**初审**

1. **用户需求分析**：已另附稿。
2. **整体函数代码初步审查简述：**

此部分由本结对小组在共同理解代码并试运行后所总结得出，如下表：

|  |  |  |
| --- | --- | --- |
| 函数定义 | 功能理解 | 初步审查 |
| static String  getAnswer(String equation) | 获取中序表达式的计算结果 | √ |
| String AnswerFromStack(Stack<String> RPNOfEqu) | 获取逆波兰表达式的计算结果 | 在个别次数运行时会出现空栈的异常报错 |
| String add(String a, String b) | 将两个数据相加 | √ |
| String minus(String a, String b) | 将两个数据相减 | √ |
| String multiply(String a, String b) | 将两个数据相乘 | √ |
| String divide(String a, String b) | 将两个数据相除 | √ |
| static boolean contain(ArrayList<String> equList, String equ) | 判断已生成的算式List中是否包含与此式重复的算式 | √ |
| static boolean  isRepeated(String equ1, String equ2) | 通过运算优先级判断两个算式（中序表达式）是否重复  规定：只有运算顺序一致才判定为重复，如"(1+2)\*(2+2)","(2+2)\*(2+1)"为不同算式 | √ |
| static Stack<String> transform(Stack<String> RPNOfEqu) | 将逆波兰式按优先级降序的方式重新排列，并且加和乘（满足交换律）的两个被操作数按升序的方式排列 | √ |
| ArrayList<String>  getEquations() | 获取所需个数（作为类成员变量传入）的算式，并存入一个List | 出现两个运算符的概率远大于其他个数的运算符 |
| String tempEqu(int opeNum) | 生成一个符合数据要求且不含括号的中间式 | √ |
| String insertBrackets(String tempEqu) | 对不含括号的临时式进行插入括号操作，控制括号内必须含有加减 | 满足控制插入括号的条件情况过少 |
| String getFraction() | 获取分数的字符串表达 | √ |
| boolean isFalseFraction() | 判断是否为假分数 | √ |
| static boolean isFraction(String s) | 判断是否是分数 | √ |
| static Fraction transform(String a) | 将整数转化为分母为1的分数，便于运算 | √ |
| static boolean isDigit(String num) | 判断字符串是否表达整数 | √ |
| static int priority(String/char operator) | 获取输入符号的优先级，  满足两种类型的输入 | √ |
| static boolean validEquation(String tempEqu) | 判断当前算式是否合法 | √ |
| static boolean validAnswer(int max, int min, String answer) | 判断计算结果是否合法 | √ |
| static Stack<String> transformToRPN(String equation) | 将中序表达式转换成逆波兰表达式 | √ |
| static ArrayList<String> toStringArray(String equation) | 将算式中的数字和运算符分开存放为数组 | √ |