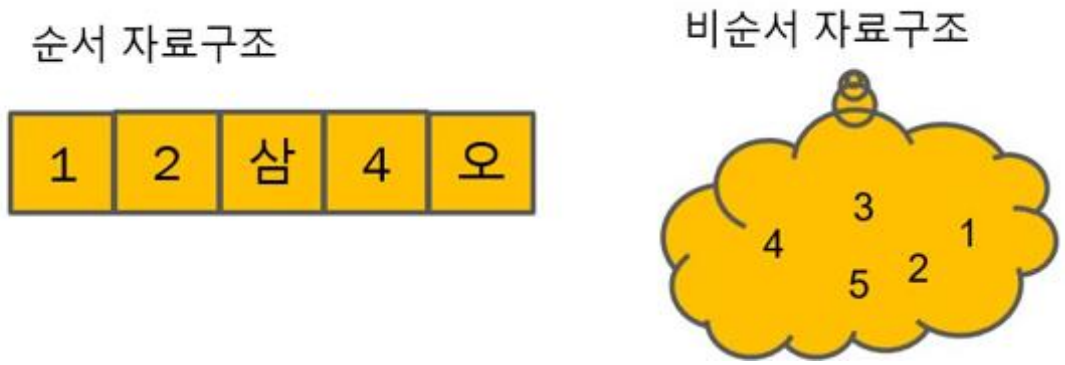
UNISTUDY

파이썬

11강. 자료구조(str, 리스트)

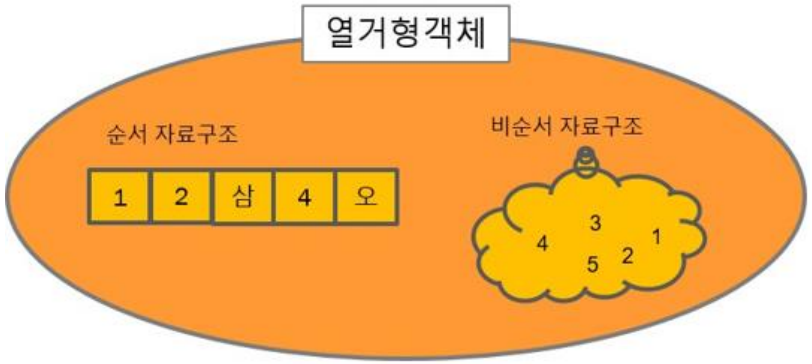
1. 자료구조

- 프로그래밍에 의해서 만들어진 객체가 메모리에 배정될 때 기억공간에 적재되는 구조를 자료구조 (Data Structure)라고 부른다.
- 적재되는 모양에 따라서 자료구조는 크게 순서 자료구조와 비순서 자료구조로 구분할 수 있다.
- 순서 자료구조와 비순서 자료구조를 나타내고 있다. 순서 자료구조는 칸막이(partition)로 구분된 영역에 원소들이 순서대로 적재된다. 하지만 비순서 자료구조 는 하나의 큰 메모리 영역에 원소들이 적재되기 때문에 순서가 보장되지 않는다.



1. 자료구조

- 자료구조를 열거형객체와 관련해서 생각해 볼 때 다음과 같은 포함관계를 갖는다. 열 거형객체와 자료구조의 관계를 나타내고 있다. 열거형객체와 자료구조의 특징을 정리하면 다음과 같다.
- 열거형객체 : 하나의 메모리 영역에 여러 개의 자료가 나열된 집합 자료구조를 의미한다.
- 자료구조 : 열거형객체의 자료구조 형태가 순서를 갖고 있는 순서 자료구조와 순서가 없는 비순서 자 료구조로 구분된다. 순서와 비순서의 차이점은 참조변수의 색인으로 접근할 수 있으면 순서, 색인을 이용할 수 없으면 비순서로 구분한다.



2. 순서 자료구조

- 파이썬에서는 순서 자료구조(Sequence Data Structure)를 생성하기 위해서 str, list, tuple 등 의 클래스를 제공한다.
- 클래스는 특정 객체가 만들어지도록 설계되어진 설계도를 의미하는데, 클래스 의 상세한 문법은 나중에 다루도록 한다.
- 순서 자료구조의 객체를 생성하는 관련 클래스들 에 대해서 알아본다.

3. str

- str 클래스는 문자열 객체를 만들어주는 클래스이다. 다음과 같이 프롬프트(>>>) 다음에 help(str)
- 명령문으로 도움말을 확인할 수 있다.
- 가장 첫 번째 'class range in module builtins' 문장에서 str 클래스는 builtins 모듈에 포함된 클래스임을 나타내고 있다. str 클래스는 object='' 인수를 이용하여 문자열 객체를 만든다. 예를 들면 str_var = str(object='대한민국') 명령문으로 '대한민국' 문자열 객체를 생성할 수 있다. 하지만 위와 같은 str 클래스의 형식을 이용하지 않고 str_var = '대한민국' 명령문으로 간편하게 작성할 수 있도록 편의를 제공하고 있다.

```
>>>help(str)
Help on class range in module builtins:

class str(object)
 |  str(object='') -> str
```

3. str

chapter04.lecture.step01_str.py

```
# (1) str 클래스 형식
str_var = str(object='string')
print(str_var)
print(type(str_var))
print(str_var[0])
print(str_var[-1])

# (2) str 클래스 간편 형식
str_var2 = 'string'
print(str_var2)


print(type(str_var2))
print(str_var2[0])
print(str_var2[-1])
```

Python Console

```
string
<class 'str'>
s
g

string


<class 'str'>
s
g
```

3. str

- 문장과 단어 추출 예
- (1) str 클래스 형식
- str 클래스에 의해서 생성된 ‘string’ 문자열 객체와 자료형 그리고 참조변수의 색인으로 첫 번째 문자와 오른쪽 마지막 문자가 참조되는 예문이다.
- (2) str 클래스 간편 형식
- str 클래스를 이용하지 않고 간편 형식으로 ‘string’ 문자열 객체가 생성되는 예문이다. 객체와 자료형 그리고 참조변수의 색인으로 참조되는 내용이 모두 str_var 참조변수와 동일한 것을 알 수 있다. 따라서 문자열 객체는 간편 형식을 더 많이 이용한다.

4. 리스트(list)

- 리스트는 여러 개의 자료를 순서대로 적재하는 가변 길이 순차 자료구조를 생성하는 클래스이다. 다음 은 리스트 클래스의 도움말을 확인한 결과이다.

```
>>>help(list)
Help on class list in module builtins:

class list(object)
 |  list(iterable=(), /)
 |
 |  Built-in mutable sequence.
 |
 |  If no argument is given, the constructor creates a new empty list.
 |  The argument must be an iterable if specified
```

4. 리스트(list)

- 가장 첫 번째 'class range in module builtins' 문장에서 list 클래스는 builtins 모듈에 포함된 클래스임을 나타내고 있다. list 클래스는 object 인수를 이용하여 객체를 만든다. '가변 길이 순차 자료구조이다. 만약 인수가 없으면 생성자8)는 빈 list을 만든다. 인수는 반복가능 해야 한다.' 라고 리 스트 클래스를 설명하고 있다.

5. 리스트 객체 특징

- 순서 자료구조를 갖는 열거형객체를 생성할 수 있다.
- 다음 형식과 같이 대괄호([])안에 콤마(,)를 이용하여 순서대로 값을 나열한다.

```
변수 =  [값1, 값2, ..... 값n]
```

- 값의 자료형은 숫자형, 문자형, 논리형 등을 함께 사용할 수 있다.
- 색인(index)을 이용하여 자료를 참조할 수 있고, 슬라이싱, 연결, 반복, 요소 검사 등이 가능하다.
- 값을 추가, 삽입, 수정, 삭제가 가능하다.

5. 리스트 객체 특징

chapter04.lecture.step02_list.py

```
# (1) 단일 list 예

lst = [1, 2, 3, 4, 5]
print(lst)
print( type(lst) )

for i in lst :
    print(lst[:i]) # i 전까지
```

Python Console

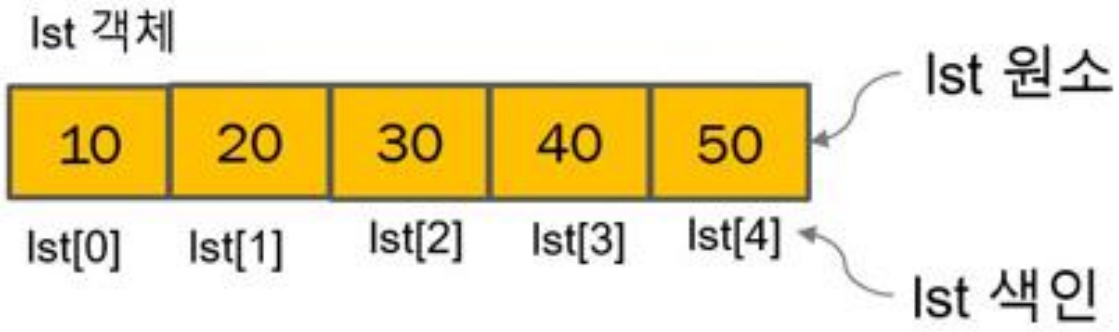
```
[1, 2, 3, 4, 5]
<class 'list'>

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

5. 리스트 객체 특징

- 단일 리스트 객체 예
- (1) 단일 리스트 예
- 1~5까지 5개의 원소를 갖는 lst 객체를 생성하고, list의 전체 원소와 자료형을 출력하고 있다. 다음 네 번째 줄은 lst가 for 문에서 열거형객체로 이용되면서 원소의 길이만큼 5회 반복된다. 5회 반복과정에서 i변수는 lst의 색인으로 이용된다. 색인[:i]은 처음부터 i번째 이전까지 슬라이싱 되기 때문에 lst 원소를 처음부터 하나씩 꺼내서 출력된다. 한편 단일 리스트란 하나의 리스트로만 구성된 리스트 형식을 의미한다.(리스트 내에 다른 리스트가 포함된 경우는 중첩 리스트라고 한다.)

6. 리스트 색인

- 리스트는 순서 자료구조이기 때문에 색인을 이용해서 리스트의 원소를 참조할 수 있다. 즉 색인은 자료 가 저장된 위치를 의미한다.
- lst 객체의 첫 번째 색인은 lst[0]으로 표현하며, 이후 나머지 원소의 색인은 1씩 증가하는 형식으로 표현한다.



6. 리스트 색인

chapter04.lecture.step02_list.py

```
# (2) 단일 list 색인
x = list(range(1, 11))
print(x)
print(x[:5])
print(x[-5:])
print('index 2씩 증가 ')
print(x[::2]) # 홀수 색인
print(x[1::2]) # 1부터 시작 하는 짝수 색인
```

Python Console

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
index 2씩 증가
[1, 3, 5, 7, 9]
[2, 4, 6, 8, 10]
```

6. 리스트 색인

- (2) 단일 리스트색인
- list(range(1, 11)) 명령어는 range 클래스에 의해서 열거형객체를 생성한 후 리스트 객체로 변환하 는 명령문이다. 참조 변수 x를 출력하면 대괄호 안에 1~10의 원소를 갖는 열거형객체가 출력된다. 색인(x[:5])에 의해서 첫 번째 원소부터 5번째 원소가 출력된다. 색인(x[-5:]에 의해서 오른쪽 끝을 기준으로 5번째부터 마지막까지의 원소가 출력된다. 끝으로 색인(x[::2])은 'start:stop:step' 형식에 의해서 전체 원소를 대상으로 2씩 증가하는 색인에 의해서 홀수 번째 원소가 출력되고, 색인 (x[1::2])은 2번째 원소부터 끝 까지 2씩 증가하는 색인에 의해서 짝수 번째 원소가 출력된다.

7. 중첩 list

- 리스트 내에 또 다른 리스트가 포함된 형식을 중첩 리스트라고 한다.
- 중첩 리스트는 바깥쪽 리스트의 색인을 통해서 안쪽 리스트의 원소를 참조할 수 있다.

7. 중첩 list

chapter04.lecture.step02_list.py

```
# (1) 단일 리스트 객체 생성
a = ['a', 'b', 'c']
print(a)


# (2) 중첩 리스트 객체 생성
b = [10, 20, a, 5, True, '문자열'] # 서로 다른 자료형
print(b[0]) # 10
print(b[2]) # ['a', 'b', 'c'] -> 중첩 list
print(b[2][0]) # a -> 중첩 list 1번 원소
print(b[2][1:]) # ['b', 'c'] -> 중첩 list 2번 이후 원소
```

Python Console

```
['a', 'b', 'c']


10
['a', 'b', 'c']
a
['b', 'c']
```

7. 중첩 list

- 중첩 리스트 객체 예
- (1) 단일 리스트 객체 생성
- 변수 a는 세 개의 영문자를 갖는 단일 list 객체가 생성된다.
- (2) 중첩 리스트 객체 생성
- 변수 b는 변수 a를 포함하고 있는 중첩 리스트 객체이다. 변수 b에 포함된 원소 중에서 3번째 원소가 중첩 리스트인 변수 a에 해당되기 때문에 중첩 리스트로 접근하기 위해서는 b 변수의 색인(b[2])으로 접근한다. 그리고 중첩 리스트의 특정 원소를 참조하기 위해서는 b 변수의 색인과 a 변수의 색인을 순서대로 나열한다. 예를 들면 a의 첫 번째 원소를 참조하기 위해서는 b[2][0] 형식으로 색인을 작성한다.

8. 리스트 연산

- 리스트는 사칙연산에서 제공되는 기호를 이용하여 결합(+)과 두 배 확장(*)이 가능하다. 또한 리스트 객체와 리스트 객체 간의 확장과 추가 연산이 가능하다.

8. 리스트 연산

chapter04.lecture.step02_list.py

```
# (1) 리스트 결합
x =  [1, 2, 3, 4]
y = [1.5, 2.5]
z = x + y  # new object
print(z) # [1, 2, 3, 4, 1.5, 2.5]

# (2) 리스트 확장
x.extend(y)  # x 확장
print(x) # [1, 2, 3, 4, 1.5, 2.5]

# (3) 리스트 추가
x.append(y) # x 추가
print(x) # [1, 2, 3, 4, 1.5, 2.5, [1.5, 2.5]]

# (4) 리스트 두 배 확장
lst = [1, 2, 3, 4] # list 생성
result = lst * 2 # 각 원소 연산 안됨
print(result) # [1, 2, 3, 4, 1, 2, 3, 4]
```

Python Console

```
[1, 2, 3, 4, 1.5, 2.5]
[1, 2, 3, 4, 1.5, 2.5]
[1, 2, 3, 4, 1.5, 2.5, [1.5, 2.5]]
[1, 2, 3, 4, 1, 2, 3, 4]
```

8. 리스트 연산

- 추가, 삭제, 수정, 삽입 예
- (1) 리스트 결합
- 변수 x의 객체와 변수 y의 객체가 결합되어 새로운 변수 z의 객체가 생성된다.
- (2) 리스트 확장
- 변수 x의 객체를 변수 y의 객체로 확장한다.
- (3) 리스트 추가
- 변수 x의 객체에 변수 y의 객체를 추가하여 중첩 리스트 형태로 만든다.
- (4) 리스트 두 배 확장
- 변수 lst의 객체에 곱셈(*) 기호를 이용하면 lst 객체의 원소가 두 배로 확장되어 새로운 객체가 생성된다.

9. 리스트 정렬과 요소 검사

- 리스트의 원소가 숫자인 경우 오름차순 또는 내림차순으로 정렬(sort) 할 수 있다. 또한 리스트 원소는 대량의 자료를 보관할 수 있기 때문에 리스트 원소 중에서 특정 값의 존재 유무를 알려주는 기능도 제 공한다.
- 만약 찾는 값이 있으면 True가 반환되고, 없으면 False가 반환된다

9. 리스트 정렬과 요소 검사

chapter04.lecture.step02_list.py

```
# (1) 리스트 정렬
print(result) # [1, 2, 3, 4, 1, 2, 3, 4]
result.sort() # 오름차순 정렬
print(result) #[1, 1, 2, 2, 3, 3, 4, 4]
result.sort(reverse = True) # 내림차순 정렬
print(result) # [4, 4, 3, 3, 2, 2, 1, 1]

# (2) 리스트 요소 검사
import random
r = [] # 빈 list
for i in range(5) :
    r.append(random.randint(1,5))

print(r)
if 4 in r :
    print('있음')
else :
    print('없음')
```

Python Console

```
[1, 2, 3, 4, 1, 2, 3, 4]
[1, 1, 2, 2, 3, 3, 4, 4]
[4, 4, 3, 3, 2, 2, 1, 1]
[3, 4, 5, 5, 3]
있음
```

9. 리스트 정렬과 요소 검사

- 리스트 정렬과 요소 검사 예
- (1) 리스트 정렬
- 변수 result의 객체를 대상으로 sort()함수를 적용하면 오름차순 정렬되고, reverse = True 인자 를 추가하면 내림차순 정렬된다. sort()함수의 reverse 인자는 False가 기본값이다.
- (2) 리스트 요소 검사
- 변수 r은 빈 목록으로 생성된 리스트 객체를 참조한다. for문에서 range 클래스에 의해서 생성된 5개 원소를 갖는 열거형객체의 원소를 하나씩 변수 i로 할당 받아서 5회 반복한다. 반복 과정에서 1과 5 사이의 난수 정수를 하나씩 만들어서 변수 r의 리스트 객체에 추가된다. 끝으로 if문을 이용해서 변수 r의 리스트 객체에 원소 4가 포함되어 있는지의 요소 검사를 한다. 만약 변수 r에 난수 4가 포함되어 있으면 True가 반환되어 '있음'이 출력되고, 없으면 False가 반환되어 '없음'이 출력된다.

THANK YOU