Stack Data Structure

- Linear data structure
- LIFO (Last in First Out)

Operations :-

- Push - Add a item, if stack is full (Overflow).
- Pop - Remove a item, if empyt (Underflow).
- Peek or Top - Returns top element of stack.
- isEmpty - True if empty, else False.
- is Full - True if full, else Empty.

Implementations :

There are two ways :-

- Using array
- Using Linked list

Application :-

- Balancing of symbols.
- Infix to postfix / Prefix conversion.
- Forward & backword in web browser.
- Used in many algorithms.

Balance Bracket :-

Suppose your bracket string is

Stack = " [[[( ) { [ ] }]]] "
1 2 3 4 5 6 7 8 9 10

Opening = " ([{ "
Closing = ")] } "
Matching = { " } " : " { " , "] " : " [" , ") " : " (" }
a A b B c C

1 If char in Opening
stack.append(char)

{char - individual bracket of string bracket}

So our stack is stack = [[[( ]
1 2 3

elif :

2. If char in Closing
) is in closing
4

if stack[-1] == Matching[char] :

↓

Last element of stack " ["
3

") " = " ("
4 c C

→ In dict of Matching

So They are equal.

stack.pop()

Else

False

3. If len(stack) == 0
return True