif문의 조건이 참이면, elif 블록 내의 코드를 실행
- (elif는 여러 번 넣을 수 있음)
- if문의 조건이 거짓이고 (만약 있다면) elif문의 조건도 거짓이면, else 블록 내의 코드를 실행

스터디 복습 자료

5월 14일 배운 내용 복습 (슬라이드 #3, #4)

함수란? 특정 기능을 수행하기 위해 만들어진 코드 조각
중요! 앞으로 자주 함수를 만들거나, python의 기본 함수를 사용하게 될 것

```
def function(parameter1, parameter2): # function은 함수의 이름, parameter는 매개변수
    …
    return True # return 기능은 함수의 값을 반환
val = function(a,b) # a, b를 매개변수로 받아 함수의 일을 하고 val에 함수의 값(True)을 돌려줌
```

매개변수(parameter): 함수의 입력 값.
매개변수가 필요한 이유?
함수가 그때그때 다른 입력을 받아서 내부적으로 처리할 수 있음

반환(return): 함수의 (출력) 값. 반환과 동시에 함수 종료
반환이 필요한 이유?
함수의 값을 다시 변수로 받아서 사용할 수 있음

format 함수
입력: 빈 자리에 들어갈 문자열 혹은 숫자
출력(반환): 빈 자리에 입력값들이 순서대로 채워진 문자열

변수의 종류: 문자와 숫자
숫자의 종류: 정수와 실수

	수학	python
정수	자연수, 0, 음의 정수	주어진 범위 내에서의 모든 정수
실수	유리수, 무리수	주어진 범위 내에서의 모든 실수 중 컴퓨터가 표현할 수 있는 수
관계	정수는 실수에 포함	정수와 실수가 구분됨 (3 or 3.0)

int(실수)=정수
float(정수)=실수

부동 소수점 방식(Floating Point System)
움직이지 않는다는 뜻의 부동(不動)이 아님. 소수점의 위치가 떠다니면서 움직인다는 뜻의 부동(浮動)
주어진 실수는 $x \times 2^y$ ($1 \leq x < 2$, y는 정수) 꼴로 표현되며, 특성상 완벽하게 정확할 수는 없다. 즉 컴퓨터가 정확히 표현할 수 있는 실수는 일부에 불과하다.

input 함수
입력: 사용자에게 입력을 유도할 문자열
출력(반환): 사용자가 입력한 문자열

다음 시간에 배울 내용: 리스트, for 반복문, 반복 제어(break/continue)

스터디 복습 자료

5월 21일 배운 내용 복습 - 1 (슬라이드 #5, #6, #7일부)

리스트(List)는 여러 개의 값을 순서대로 담아 두고 꺼내 쓸 수 있는 자료형
index는 0부터 시작하고, index를 이용해서 값을 하나씩 꺼낼 수 있음
알아두기 리스트의 index가 0부터 시작하는 이유는 교재 132페이지에 잘 설명되어 있음

list_members = ['정나경', '설태웅', '이정주', '오유진', '최진', '황동진', '문승기', '윤혜원']

index	0	1	2	3	4	5	6	7
value	정나경	설태웅	이정주	오유진	최진	황동진	문승기	윤혜원

print(list_members[3]) # '오유진'

리스트에 값 더하기 (list.append)
리스트에 딸려 있는 append 기능을 이용해서 값을 추가할 수 있음

list_members.append('지승훈')

index	0	1	2	3	4	5	6	7	8
value	정나경	설태웅	이정주	오유진	최진	황동진	문승기	윤혜원	지승훈

또한 리스트를 합치는 형식으로 값을 추가할 수 있음

list_members_all = list_members + ['정효리'] # 실제로는 리스트 두 개가 합쳐진 것

index	0	1	2	3	4	5	6	7	8	9
value	정나경	설태웅	이정주	오유진	최진	황동진	문승기	윤혜원	지승훈	정효리

리스트에서 값 찾기 (n in list)
'한동희' in list_members_all # False

리스트에서 값 지우기 (del, remove)
del은 index, remove는 value를 이용해 값을 지운다.

del list_members_all[8] # index==8인 값을 지움

index	0	1	2	3	4	5	6	7	8
value	정나경	설태웅	이정주	오유진	최진	황동진	문승기	윤혜원	정효리

list_members_all.remove('설태웅') # value=='설태웅'인 값을 지움 (중복될 경우 먼저 나온 값을 지움)

index	0	1	2	3	4	5	6	7
value	정나경	이정주	오유진	최진	황동진	문승기	윤혜원	정효리

리스트의 더 많은 기능들에 대해서는 방학 때 다룰 예정입니다.

스터디 복습 자료

5월 21일 배운 내용 복습 - 2 (슬라이드 #5, #6, #7일부)

반복문의 기본 개념
그때그때 함수를 직접 실행하는 것은 크기가 많아지면 힘들다. 그래서 프로그래밍 언어에서는 원하는 횟수만큼 특정한 코드를 실행할 수 있게 만들어 준다. 조건문(if)처럼, 반복문 또한 블록을 가지며 정해진 횟수만큼 실행된다.

for in list
리스트에 있는 값을 하나씩 가져와서 변수에 대입한다.

Index	0	1	2	3	4	5	6	7
Value	정나경	이정주	오유진	최진	황동진	문승기	윤혜원	정효리

```
for member in list_members_all:
    print(member) # 정나경, 이정주, 오유진, 최진, 황동진, 문승기, 윤혜원, 정효리 순서대로 출력
```

for in range(n)
변수가 0부터 시작해 1씩 커지면서 (n-1)까지 총 n번 실행한다.
리스트에 크기만큼 실행하고 싶을 때는 len(list)를 이용하면 된다.

```
for i in range(10)
    print(i) # 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 순서대로 출력
```

while 반복문
반복문은 반복 횟수에 따라서도 실행할 수 있으나, 반복 조건에 따라서도 실행할 수 있다.
while 반복문은 if문처럼 특정한 조건을 받아서, 그 조건이 참일 때에만 코드를 반복한다.

```
i = 0
length = len(list_members_all)
while (i < length):
    print(list_members_all[i])
    i = i + 1
```

for문은 정해진 횟수(≠ ∞)대로 실행되기 때문에 어느 시점에서 반복문이 끝나게 되어 있다.
하지만 while의 조건이 항상 참일 경우 (Tautology) while문이 끝나지 않는다(무한 루프).

반복 제어
반복문은 끝날 때까지 끝난 게 아니지만, 반복문을 끝내게 하는 방법이 있다. - continue, break

	continue	break
특징	반복문의 이번 반복을 끝내고 다음 반복을 시작한다.	반복문에서 즉시 빠져나간다.
공통점	반복문의 남은 코드를 무시한다.	

파이썬은 조건문과 반복문에 블록(block)을 사용하기 때문에, 이들이 너무 많아지면 가독성이 낮아진다.
무시해도 되는 항목에 반복 제어를 사용하면 경우의 수를 줄여서 if문의 개수를 절약할 수 있다.

스터디 복습 자료

5월 28일 배운 내용 복습 (슬라이드 #7, #8)

모듈? '다른 기능을 가져와 쓰는 방법'
가져와 쓰는 방법? import 모듈이름

모듈 'math' 소개
math.pi
math.sqrt(49) # 7 (제곱근을 구하는 함수, n ** 0.5 연산으로도 제곱근을 구할 수 있음)
math.ceil(5.5) # 6 올림
math.floor(5.5) # 5 내림
round(5.5) # 6 반올림 (math 모듈에 속하지 않은 기본 함수)

모듈 'random' 소개
item = random.choice(list) # list의 항목 중 무작위로 하나가 item에 들어감
random.shuffle(list) # list의 항목들의 순서가 무작위로 섞임

모듈 'urllib' 소개
웹페이지를 긁어올 수 있는 기능을 가짐

python의 모듈과 패키지

모듈/패키지	설명
Numpy	행렬 및 벡터 연산
Scipy	과학 분석 알고리즘
Pandas	행렬 및 벡터에 기반한 데이터 분석 도구
matplotlib	데이터를 차트나 플롯으로 나타내는 도구
tensorflow	딥러닝을 위한 핵심 도구
keras	딥러닝을 위한 핵심 도구

방학때 배울 내용
또 다른 자료 구조: 딕셔너리 { : } 그리고 튜플 (,)
활용하기, 예외 처리
논리 연산과 if문 더 알아보기
리스트 더 알아보기
클래스와 객체 지향 프로그래밍
상속과 다형성
조건제시법
날짜와 시간
자주 만나는 오류
파이썬 데이터 분석 입문