/\*\*

 \* @param {string} s

 \* @return {number}

 \*/

var calculate = function(s) {

    let result = 0;

    let num = 0;

    let sign = 1; // 1表示正号，-1表示负号

    const stack = [];

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

        const char = s[i];

        if (char >= '0' && char <= '9') {

            // 处理多位数字

            num = num \* 10 + parseInt(char);

        } else if (char === '+' || char === '-') {

            // 遇到运算符，先计算前面的表达式

            result += sign \* num;

            num = 0;

            sign = char === '+' ? 1 : -1;

        } else if (char === '(') {

            // 遇到左括号，将当前结果和符号压入栈

            stack.push(result);

            stack.push(sign);

            result = 0;

            sign = 1;

        } else if (char === ')') {

            // 遇到右括号，先计算括号内的表达式

            result += sign \* num;

            num = 0;

            // 然后与栈中的结果合并

            result \*= stack.pop(); // 弹出符号

            result += stack.pop(); // 弹出之前的结果

        }

        // 忽略空格

    }

    // 处理最后一个数字

    result += sign \* num;

    return result;

};

