Question: https://leetcode.com/problems/minimum-remove-to-make-valid-parentheses/

So anytime we need to check a parenthesis related problem the first ds that strikes my mind is stack, because we can keep track of opening parenthesis.

So I did something similar here, i.e., I kept track of position of opening parenthesis in the result String Builder object and while closing a parenthesis checked if there was a opening parenthesis for it or not, if then we add the closing parenthesis else we ignore it.

Now there can be a case that we added the opening parenthesis and there are no closing parenthesis, in that case our stack wont be empty. It will be having the position of opening parenthesis without a closing one in result, so we would remove those opening parenthesis.

Code:  
class Solution {

public String minRemoveToMakeValid(String s) {

Stack<Integer> openParenthesis = new Stack<Integer>();

StringBuilder res = new StringBuilder();

for(int i=0; i<s.length(); i++){

char c = s.charAt(i);

if(c=='('){

res.append(c);

openParenthesis.push(res.length()-1);

}

else if(c==')' && !openParenthesis.isEmpty()){

res.append(c);

openParenthesis.pop();

}

else if(c!=')'){

res.append(c);

}

}

while(!openParenthesis.isEmpty()){

res.deleteCharAt(openParenthesis.pop());

}

return res.toString();

}

}

Github Link :<https://lnkd.in/ecwtJeaz>