**데이터구조 및 프로그래밍실습 3분반**

**설계 프로젝트**

학번 : 202412590

이름 : 박찬희

본인의 Github 주소 : https://github.com/chanii47/data-structure-lab-2025.git

(본 과제의 코드를 본인의 Github에 업로드 하세요)

**문제 1.**

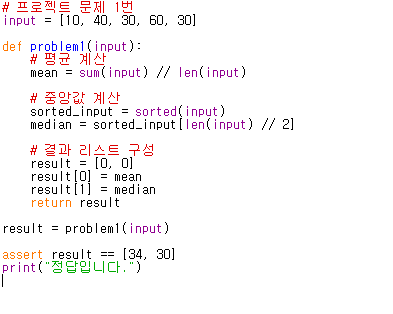
1. 본 문제를 풀기위해 사용한 데이터 구조를 제시하고 간단히 설명하세요.

- 리스트(List)

문제에서 주어진 5개의 자연수를 저장하고 처리하기 위해 파이썬의 기본 자료구조인 리스트 를 사용했다.

input 리스트: 5개의 숫자를 순서대로 저장  
sorted\_input 리스트: 입력 리스트를 정렬한 결과를 저장

리스트는 순서가 있고, 인덱스를 통해 직접 접근 가능하며, 정렬, 합산 등의 연산이 쉽기 때문에 대표값 계산에 적합함.

1. 본인이 작성한 파이썬 Code 캡처 이미지를 첨부하고 Algorithm Analysis를 수행하세요.   
   

- 시간 복잡도

평균 계산: sum(input)과 len(input)은 각각 O(n)과 O(1) (길이가 고정된 5개지만 일반화하면 n)  
중앙값 계산: sorted(input)은 O(n log n)

전체 시간 복잡도는 O(n log n) (n=5로 고정되어 있어 실제는 매우 빠름)

- 공간 복잡도  
입력 리스트를 정렬할 때 새로운 리스트(sorted\_input)를 생성하므로 추가 O(n) 공간 사용결과 리스트는 크기 2로 상수 공간 사용  
총 공간 복잡도는 O(n)

**문제 2.**

1. 본 문제를 풀기위해 사용한 데이터 구조를 제시하고 간단히 설명하세요.

- 문자열(String)

입력으로 주어진 괄호열은 문자열 형태로 처리함.  
문자열은 문자들의 순서 있는 나열로, 각 괄호 ( , )를 순차적으로 탐색하기 적합함.

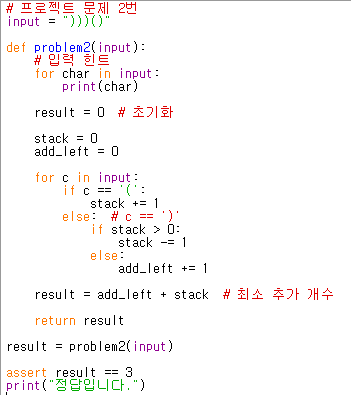
- 정수형 변수 (스택 역할 변수)

실제로 스택 자료구조를 사용하지 않고, 열린 괄호의 개수를 세기 위한 stack 변수를 사용

stack 변수는 현재까지 열린 괄호(()의 개수를 저장하며, 닫힌 괄호())가 나올 때마다 짝이 맞으면 1씩 줄임

닫힌 괄호가 너무 많아 열린 괄호가 없으면 추가해야 할 왼쪽 괄호 개수를 add\_left에 저장

1. 본인이 작성한 파이썬 Code 캡처 이미지를 첨부하고 Algorithm Analysis를 수행하세요.

  
- 시간 복잡도  
입력 문자열을 한 번 순회하면서 괄호 개수를 계산하기 때문에 O(n)

여기서 n은 입력 문자열의 길이  
  
- 공간 복잡도  
문자열 이외에 사용하는 변수는 상수 개수이므로, O(1)

**문제 3.**

1. 본 문제를 풀기위해 사용한 데이터 구조를 제시하고 간단히 설명하세요.

- 2차원 리스트 (리스트의 리스트)

입력으로 주어진 숲의 상태를 2차원 리스트 forest로 저장  
각 칸은 숫자(벌집 크기 또는 곰 위치)를 나타내며, 행과 열로 좌표화됨

- 변수 (정수형)

bear\_size: 곰의 현재 크기

honeycomb\_count: 곰이 먹은 벌집 수 (곰 크기 증가 조건에 사용)

time: 곰이 벌집을 찾아 이동한 총 시간(초)

bear\_x, bear\_y: 곰의 현재 위치 좌표

- 큐(Queue, deque)

BFS 탐색을 위해 collections.deque를 사용  
큐는 현재 위치에서 인접 칸으로의 이동을 순차적으로 처리하는 데 최적

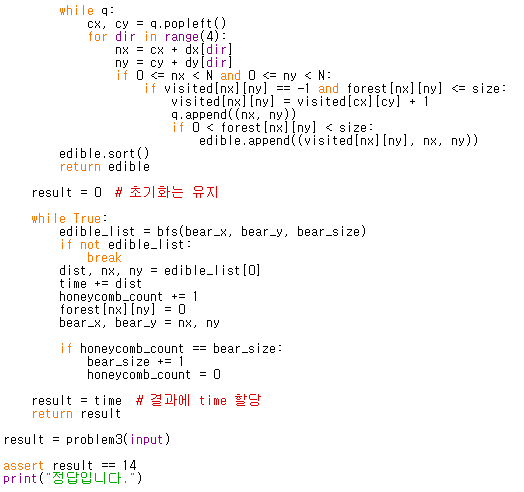
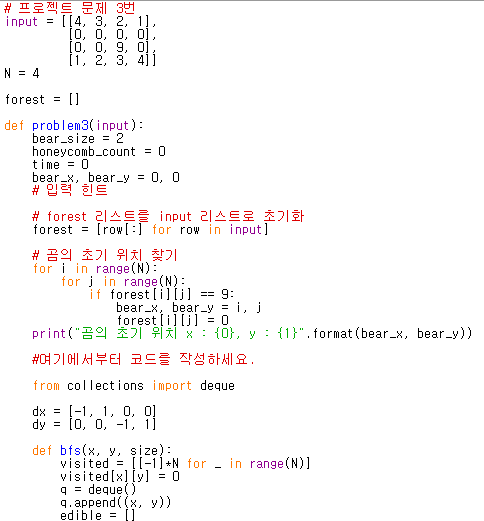
- 방문 기록용 2차원 리스트 (visited)

BFS 수행 중 방문 여부 및 거리 정보를 저장

- 리스트 (먹을 수 있는 벌집 후보 저장용)

BFS 탐색 중 먹을 수 있는 벌집들의 좌표와 거리 정보를 저장하며, 조건에 맞게 정렬하여 우선순위를 결정

1. 본인이 작성한 파이썬 Code 캡처 이미지를 첨부하고 Algorithm Analysis를 수행하세요.



- 시간 복잡도

BFS 탐색을 반복 수행하며, 한 번 BFS는 최대 O(N²) (숲 전체 칸 수)

최악의 경우 곰이 모든 벌집을 다 먹을 때까지 BFS를 여러 번 수행할 수 있으므로 전체 시간 복잡도는 대략 O(M \* N²) 이며, 여기서 M은 먹을 수 있는 벌집의 개수.

하지만 N이 최대 20이라 매우 작으므로 충분히 효율적  
  
- 공간 복잡도

2차원 리스트 forest와 방문 체크 리스트 visited 모두 크기가 N²이므로, O(N²)

추가적으로 큐와 후보 리스트 등이 있으나 이는 모두 O(N²)를 넘지 않음