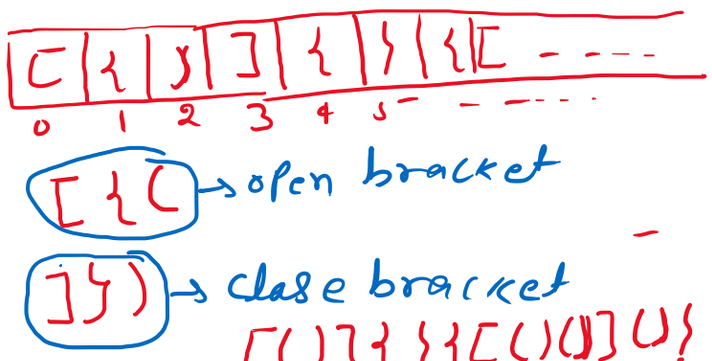
Check for Balanced Brackets in an expression

Stack (Collection object)

Input: exp = “[()]{}{[()()]()}”



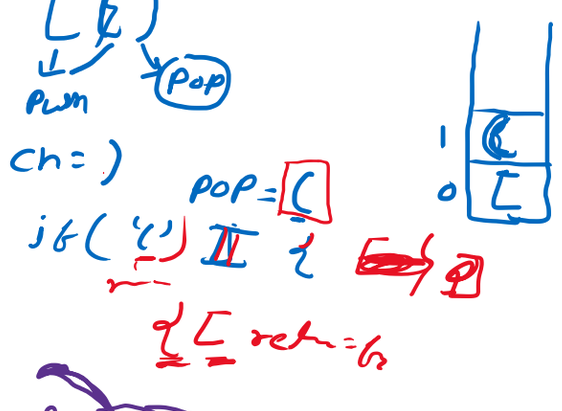
Declare a character stack (say temp).
Now traverse the string exp.
If the current character is a starting bracket (‘(‘ or ‘{‘ or ‘[‘) then push it to stack.
If the current character is a closing bracket (‘)’ or ‘}’ or ‘]’) then pop from the stack and if the popped character is the matching starting bracket then fine.
Else brackets are Not Balanced.
After complete traversal, if some starting brackets are left in the stack then the expression is Not balanced, else Balanced.

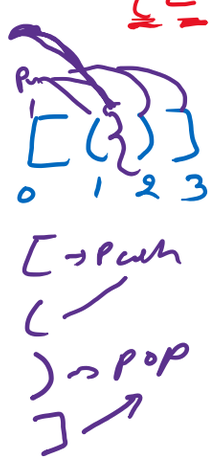
```
public boolean balancedSymbol(String str){
    Stack<Character> stack = new Stack();
    for(int i=0; i<str.length(); i++){
        char ch = str.charAt(i);
        if(ch=='(' || ch=='[' || ch=='{'){
            stack.push(ch);
            continue;
        }
        char pop = '0';
        switch(ch){
        case ')':
            pop = stac.pop();
            if(pop=='{' || pop=='['){
                return false;
            }
            break;
        case '}':
            pop = stack.pop();
            if(pop=='(' || pop=='['){
                return false;
            }
            break;
        case ']':
            pop = stack.pop();
            if(pop=='(' || pop=='{'){
                return false;
            }
            break;
        }
    }//end of for.
    return stack.isEmpty();
}
```

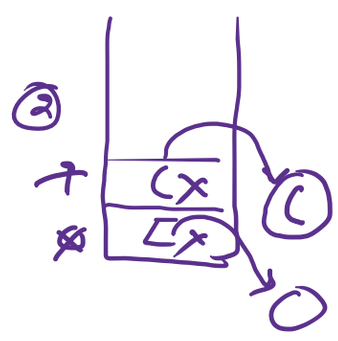
([{ → Push in to stack

continued

) }] → pop → { ([

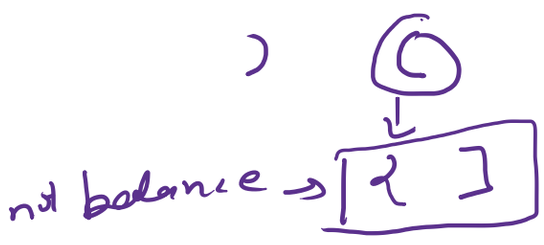






true → Balanced

false → not Balanced



([{ → Push

)] } → pop [( )] → str.

(A+B) - (C·D) ⇒

(+AB) - (-CD)

- +AB-CB ⇒ infix

```
public static boolean balanceSymbol(String str) {
        Stack<Character> stack = new Stack<>();
        for(int i=0;i<str.length();i++) {
            char ch = str.charAt(i);
            if(ch == '(' || ch == '{' || ch == '[') {
                stack.push(ch);
                continue;
            }
            char popped = '0';
            switch(ch) {
            case ')':
                popped = stack.pop();
                if(popped == '{' || popped == '[') {
                    return false;
                }
                break;

            case '}':
                popped = stack.pop();
                if(popped == '(' || popped == '[') {
                    return false;
                }
                break;

            case ']':
                popped = stack.pop();
                if(popped == '(' || popped == '{') {
                    return false;
                }

            }
            }
            return stack.isEmpty();
    }
```