**데이터구조 및 프로그래밍실습 3분반**

**설계 프로젝트**

학번 : 202017223

이름 : 채승민

본인의 Github 주소 : https://github.com/min011112

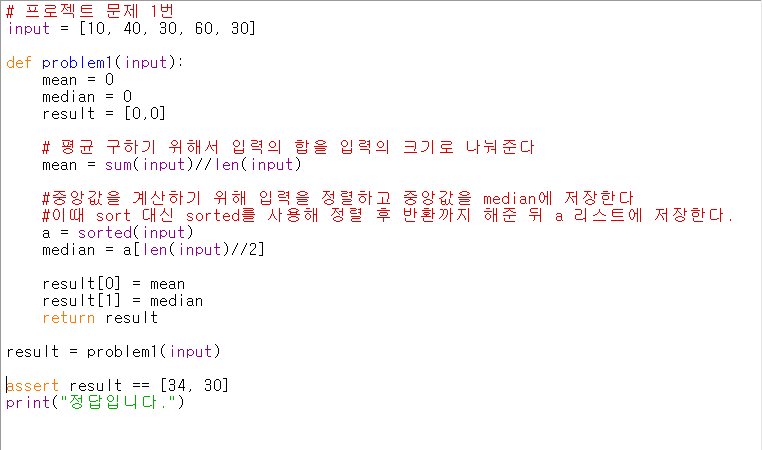
(본 과제의 코드를 본인의 Github에 업로드 하세요)

**문제 1.**

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

해당 문제를 풀기 위해서 List라는 데이터 구조를 사용하였다. List는 저장된 데이터에 순서를 부여해 관리하는 데이터 구조이다. List의 성질을 이용해 저장된 데이터 값의 총합을 계산해 평균을 구하였고 List를 정렬한 뒤 중앙값을 찾기 위해 가운데 순서에 위치한 값을 반환하여 문제를 해결하였다.

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



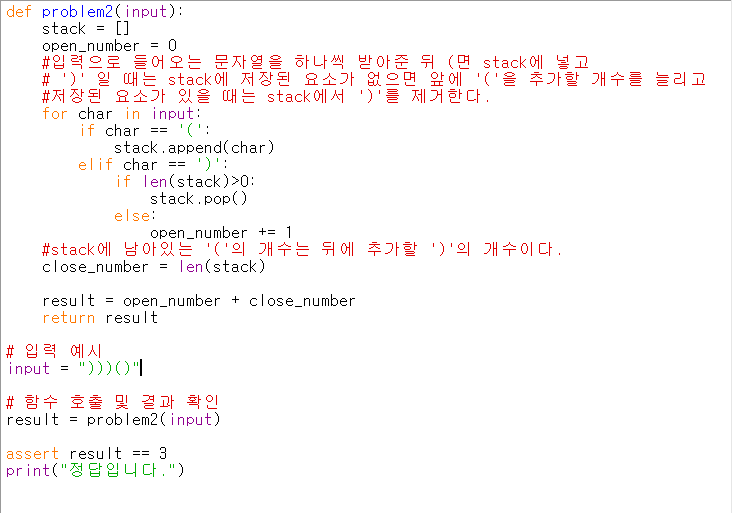
먼저 평균을 구하는 부분의 시간 복잡도를 보면 sum(input)이 리스트에 저장된 모든 값에 대해서 합 연산을 수행하기 때문에 시간 복잡도는 O(n)이고, 중앙값을 계산하는 연산을 보면 sorted 함수의 시간 복잡도는 O(nlogn)이고 중앙값을 찾는 연산은 1번의 계산만 필요하므로 O(1), 따라서 중앙값을 찾는 연산의 시간 복잡도는 O(nlogn)이다. O(n)보다 O(nlogn)이 더 우세하므로 해당 알고리즘의 시간 복잡도는 O(nlogn)이다.

**문제 2.**

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

괄호열을 복원하기 위해서는 열린 괄호가 나왔으면 닫힌 괄호가 뒤에 나와야 하고 가장 먼저 열린 괄호는 가장 마지막에 닫혀야 한다. 따라서 선입후출인 Stack을 사용하여 해당 문제를 해결하였다. Stack은 선입후출을 따르는 데이터 구조로 가장 마지막에 들어온 데이터가 항목의 맨 위에 추가되고 가장 먼저 제거되는 구조이다.

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



해당 문제를 해결하는 함수는 조건에 따라 pop과 append가 실행되는 반복문이다.

반복문 속 pop과 append는 리스트의 끝부분에서 연산되므로 시간 복잡도는 O(1)이고 반복문은 input의 크기만큼 반복되기 때문에 반복문 전체의 시간 복잡도는 O(n)이다. 따라서 해당 함수의 시간 복잡도는 O(n)이다.

**문제 3.**

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

곰이 이동할 때는 항상 최단 거리로 이동해야 하므로 DFS와 BFS 중 항상 최단 거리를 찾을 수 있는 BFS를 이용해야 한다. BFS는 너비 우선 탐색 알고리즘으로 그래프를 탐색할 때 시작 노드에서 갈 수 있는 노드들을 우선으로 탐사한 후 그 다음 깊이로 넘어가는 기법이기 때문에 선입선출 방식을 사용하는 Queue를 이용하여 구현한다. Queue는 선입선출 방식을 사용하는 데이터 구조로 가장 먼저 들어온 데이터가 항목의 맨 위에 위치해 제거될 때도 먼저 제거되는 구조이다.

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



알고리즘 분석을 해보면 먼저 곰의 초기 위치를 찾는 연산은 N\*N번 이루어지므로 해당 연산의 시간 복잡도는 O()이다. BFS을 수행하는 함수의 연산은 전체 노드에 대해서 방문할 수 있는 노드를 찾고 찾은 노드들을 queque에 넣는 연산이기 때문에 총 개의 노드에 대하여 탐색을 실시하므로 해당 함수의 시간 복잡도는 O()이다. 마지막으로 전체 함수의 시간 복잡도를 계산해보면 연산이 가장 많은 최악의 경우는 모든 칸이 곰이 먹을 수 있는 벌집인 경우이다. 이때는 BFS 함수를 번 호출하고 BFS 함수의 시간 복잡도는 O()이므로 전체 함수의 최악의 경우의 시간 복잡도는 O()이다. 따라서 해당 알고리즘의 시간 복잡도는 O()이다.