자료구조(Data Structure)와 리스트(List) 자료형

자료구조(Data Structure)

- **숫자형, 문자형 데이터는 하나의 변수에 1개의 데이터를 저장**

- **프로그램에서 사용되는 데이터는 복잡할 수 있음**
  - 매분 마다 수집 된 하루 치 주식 가격 데이터 → 1440개의 변수 사용?
  - 전 세계 모든 국가들의 이름을 저장 → 206개의 변수 사용?
  - 한 사람의 정보를 저장 → 수십 개의 변수 사용? (이름, 성별, 키, 몸무게, 주소, ...)

- **관련 있는 데이터들을 그룹화 하여 저장하고 처리할 수는 없을까?**

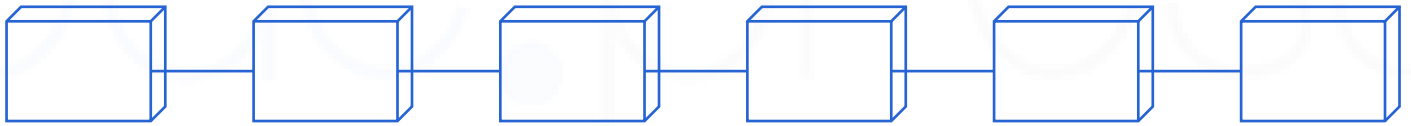
자료구조(Data Structure)

- **관련 있는 데이터들을 그룹화 하여 하나의 변수에 저장**
  - 그룹화 된 데이터들에 대한 추가적인 연산도 제공
- **매분 마다 수집 된 하루 치 주식 가격 데이터**
  - today_stock_prices = [34500, 34200, 34000, 34100, …. , 35600]
- **전 세계 모든 국가들의 이름을 저장**
  - names_of_country = [Korea, USA, Italy, China, … , Japan]
- **한 사람의 정보를 저장**
  - personal_info = { ‘name’: 'Jinsoo Kim’, ‘gender’: ‘Male’, ‘height’: 180, ‘nationality’: ‘Korea’ }

파이썬 리스트(List)

- 다양한 자료형의 데이터들을 그룹화하여 순서대로 저장하는 자료구조
  - [34500, 34200, 34000, 34100, …. , 35600]
  - ['Jinsoo Kim', 'Male', 20, 181.9, 'Korea']

- 데이터를 담을 수 있는 공간이 연결되어 있는 형태



- 파이썬에서 가장 많이 사용 되는 자료구조 중 하나

파이썬 리스트(List)의 생성

- 리스트는 대괄호'[]'를 사용하여 생성

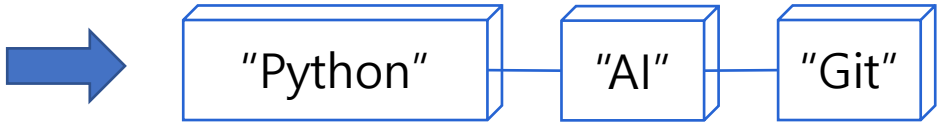
```
course_list = []
```

```
course_list = ["Python"]
```

- 리스트 내의 데이터들은 콤마 ',' 를 사용하여 구분

```
course_list = ["Python", "AI"]
```


```
course_list = ["Python", "AI", "Git"]
```



리스트 안의
데이터 가져오기

리스트 인덱싱(Indexing)

- 문자열 인덱싱

```
name = "CodePresso"
print(name[0])
```

C

문자열	C	o	d	e	P	r	e	s	s	o
인덱스	0	1	2	3	4	5	6	7	8	9

- 리스트 인덱싱

```
course_list = ["Python", "AI", "Git", "Clean Code", "Tensorflow2.0"]
print(course_list[0])
```

Python

문자열	"Python"	"AI"	"Git"	"Clean Code"	"Tensorflow2.0"
인덱스	0	1	2	3	4

리스트 인덱싱(Indexing)

- 리스트의 인덱스를 활용하여 특정 데이터에 접근 가능
- 인덱스는 0부터 시작, 리스트의 길이 - 1 까지 존재

```
course_list = ["Python", "AI", "Git", "Clean Code", "Tensorflow2.0"]
print(course_list[1])
```

→ AI

문자열	"Python"	"AI"	"Git"	"Clean Code"	"Tensorflow2.0"
인덱스	0	1	2	3	4

```
course_list = ["Python", "AI", "Git", "Clean Code", "Tensorflow2.0"]
print(course_list[4])
```

→ Tensorflow2.0

문자열	"Python"	"AI"	"Git"	"Clean Code"	"Tensorflow2.0"
인덱스	0	1	2	3	4

리스트의 IndexError

- 인덱스 범위를 벗어나면 IndexError 발생

```
course_list = ["Python", "AI", "Git", "Clean Code", "Tensorflow2.0"]
print(course_list[5])
```

문자열	"Python"	"AI"	"Git"	"Clean Code"	"Tensorflow2.0"	?
인덱스	0	1	2	3	4	

```
Traceback (most recent call last):
  File "./prog.py", line 54, in <module>
IndexError: list index out of range
```

파이썬 리스트 음수 인덱스

- 파이썬 리스트는 음수의 인덱스를 제공
- 리스트의 크기가 클 때 뒷부분의 데이터를 접근하기에 용이

문자열	"Python"	"AI"	"Git"	"Clean Code"	"Tensorflow2.0"
인덱스	0	1	2	3	4
음수 인덱스	-5	-4	-3	-2	-1

```
course_list = ["Python", "AI", "Git", "Clean Code", "Tensorflow2.0"]
print(course_list[4])
print(course_list[-1])
```

```
Tensorflow2.0
Tensorflow2.0
```

리스트 슬라이싱(Slicing)

- 인덱스의 시작/끝 값을 지정하여 연속 데이터에 접근
- 슬라이싱 문법 - 리스트변수명[시작 인덱스:끝 인덱스]
  - 시작 인덱스의 값을 포함
  - 끝 인덱스의 값은 포함하지 않음, 끝 인덱스 바로 직전 인덱스 까지 포함
- 리스트 슬라이싱의 결과는 또 다른 리스트

```
course_list = ["Python", "AI", "Git", "Clean Code", "Tensorflow2.0"]
print(course_list[0:3])
```

```
['Python', 'AI', 'Git']
```

문자열	"Python"	"AI"	"Git"	"Clean Code"	"Tensorflow2.0"
인덱스	0	1	2	3	4
음수 인덱스	-5	-4	-3	-2	-1

리스트 슬라이싱(Slicing) 예제

```
course_list = ["Python", "AI", "Git", "Clean Code", "Tensorflow2.0"]
print(course_list[1:2])
```

```
['AI']
```

문자열	"Python"	"AI"	"Git"	"Clean Code"	"Tensorflow2.0"
인덱스	0	1	2	3	4
음수 인덱스	-5	-4	-3	-2	-1

```
course_list = ["Python", "AI", "Git", "Clean Code", "Tensorflow2.0"]
print(course_list[-3:-1])
```

```
['Git', 'Clean Code']
```

문자열	"Python"	"AI"	"Git"	"Clean Code"	"Tensorflow2.0"
인덱스	0	1	2	3	4
음수 인덱스	-5	-4	-3	-2	-1

리스트 슬라이싱(Slicing) 예제

- 시작 인덱스 값이 없으면 가장 처음 데이터 부터 사용

```
course_list = ["Python", "AI", "Git", "Clean Code", "Tensorflow2.0"]
print(course_list[:3])
```

```
['Python', 'AI', 'Git']
```

문자열	"Python"	"AI"	"Git"	"Clean Code"	"Tensorflow2.0"
인덱스	0	1	2	3	4
음수 인덱스	-5	-4	-3	-2	-1

리스트 슬라이싱(Slicing) 예제

- 끝 인덱스 값이 없으면 가장 마지막 데이터 까지 사용

```
course_list = ["Python", "AI", "Git", "Clean Code", "Tensorflow2.0"]
print(course_list[3:])
```

```
['Clean Code', 'Tensorflow2.0']
```

문자열	"Python"	"AI"	"Git"	"Clean Code"	"Tensorflow2.0"
인덱스	0	1	2	3	4
음수 인덱스	-5	-4	-3	-2	-1

리스트 슬라이싱(Slicing) 예제

```
course_list = ["Python", "AI", "Git", "Clean Code", "Tensorflow2.0"]
print(course_list[:])
```

```
['Python', 'AI', 'Git', 'Clean Code', 'Tensorflow2.0']
```

문자열	"Python"	"AI"	"Git"	"Clean Code"	"Tensorflow2.0"
인덱스	0	1	2	3	4
음수 인덱스	-5	-4	-3	-2	-1

리스트 데이터 다루기

리스트 데이터 다루기

- 파이썬 리스트는 내부 데이터 변경이 가능한 자료구조

- 기존에 생성 된 리스트 데이터를 변경 가능
  - 새로운 데이터 추가
  - 기존 데이터 수정
  - 기존 데이터 삭제

리스트에 데이터 새로운 추가하기

- 리스트의 특정한 위치에 새로운 데이터를 추가

- 데이터 추가를 위한 다양한 방법을 제공
  - append()
  - insert()
  - extend()

리스트에 데이터 새로운 추가하기 - append()

- 리스트의 가장 마지막 위치에 새로운 데이터 추가
- 리스트변수명.append(추가할 데이터)

```
course_list = ["Python", "AI"]
print(course_list)
```

```
['Python', 'AI']
```

문자열	"Python"	"AI"
인덱스	0	1

```
course_list.append("Cloud Computing")
print(course_list)
```

```
['Python', 'AI', 'Cloud Computing']
```

문자열	"Python"	"AI"	"Cloud Computing"
인덱스	0	1	2

리스트에 데이터 새로운 추가하기 - insert()

- 인덱스 번호를 사용하여 특정한 위치에 새로운 데이터 추가
- 리스트변수명.insert(인덱스번호, 추가할 데이터)

```
course_list = ["Python", "AI"]
print(course_list)
```

```
['Python', 'AI']
```

문자열	"Python"	"AI"
인덱스	0	1

```
course_list.insert(0, "Git")
print(course_list)
```

```
['Git', 'Python', 'AI']
```

문자열	"Git"	"Python"	"AI"
인덱스	0	1	2

리스트에 데이터 새로운 추가하기 - insert()

```
course_list = ["Python", "AI"]
print(course_list)
```

```
['Python', 'AI']
```

문자열	"Python"	"AI"
인덱스	0	1

```
course_list.insert(1, "Git")
print(course_list)
```

```
['Python', 'Git', 'AI']
```

문자열	"Python"	"Git"	"AI"
인덱스	0	1	2

리스트에 데이터 새로운 추가하기 - insert()

```
course_list = ["Python", "AI"]
print(course_list)
```

```
['Python', 'AI']
```

문자열	"Python"	"AI"
인덱스	0	1

```
course_list.insert(2, "Git")
print(course_list)
```

```
['Python', 'AI', 'Git']
```

문자열	"Python"	"AI"	"Git"
인덱스	0	1	2

리스트에 데이터 새로운 추가하기 - extend()

- 리스트에 또 다른 리스트를 추가, 두 개의 리스트를 결합
- 리스트1변수명.extend(리스트2변수명)

```
course_list = ["Python", "AI"]
new_course_list = ["Git", "Clean Code"]
```

course_list

문자열	"Python"	"AI"
인덱스	0	1

new_course_list

문자열	"Git"	"Clean Code"
인덱스	0	1

리스트에 데이터 새로운 추가하기 - extend()

```
course_list = ["Python", "AI"]
new_course_list = ["Git", "Clean Code"]

course_list.extend(new_course_list)
print(course_list)
```

```
['Python', 'AI', 'Git', 'Clean Code']
```

문자열	"Python"	"AI"	"Git"	"Clean Code"
인덱스	0	1	2	3

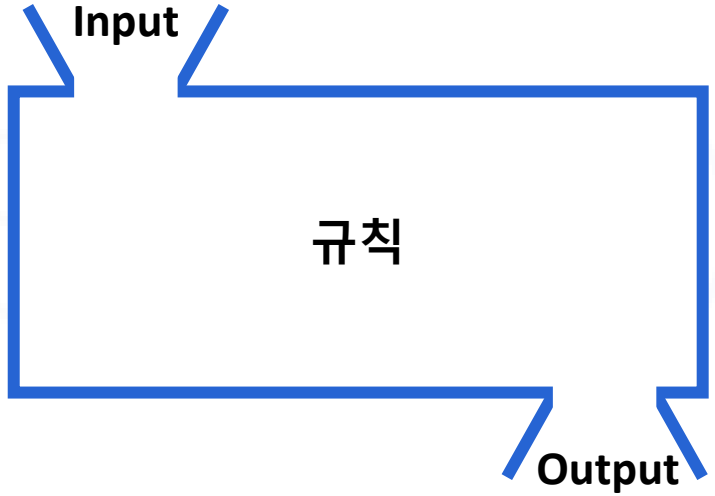
리스트에 데이터 새로운 추가하기 - 요약

- append() - 리스트의 가장 마지막에 새로운 데이터 추가
- insert() - 지정한 인덱스에 새로운 데이터 추가
- extend() - 2개의 리스트를 연결

함수에 대한 간단한 설명

- **print(), type(), append(), insert(), extend() 는 무엇일까요?**

- **함수 (또는 메소드)**
  - 특정 동작을 수행하는 코드의 묶음
  - 함수를 사용(호출)하면 특정 작업을 위임 가능
  - **함수이름()**의 형태
  - 함수의 이름은 그 함수가 어떤 동작을 하는지 표현
  - ()안에는 함수 내부에서 사용 되는 데이터를 입력(Argument)
    - course_list.append(**"Cloud Computing"**)



기존 리스트 데이터 변경하기

- 인덱스를 사용하여 특정 데이터를 변경 가능

```
course_list = ["Python", "AI", "Git"]
```

문자열	"Python"	"AI"	"Git"
인덱스	0	1	2

```
course_list[1] = "Deep Learning"
print(course_list)
```

문자열	"Python"	"Deep Learning"	"Git"
인덱스	0	1	2

기존 리스트 데이터 변경하기

- 인덱스를 사용하여 특정 데이터를 변경 가능

```
course_list = ["Python", "AI", "Git"]
```

문자열	"Python"	"AI"	"Git"
인덱스	0	1	2

```
course_list[2] = "Github"
print(course_list)
```

문자열	"Python"	"AI"	"Github"
인덱스	0	1	2

기존 리스트 데이터 삭제하기 - remove()

- 삭제 대상 데이터를 명시
- remove() 함수 사용. 리스트변수명.remove(삭제 대상 데이터)

```
course_list = ["Python", "AI", "Git"]
```

문자열	"Python"	"AI"	"Git"
인덱스	0	1	2

```
course_list.remove("AI")
print(course_list)
```

```
['Python', 'Git']
```

문자열	"Python"	"Git"
인덱스	0	1

기존 리스트 데이터 삭제하기 - remove()

- remove()함수는 명시한 데이터의 첫 번째 항목만 삭제

```
course_list = ["Python", "AI", "Git", "AI"]
```

문자열	"Python"	"AI"	"Git"	"AI"
인덱스	0	1	2	3

```
course_list.remove("AI")
print(course_list)
```

```
['Python', 'Git', 'AI']
```

문자열	"Python"	"Git"	"AI"
인덱스	0	1	2

기존 리스트 데이터 삭제하기 - pop()

- 삭제 대상 데이터의 인덱스를 명시하여 삭제
- pop() 함수 사용. 리스트변수명.pop(삭제 대상 인덱스)

```
course_list = ["Python", "AI", "Git"]
```

문자열	"Python"	"AI"	"Git"
인덱스	0	1	2

```
course_list.pop(1)
print(course_list)
```

```
['Python', 'Git']
```

문자열	"Python"	"Git"
인덱스	0	1

기존 리스트 데이터 삭제하기 - pop()

- 인덱스를 명시하지 않을 경우 가장 마지막 데이터 삭제

```
course_list = ["Python", "AI", "Git"]
```

문자열	"Python"	"AI"	"Git"
인덱스	0	1	2

```
course_list.pop()
print(course_list)
```

```
['Python', 'AI']
```

문자열	"Python"	"AI"
인덱스	0	1

요약

구분	방법	설명	예시
데이터 추가	append() 함수 사용	리스트 가장 뒤에 데이터 추가	course_list**.append(“Git")**
	insert() 함수 사용	인덱스를 명시하여 데이터 추가	course_list**.insert(2, "Git")**
	extend() 함수 사용	2개의 리스트를 결합	course_list**.extend(new_course_list)**
데이터 변경	인덱스로 데이터 변경	특정 인덱스의 데이터 변경	course_list**[1] =** "Deep Learning"
데이터 삭제	remove() 함수 사용	데이터를 명시하여 삭제	course_list**.remove("AI")**
	pop() 함수 사용	인덱스에 해당하는 데이터 삭제	course_list**.pop(1)**