

Data Structure

자료구조



리스트



한국기술교육대학교
온라인평생교육원

학습내용

- 리스트의 개념과 추상 자료형
- 배열 구조와 연결된 구조
- 배열과 파이썬 리스트

학습목표

- 리스트의 개념과 추상 자료형을 설명할 수 있다.
- 배열 구조와 연결된 구조를 설명할 수 있다.
- 배열과 파이썬 리스트의 관계를 이해하고 활용할 수 있다.



리스트의 개념과 추상 자료형

리스트

리스트의 개념과 추상 자료형

1 리스트의 개념

리스트(list)

- 자료를 정리하는 방법 중 하나
- 대표적인 선형 자료구조로 **선형 리스트 (linear list)**라고도 불림
- 리스트의 표현
 - $L = [item_0, item_1, item_2, \dots, item_{n-1}]$
 - 항목들이 **순서** 또는 **위치**를 가짐

리스트

리스트의 개념과 추상 자료형

1 리스트의 개념

◎ 리스트의 예

1 죽기 전에 해야 할 버킷 리스트

2 배달해야 할 물건들

3 오늘 할 일: [청소, 자료구조 공부, 아르바이트]

4 다항식 표현 [$3x^7, 2x^4, 8x^2, 7$]

리스트의 개념과 추상 자료형

리스트

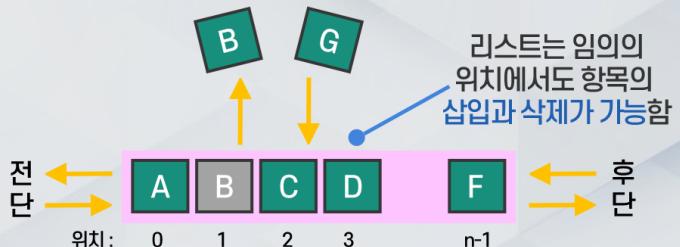
리스트의 개념과 추상 자료형

1 리스트의 개념

● 리스트의 특징

가장 자유로운 선형 자료구조

▶ 임의의 위치에서 삽입과 삭제가 가능



리스트에서 항목의 접근을 제한한 자료 구조들

▶ Stack, Queue, Deque 등

리스트

리스트의 개념과 추상 자료형

1 리스트의 개념

● 리스트와 집합(set)

리스트와 집합(set)의 차이점

- 각 항목들은 **위치**가 존재
- 항목 간에 **순서**가 있음
- 동일한 항목의 **중복**을 허용



리스트의 개념과 추상 자료형

리스트

리스트의 개념과 추상 자료형

2 리스트의 추상 자료형

● 리스트와 집합(set)

데이터 : 리스트에 저장할 자료는?

같은 유형의 서로
비교할 수 있는 요소들

연산 : 리스트로 할 수 있는 일은?

- 어떤 위치에 항목을 삽입(insert)
- 어떤 위치에 있는 항목을 꺼내서(delete) 반환
- 리스트가 공백상태(Empty)인지 검사
- 리스트가 포화상태(Full)인지 검사
- 전체 항목의 개수(size)를 세어 반환
- 새로운 리스트(공백)를 만듦

리스트

리스트의 개념과 추상 자료형

2 리스트의 추상 자료형

● 리스트와 집합(set)

연산 : 리스트의 고급 기능들은?

- 리스트를 초기화(clear)
- 리스트에 어떤 항목이 있는지 탐색(find)
- 어떤 위치에 항목을 꺼내지 않고 반환(getEntry)
- 어떤 위치에 있는 항목을 새로운 항목으로 대치(replace)
- 리스트의 맨 뒤에 항목을 삽입(append)
- 리스트의 맨 뒤 항목을 꺼내고 반환(pop)
- 리스트를 화면에 보기 좋게 출력
- ...



리스트의 개념과 추상 자료형

리스트

리스트의 개념과 추상 자료형

2 리스트의 추상 자료형

- 리스트 추상 자료형(ADT)의 정의

데이터

같은 유형의 서로 비교할 수 있는 요소들

리스트

리스트의 개념과 추상 자료형

2 리스트의 추상 자료형

- 리스트 추상 자료형(ADT)의 정의

▶ 연산

List()	비어 있는 새로운 리스트를 만듦
insert(pos,e)	pos위치에 새로운 요소 e 삽입
delete(pos)	pos위치에 있는 요소를 꺼내고(삭제) 반환
isEmpty()	리스트가 비어 있는지를 검사
isFull()	리스트가 가득 차 있는지를 검사
size()	리스트 요소의 개수를 반환



리스트의 개념과 추상 자료형

리스트

리스트의 개념과 추상 자료형

2 리스트의 추상 자료형

● 리스트 추상 자료형(ADT)의 정의

▶ 연산

clear()	리스트를 초기화
getEntry(pos)	pos 위치에 있는 요소를 반환
find(e)	리스트에서 항목 e를 찾아 인덱스를 반환
replace(pos,e)	pos에 있는 항목을 e로 변경
append(e)	리스트의 맨 뒤에 새로운 항목을 추가
pop()	리스트의 맨 뒤 항목을 꺼내고 반환
display()	리스트를 화면에 보기 좋게 출력

배열 구조와 연결된 구조

리스트

배열 구조와 연결된 구조

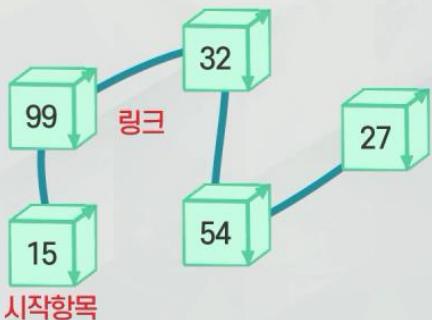
1 배열 구조와 연결된 구조의 개념

● 배열 구조의 특징

연결된 구조로 구현된 리스트 ADT

- 구현이 비교적 복잡
- 항목 접근이 비효율적: $O(n)$

연결된 구조의 리스트



- 삽입, 삭제가 효율적
- 크기가 제한되지 않음(바인더 노트와 유사)

리스트

배열 구조와 연결된 구조

2 리스트 ADT의 구현 사례

여러 프로그래밍 언어에서 자료구조 '리스트'를 이미 구현하여 제공

▶ 배열 구조 또는 연결된 구조로 구현

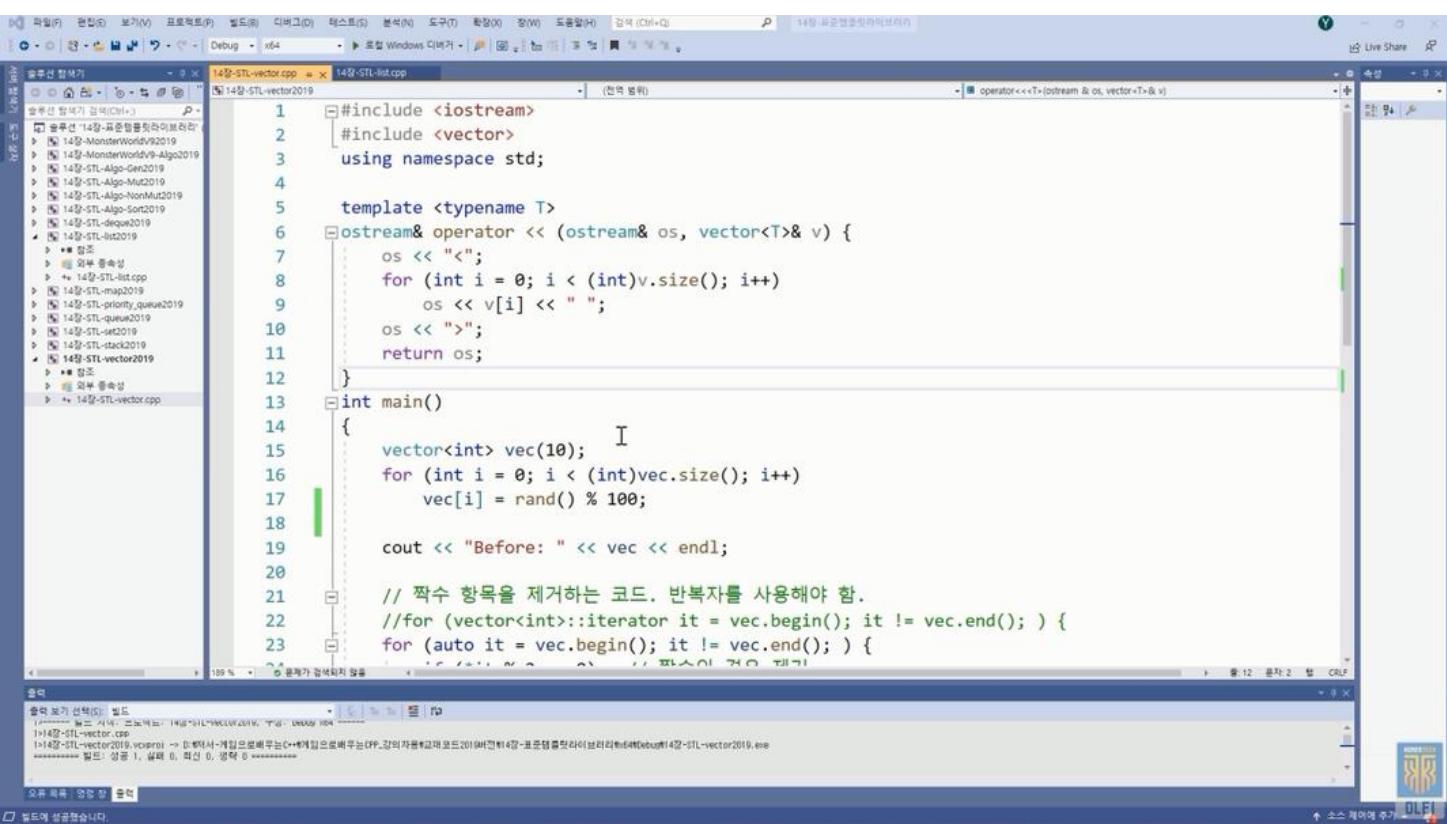
배열 구조로 구현된 리스트 ADT

- 파이썬의 리스트
- C++ STL의 vector 등

연결된 구조로 구현된 리스트 ADT

- C++ STL의 list
- Java의 LinkedList

배열 구조와 연결된 구조



The screenshot shows the Microsoft Visual Studio IDE interface. The title bar says "14장-STL-vector2019". The main window displays C++ code for a class named "operator<<". The code includes the standard headers #include <iostream> and #include <vector>. It defines a template operator<< that takes an ostream& and a vector<T>& v. The implementation uses a for loop to iterate through the vector's elements and output them separated by spaces. The main() function creates a vector of 10 integers, fills it with random values between 0 and 99, and then prints the vector to the console. A note in the code indicates that the for loop lacks a break condition, which is highlighted in red.

실습 단계

vector: 자료구조 리스트를 배열 구조로 구현한 예

C++에서 include 전처리기를 통해 vector 포함

메인 함수에 vector<int> 새로운 vector 객체를 만듦

vec.size(vector의 크기)

rend는 무작위로 숫자를 발생시킴

[i]를 이용하여 vector 각각의 항목에 접근 가능(배열 구조 리스트의 가장 큰 특징)

짝수를 없애는 코드

실행 → 숫자 랜덤 발생 → after(짝수 항목 제거 코드)

vector= 배열구조로 구현된 자료구조 리스트 / list= 연결 구조로 구현된 자료구조 리스트

배열과 파이썬 리스트

리스트

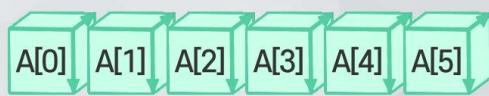
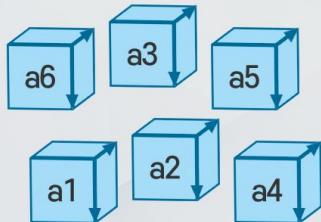
배열과 파이썬 리스트

1 배열이란?

● 배열의 개념

배열

여러 변수를 하나로 묶어 사용하는 것



- 하나의 식별자를 이용해 많은 항목들을 관리
- 반복문을 사용할 수 있음
- 크기가 고정 (예 C언어의 배열)

리스트

배열과 파이썬 리스트

1 배열이란?

● 파이썬에서의 배열

배열을 위해 리스트(list)와 튜플(tuple)을 제공

파이썬의 리스트(list)

- C언어의 배열이 진화된 형태
- 튜플과 비슷하지만 원소를 변경할 수 있다는 점이 차이점
- 파이썬에서 ‘배열’의 용도로 가장 많이 사용

배열과 파이썬 리스트

리스트

배열과 파이썬 리스트

2 파이썬 리스트

C언어의 배열에서 어떻게 진화했나?

- 배열의 크기 제한 극복
- 클래스로 구현되어 다양한 연산들을 제공

‘파이썬 리스트’는 ‘리스트 ADT’를 배열 구조로 구현한 하나의 예

파이썬 리스트를 대부분 배열의 용도로만 사용

‘배열’과 ‘파이썬 리스트’를 비슷한 개념으로 사용

C와 같은 다른 언어와의 호환을 위함

리스트

배열과 파이썬 리스트

2 파이썬 리스트

◎ 파이썬 리스트의 기본 기능

- 리스트 선언

A = []	# 빈 리스트
B = [1, 2, 3, 4]	# [1, 2, 3, 4]
C = [“hi”, “list”]	# [“hi”, “list”]

- C언어의 배열 선언

int B[4] = {1, 2, 3, 4};	/* 배열 선언 및 초기화 */
--------------------------	-------------------



배열과 파이썬 리스트

리스트

배열과 파이썬 리스트

2 파이썬 리스트

● 파이썬 리스트의 기본 기능

항목의 접근

- 파이썬 리스트는 '배열 구조'로 구현
- 인덱스 연산자를 이용해 항목에 접근 가능
- 인덱스는 0번부터 시작
- 음수 인덱스도 허용

`B = [1, 2, 3, 4]` # [1, 2, 3, 4]

`B[2]` # 앞에서 2+1번째 원소 → 3
`B[-3]` # 뒤에서 3번째 원소 → 2

리스트

배열과 파이썬 리스트

2 파이썬 리스트

● 파이썬 리스트의 기본 기능

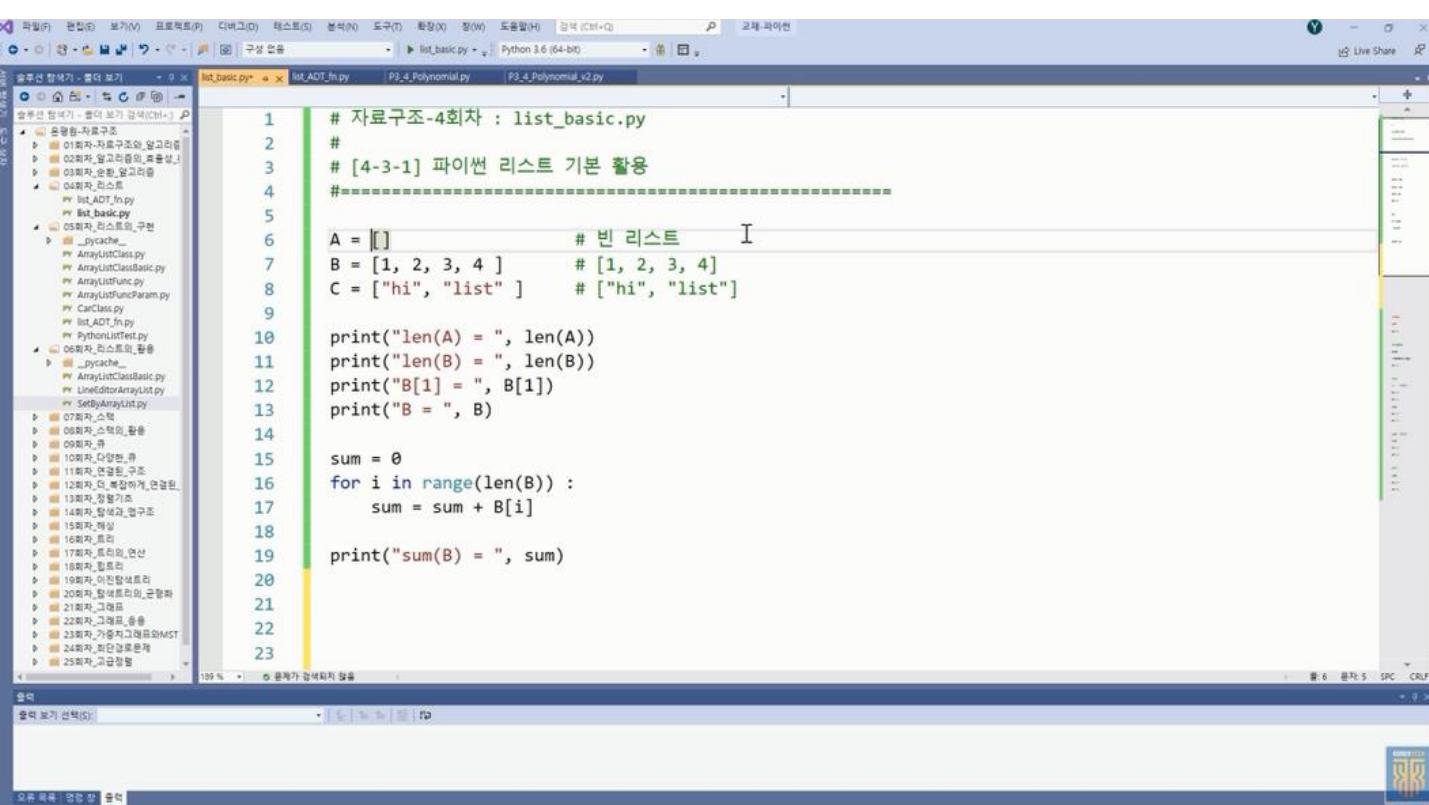
부분 리스트 만들기

- 부분 리스트 만들기: **슬라이스** 기능 사용 가능

`A[i:j]` # $A[i]$ 부터 $A[j-1]$ 까지 나열
`A[i:j:k]` # $A[i]$ 부터 $A[j-1]$ 까지 k 씩 건너뛰며 나열

`B = [1, 2, 3, 4]` # [1, 2, 3, 4]
`A = B[1:3]` # [2, 3]

배열과 파이썬 리스트

```

# 자료구조-4회차 : list_basic.py
#
# [4-3-1] 파이썬 리스트 기본 활용
=====
A = []           # 빈 리스트    I
B = [1, 2, 3, 4] # [1, 2, 3, 4]
C = ["hi", "list"] # ["hi", "list"]

print("len(A) = ", len(A))
print("len(B) = ", len(B))
print("B[1] = ", B[1])
print("B = ", B)

sum = 0
for i in range(len(B)) :
    sum = sum + B[i]

print("sum(B) = ", sum)

```

실습 단계
list의 기본적 활용 방법
빈 리스트 대괄호[] 이용
내장함수 'len' 이용 → 항목들의 수 반환. 예) len(A)=0, len(B)=4
B[1], 인덱스 연산자 사용가능(배열구조 리스트의 가장 큰 특징)
리스트 B의 모든 항목 더하기 방법
실행 결과: len(A)=0 / len(B)=4 / B[1]=2 / B=[1,2,3,4] / sum((B)=10
리스트 만들기 고급: 곱하기 연산자를 이용하여 간단히 만들기 가능
리스트 만들기 고급, 리스트 내포 실행(자주 사용하는 기능)
리스트 얕은 복사
리스트 깊은 복사
슬라이스 기능

배열과 파이썬 리스트

리스트

배열과 파이썬 리스트

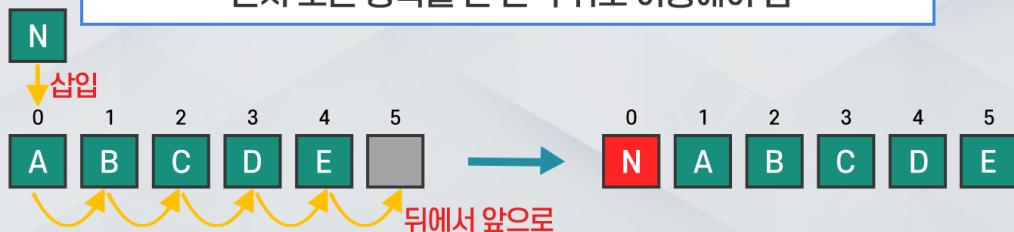
2 파이썬 리스트

● 리스트 연산들의 시간 복잡도

insert(pos,e) 연산

- pos 위치에 새로운 항목 e를 삽입
- 시간 복잡도: $O(n)$
- 예 list.insert(0,N)

배열 구조에서 맨 앞에 **새로운 항목을 삽입**하려면
먼저 모든 항목을 한 칸씩 뒤로 이동해야 함



리스트

배열과 파이썬 리스트

2 파이썬 리스트

● 리스트 연산들의 시간 복잡도

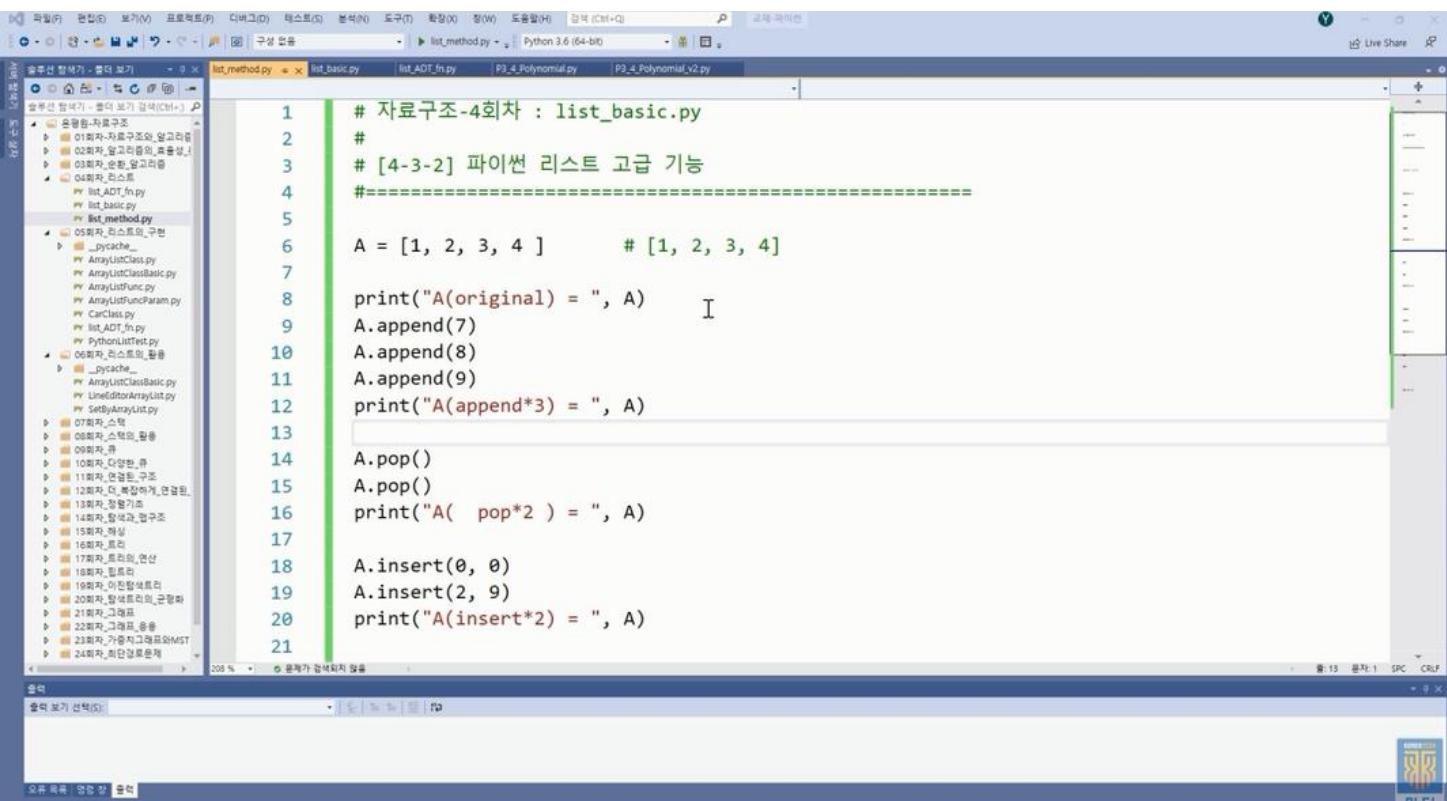
pop(pos) 연산

- pos 위치에 있는 항목을 삭제
- 시간 복잡도: $O(n)$
- 예 list.pop(0)

배열 구조에서 **맨 앞에서 항목을 삭제**하면
이후의 모든 항목들을 한 칸씩 앞으로 이동해야 함



배열과 파이썬 리스트



The screenshot shows the PyCharm IDE interface. The top bar has tabs for '파일(F)', '편집(E)', '보기(V)', '프로젝트(P)', '디버그(D)', '타스토(S)', '분석(N)', '도구(T)', '환경(X)', '창(W)', '도움말(H)', and '급색(Ctrl+Q)'. The current file is 'list_method.py' in the 'list_basic.py' tab. The code in the editor is:

```
# 자료구조-4회차 : list_basic.py
#
# [4-3-2] 파이썬 리스트 고급 기능
#=====
A = [1, 2, 3, 4]      # [1, 2, 3, 4]
print("A(original) = ", A)
A.append(7)
A.append(8)
A.append(9)
print("A	append*3) = ", A)

A.pop()
A.pop()
print("A( pop*2 ) = ", A)

A.insert(0, 0)
A.insert(2, 9)
print("A(insert*2) = ", A)
```

The project sidebar on the left lists various files and folders related to the course, such as '01회차-시작구조', '02회차- 알고리즘, 풀이문법', etc.

실습 단계

파이썬 리스트 고급기능_append: 항목을 리스트의 맨 뒤에 추가하는 연산

append(7), append(8), append(9) 출력

파이썬 리스트 고급기능_pop: 리스트의 맨 뒷단에 새롭게 추가하고 꺼내는 종류의 연산

시간 복잡도: O(1)

파이썬 리스트 고급기능_insert(0, 0)/ insert(2, 9) 항목 삽입

매개변수가 있는 경우: 매개변수 위치의 항목 삭제

최종 출력 결과 확인



배열과 파이썬 리스트

리스트

배열과 파이썬 리스트

3 파이썬 리스트의 용량 확장 방법

C언어의 배열은 선언할 때 크기가 고정

파이썬 리스트는 사용 중에 크기를 늘릴 수 있음

- append()연산, insert()연산 등

▶ 어떤 방법으로 늘릴 수 있을까?

리스트

배열과 파이썬 리스트

3 파이썬 리스트의 용량 확장 방법

● 용량 확장 아이디어

필요한 양(리스트의 크기)보다 넉넉한 크기의 메모리(리스트의 용량)를 사용

항목을 삽입할 때마다 남은 공간을 하나씩 사용

남은 공간이 없는 상태에서 삽입을 해야 하는 경우

- 용량을 증가시켜 남은 공간을 확보한 다음 항목을 삽입



배열과 파이썬 리스트

리스트

배열과 파이썬 리스트

3 파이썬 리스트의 용량 확장 방법

◎ 시간 복잡도

용량 확장 과정의 시간 복잡도

- 리스트의 항목 수가 n 이라면 $\rightarrow O(n)$
- 삽입 연산의 복잡도에 영향을 줌

append(e) 연산

- 대부분의 경우: 남은 공간이 있는 경우 $\rightarrow O(1)$
- 최악의 경우: 용량 확장이 필요한 경우 $\rightarrow O(n)$

insert(pos,e) 연산

- 원래 시간 복잡도가 $O(n)$