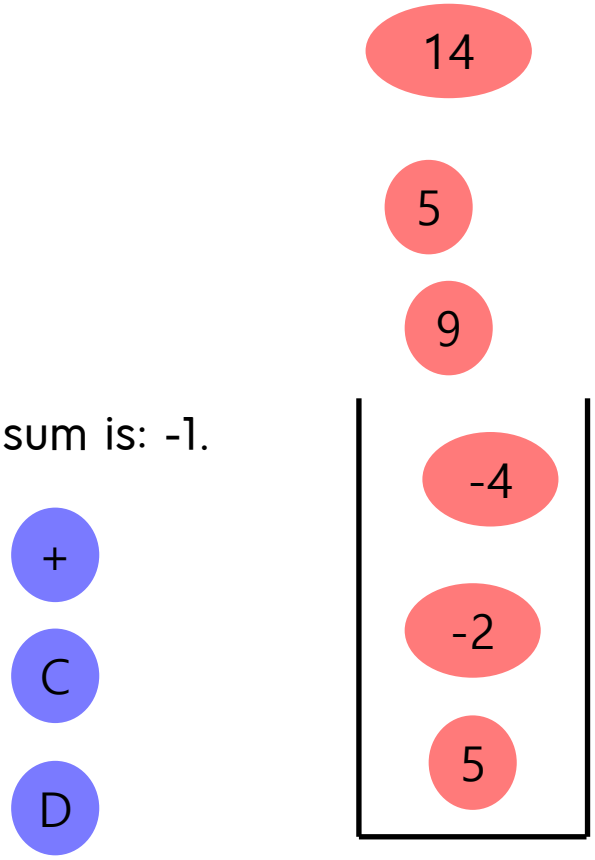
Baseball Game

Problem

Input: ["5","-2","4","C","D","9","+","+"]
Output: 27
1: The sum is: 5.
2: -2 points. The sum is: 3.
3: 4 points. The sum is: 7.
C: 3번 데이터 삭제. The sum is: 3.
4: 2번의 -2 값 더블 -4을 얻는다 -4+3=-1. The sum is: -1.
5: 9 points. The sum is: 8.
6: -4 + 9 = 5 points. The sum is 13.
7: 9 + 5 = 14 points. The sum is 27.

Soution



Solution

BFS

1. 문제를 정확히 이해
2. 알고리즘 정하고 담을 그릇 정한다
3. for 문 돌리기
4. 생각->프로그램(한국말로 생각하고->Java)
   결과를 해석하여 이미지화시킨다

I Can Image

유효한 괄호(Valid Parentheses)

설명

String s가 주어집니다. s는 '(', ')', '{', '}', '[' ']' 로 이루어집니다.
유효한 괄호인지 체크하여 boolean값으로 리턴하세요.

입력 문자열은 다음과 같은 경우에 유효합니다.
1. 열린 괄호는 동일한 유형의 괄호로 닫아야합니다.
2. 열린 괄호는 올바른 순서로 닫아야합니다.

입출력

Input: s = "()"
Output: true

Input: s = "()[]{}"
Output: true

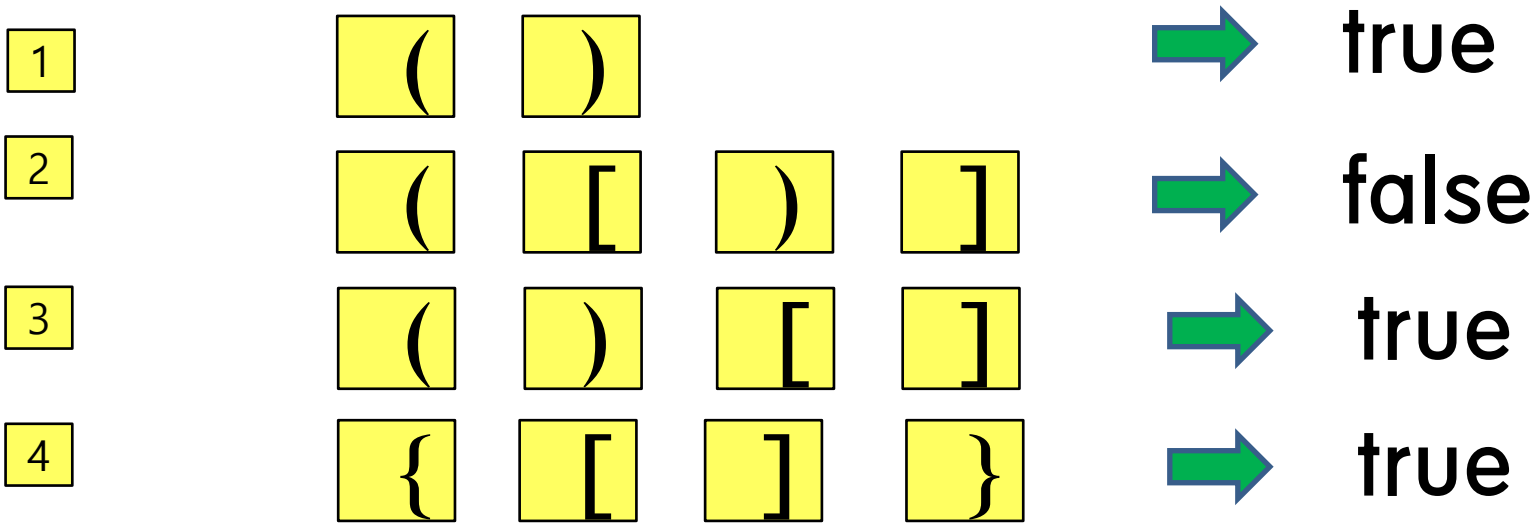
Input: s = "([)]"
Output: false

Input: s = "{[]}"
Output: true

제한사항

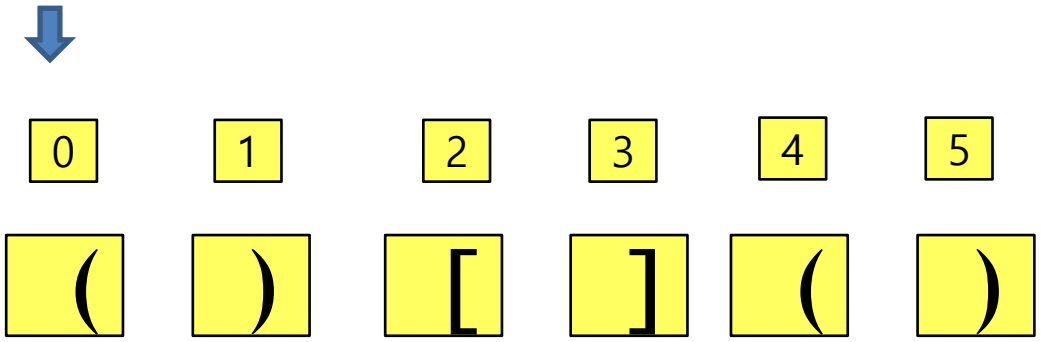
1 <= s.length <= 10^4
s consists of parentheses only '()[]{}'.

문제분석



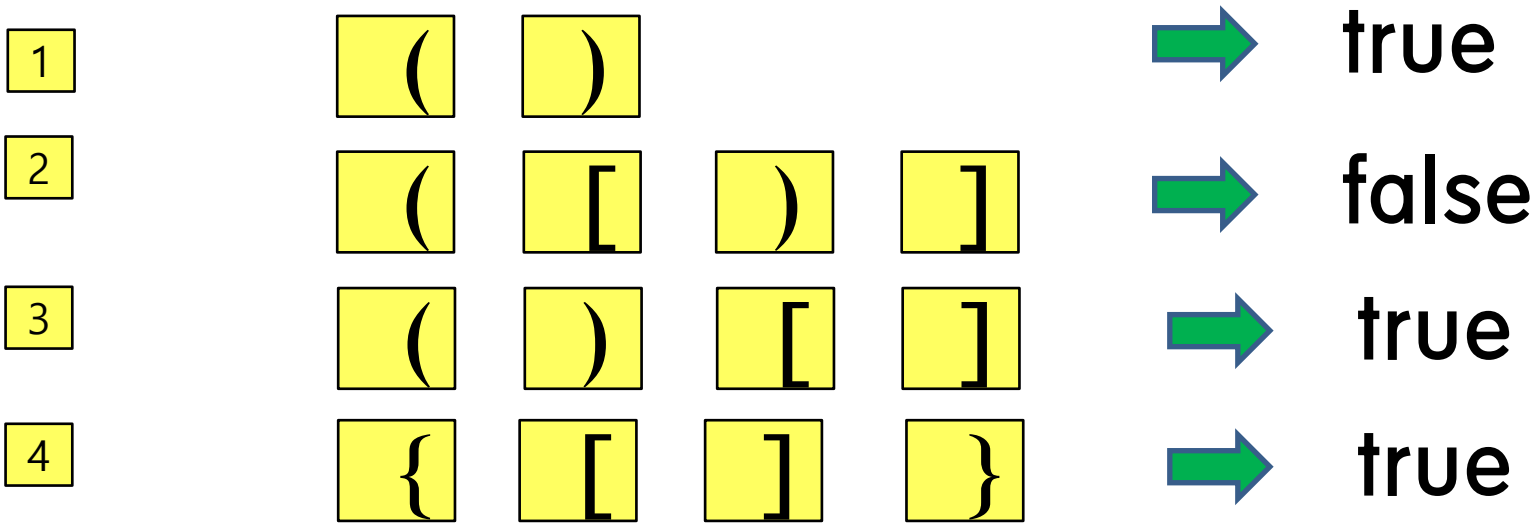
1. 2번 케이스 순서 불일치

문제분석



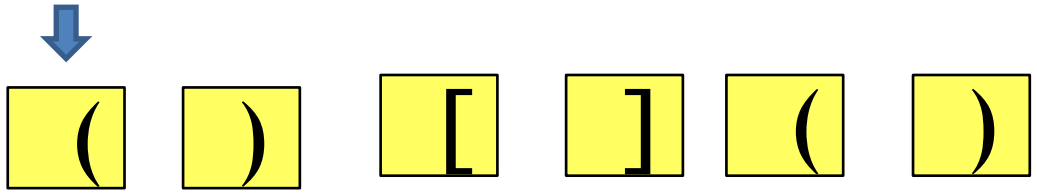
1. 열린괄호, 닫힌 괄호 개수를 이용
2. ( ) 열린괄호 2개 닫힌괄호 2개
3. [ ] 열린괄호 1개 닫힌괄호 1개
4. Count=0을 이용해서 카운트를 하고 0이 아니면 false
5. 순서고려 중요

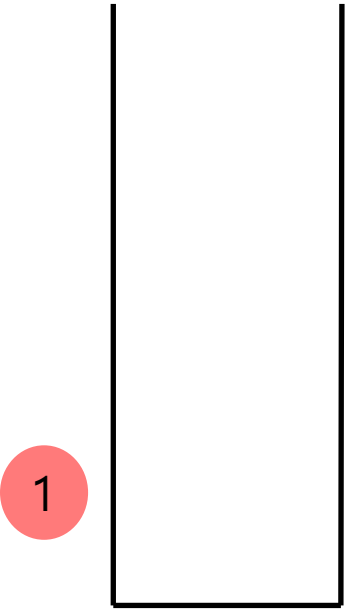
문제분석



1. 괄호 문제는 무조건 스택이용
2. 스택 LIFO 기능 이용

코딩화





push(), peek(), pop()
스택 이용

시간복잡도/공간복잡도 계산

시간복잡도

1. 대상(Source) : 문제에서 입력받은 파라미터(array 등) (속도)
   Time Complexity : O(N)
   대상 : String s
   이유 : for문 한번 실행

공간복잡도

2. 대상(Source) : 실제 사용되는 저장 공간을 계산(메모리 사용량)
   예) 프로그램을 실행 및 완료하는데 필요한 저장공간
   Space Complexity : O(N)
   대상 : Stack<Character> stack = new Stack<>();
   이유 : 스택에 s의 길이만큼 push. pop

참고

O(1) : 상수
O(n) : for문 => 데이터를 한번씩 다 호출하니까 (제일 많음)
O(log N) : sort, prirotiyQueue, binary Search Tree, Tree
O(Klog N) : k번만큼 소팅하는 경우
O(n^2) : 이중for문
O(m*n) : 이중for문인데, n이 다른경우bfs,dfs 류 (예 n=100 인데 m=5인경우)

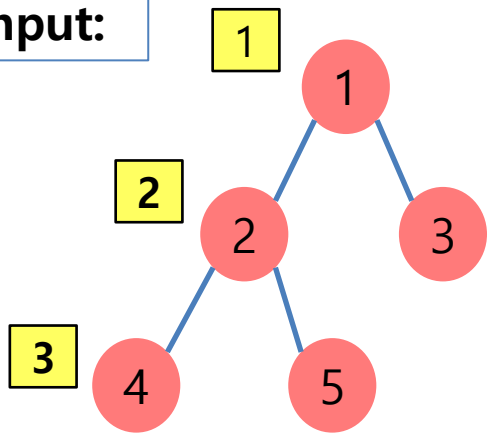
이진 트리 레벨 순서 순회 (BinaryTree Level Order)

설명

이진 트리가 주어지면 노드 값의 레벨 순서 순회를 반환 합니다 .
(즉, 왼쪽에서 오른쪽으로, 레벨별로)

입출력

Input:



Output: [[1],[2,3],[4,5]]

참고)

```
class TreeNode{
        int val;
        TreeNode left, right;
        TreeNode(int x){
                this.val = x;
        }
}
```

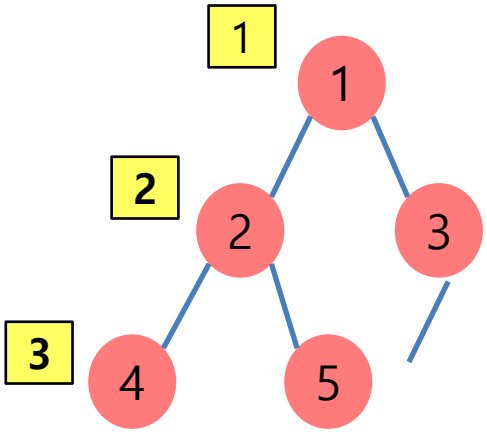
제한사항

The number of nodes in the tree is in the range [0, 2000].
-1000 <= Node.val <= 1000

BinaryTree Level Order

Problem

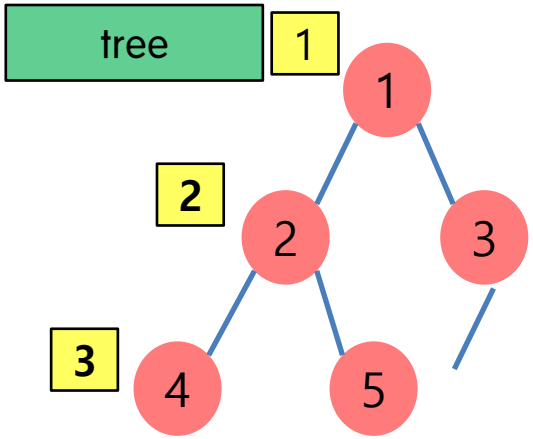
Input:

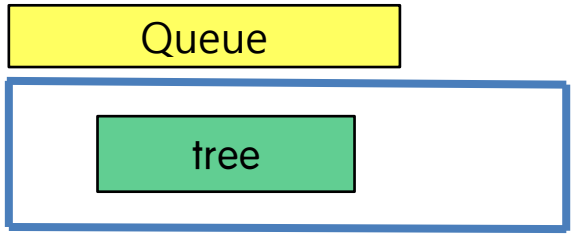


Output: [[1],[2,3],[4,5]]

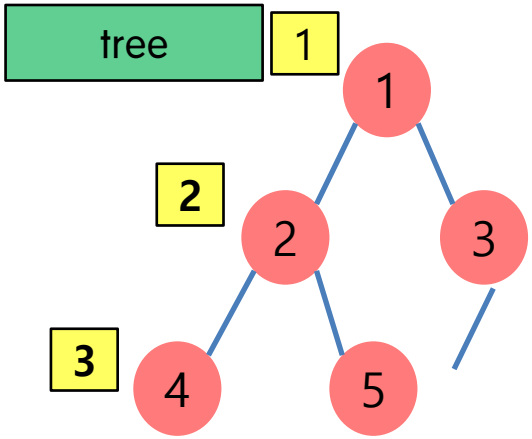
Queue bfs
Stack dfs

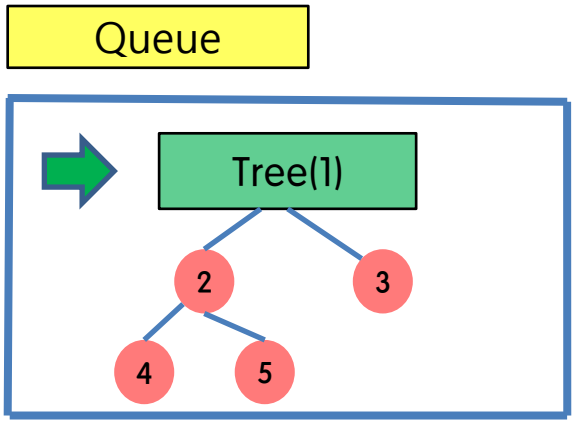
BinaryTree Level Order



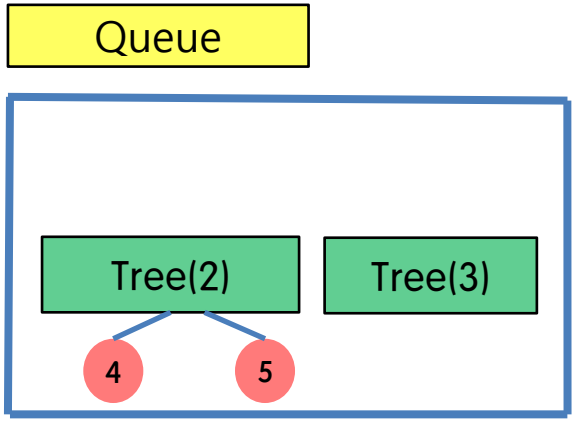


BinaryTree Level Order



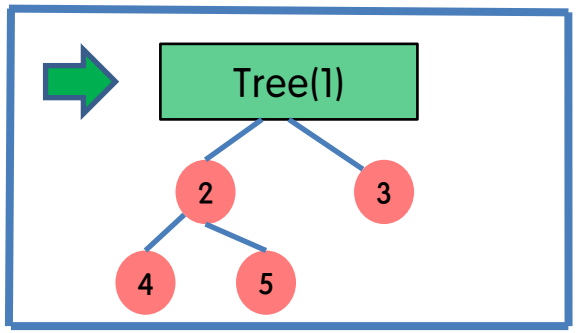


List [1]



List [2,3]

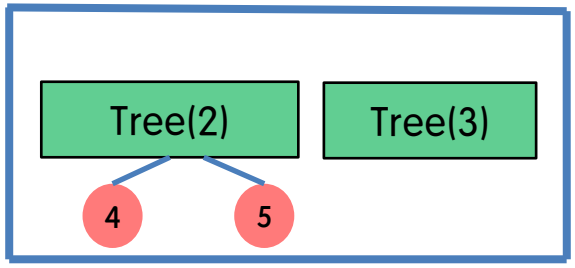
Queue



List [1]

List<List [1]>

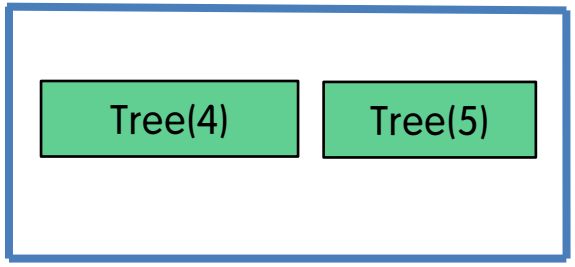
Queue



List [2,3]

List<List [1], [2,3]>

Queue



List [4,5]

List<List [1], [2,3], [4,5]> >