227. Basic Calculator II

link

自己写了一个用stack先算加减乘除, 最后倒叙算结果的, 注意最后1-1+1在stack里面会变成 `[1, +, 1, -, 1]` 每次pop两个然后加起来

```
    Deque<String> stack = new ArrayDeque<>();
    int i = 0;
    StringBuilder sb = new StringBuilder();
    while(i < s.length()){
        if(s.charAt(i) == ' '){
            i++;
            continue;
        }
        while(i < s.length() && Character.isDigit(s.charAt(i))){
            sb.append(s.charAt(i));
            i++;
        }
        if(sb.length() > 0){
            String num = sb.toString();
            if(!stack.isEmpty() && (stack.peek().equals("*") ||
stack.peek().equals("/"))){
                String operator = stack.pop();
                String first = stack.pop();
                if(operator.equals("*")){
                    num = String.valueOf(Integer.parseInt(first) *
Integer.parseInt(num));
                }else{
                    num = String.valueOf(Integer.parseInt(first) /
Integer.parseInt(num));
                }
            }
            stack.push(num);
            sb.setLength(0);
        }else{
            stack.push(String.valueOf(s.charAt(i++)));
        }
    }
    if(stack.size() == 1)
        return Integer.parseInt(stack.pop());
    int res = 0;
    while(stack.size() > 1){
        String num = stack.pop();
        String operator = stack.pop();
        if(operator.equals("+")){
            res += Integer.parseInt(num);
        }else{
            res += -Integer.parseInt(num);
        }
    }
    return res + Integer.parseInt(stack.pop());
```

}

没有用stack版本的需要学习的 `1 + 2 * 3 + 3 * 4`

1. `num = num * 10 + c - '0'`来计算num

   1.2 每次计算前一个的operator 默认1之前有一个 `0 +`

   1.3 第一次1后面的+， `sum = 0, tmpSum = 1`，算到2后面的*就把之前的加起来，只有遇到*活着/ 才把之前的加上

```
public int calculate(String s) {
    int sum = 0;
    int tempSum = 0;
    int num = 0;
    char lastSign = '+';
    for (int i = 0; i < s.length(); i++) {
        char c = s.charAt(i);
        if (Character.isDigit(c)) num = num * 10 + c - '0';
        if (i == s.length() - 1 || !Character.isDigit(c) && c!=' ') {
            switch(lastSign) {
                case '+':
                    sum+=tempSum;
                    tempSum = num;
                    break;
                case '-':
                    sum+=tempSum;
                    tempSum = -num;
                    break;
                case '*':
                    tempSum *= num;
                    break;
                case '/':
                    tempSum /= num;
                    break;
            }
            lastSign = c;
            num=0;
        }
    }
    sum+=tempSum;
    return sum;
}
```