面向对象设计与构造第一次作业

第一部分：训练目标

通过对数学意义上的表达式结构进行建模，完成单变量多项式的括号展开，初步体会层次化设计的思想的应用和工程实现。

第二部分：预备知识

- 1、Java 基础语法与基本容器的使用。
- 2、扩展 BNF 描述的形式化表述。
- 3、正则表达式、递归下降或其他解析方法。

第三部分：题目描述

本次作业需要完成的任务为：读入一个包含加、减、乘、乘方以及括号（其中括号的深度**至多为 1 层**）的**单变量**表达式，输出**恒等变形展开所有括号后**的表达式。

在本次作业中，**展开所有括号**的定义是：对原输入表达式 *E* 做**恒等变形**，得到新表达式 *E'*，且 *E'* 中不含有字符 `(` 、 `)` 和空白字符 。

第四部分：基本概念

一、基本概念的声明

- **带符号整数 支持前导 0** 的**十**进制带符号整数（若为正数，则正号可以省略），无进制标识。如：`+02`、`-16`、`20220928` 等。
- **因子**
  - **变量因子**
    - 幂函数
      - **一般形式** 由自变量 x，指数符号 `^` 和指数组成，指数为一个**非负**带符号整数，如：`x ^ +2`,`x ^ 02`,`x ^ 2` 。
      - **省略形式** 当指数为 1 的时候，可以省略指数符号 `^` 和指数，如：`x` 。
  - **常数因子** 包含一个带符号整数，如：`233` 。
  - **表达式因子** 用一对小括号包裹起来的表达式，可以带指数，且指数为一个**非负**带符号整数，例如 `(x^2 + 2*x + x)^2` 。表达式的定义将在表达式的相关设定中进行详细介绍。
- **项** 由乘法运算符连接若干因子组成，如 `x * 02` 。此外，**在第一个因子之前，可以带一个正号或者负号**，如 `+ x * 02`、`- +3 * x` 。注意，**空串不属于合法的项。**
- **表达式** 由加法和减法运算符连接若干项组成，如：`-1 + x ^ 233 - x ^ 06 +x` 。此外，**在第一项之前，可以带一个正号或者负号，表示第一个项的正负**，如：`- -1 + x ^ 233`、`+ -2 + x ^ 19911226` 。注意，**空串不属于合法的表达式。**
- **空白字符** 在本次作业中，空白字符包含且仅包含空格 `<space>`（ascii 值 32）和水平制表符 `\t`（ascii 值 9）。其他的空白字符，均属于非法字符。

对于空白字符，有以下几点规定：

- 带符号整数内不允许包含空白字符，注意**符号与整数之间**也不允许包含空白字符。
- 因子、项、表达式，在不与前两条条件矛盾的前提下，可以在任意位置包含任意数量的空白字符。

二、设定的形式化表述

- 表达式 → 空白项 [加减 空白项] 项 空白项 | 表达式 加减 空白项 项 空白项
- 项 → [加减 空白项] 因子 | 项 空白项 '*' 空白项 因子
- 因子 → 变量因子 | 常数因子 | 表达式因子
- 变量因子 → 幂函数
- 常数因子 → 带符号的整数
- 表达式因子 → '(' 表达式 ')' [空白项 指数]
- 幂函数 → 'x' [空白项 指数]
- 指数 → '^' 空白项 ['+'] 允许前导零的整数 **(注：指数一定不是负数)**
- 带符号的整数 → [加减] 允许前导零的整数
- 允许前导零的整数 → ('0'|'1'|'2'|...|'9'){'0'|'1'|'2'|...|'9'}
- 空白项 → {空白字符}
- 空白字符 → （空格） | `\t`
- 加减 → '+' | '-'

其中

- `{}` 表示允许存在 0 个、1 个或多个。
- `[]` 表示允许存在 0 个或 1 个。
- `()` 内的运算拥有更高优先级，类似数学中的括号。
- `|` 表示在多个之中选择一个。
- 上述表述中使用单引号包裹的串表示字符串字面量，如 '(' 表示字符 `(`。

式子的具体含义参照其数学含义。

若输入字符串能够由“表达式”推导得出，则输入字符串合法。具体推导方法请参阅 **“第一单元形式化表述说明”文档。**

除了满足上述形式化表述之外，我们本次作业的输入数据的**额外限制**请参见**第五部分：输入/输出说明 的数据限制部分。**

第五部分：输入/输出说明

一、公测说明

输入格式

本次作业的输入数据**仅包含一行**，表示待展开括号的表达式。

输出格式

输出展开括号之后的表达式。

数据限制

- 输入表达式**一定满足**基本概念部分给出的**形式化描述**。
- 输入表达式中**至多包含1层括号**。
- 对于规则 “指数 → ^ 空白项 带符号的整数”，我们保证**此处的带符号整数中不会出现 - 号**，且保证**输入数据的指数最大不超过 8**。
- 在表达式化简过程中，如果遇到了需要计算 `0^0` 的值进行化简的这种情况，默认 `0^0 = 1`。
- 输入表达式的**有效长度**至多为 200 个字符。其中输入表达式的**有效长度**指的是输入表达式去除掉所有**空白符**后剩余的字符总数。
- 根据文法可以注意到，整数的范围并不一定在 `int` 或 `long` 范围内。

判定模式

本次作业中，对于每个测试点的判定分为**正确性判定**和**性能判定**。其中，正确性判定总分为 85 分，性能判定总分为 15 分，本次作业得分为二者之和。

注意：**获得性能分的前提是，在正确性判定环节被判定为正确**。如果被判定为错误，则性能分为0分。

正确性判定：

- 输出的表达式须符合表达式的**形式化描述**、**不能包含括号**和空白字符且与原表达式**恒等**。
- 本次作业中对于恒等的定义：设 *f(x)* 的定义域为 *D1*，*D1 包含于 R*，*g(x) 的定义域为 D2*，*D2 包含于 R*，对任意 x∈D1∩D2，f(x)=g(x) 成立。

性能判定：

- 在本次作业中，性能分的唯一评判依据是**输出结果的有效长度**，有效长度的定义在**数据限制部分**已经给出。

设某同学给出的**正确答案**的有效长度为 L_p，目前**所有人**给出的正确答案中有效长度**最小的**为 L_{min}。

记 $x = \frac{L_p}{L_{min}}$，则该同学**性能分百分比**为：

$$r(x) = 100\% \cdot \begin{cases} 1 & x = 1 \\ -31.8239x^4 + 155.9038x^3 - 279.2180x^2 + 214.0743x - 57.9370 & 1 < x \le 1.5 \\ 0 & x > 1.5 \end{cases}$$

举例来说，就是这样：

x	***r(x)***
1.0	100.0%
1.05	79.9%
1.1	60.5%
1.2	29.0%

x	*r*(*x*)
1.3	10.9%
1.4	4.5%
1.5	0.0%

该答案得到的性能分即为 r(x)×15。

二、互测说明

互测时，你可以通过提交**输入数据**和**预期正确输出**，该组数据会被用来测试同一个互测房间中的其他同学的程序。输入数据必须符合上述的文法规则,并且满足代价函数要求。提交的预期输出只需要包含一行。

数据限制

- 输入表达式**一定满足**基本概念部分给出的**形式化描述。**
- 输入表达式中**至多包含1层括号。**
- 对于规则 “指数 →→ ^ 空白项 带符号的整数”，保证**此处的带符号整数中不会出现 - 号**，且保证**输入数据的指数最大不超过 8。**
- 输入表达式的**有效长度**至多为 **50** 个字符。其中输入表达式的**有效长度**指的是输入表达式去除掉所有**空白符**后剩余的字符总数。（**本条与公测部分的限制不同**）
- 除此之外，为了限制不合理的 hack，我们要求输入表达式的代价 `Cost(Expr) <= 10000`，其中表达式代价的计算方法如下（**本条与公测部分的限制不同**）：

代价函数

- `Cost(常数) = max(1, len(常数))`（常数的前导零不算在其长度内）
- `Cost(x) = 1`
- `Cost(a + b) = Cost(a - b) = Cost(a) + Cost(b)`
- `Cost(a * b) = Cost(a) * Cost(b)`（多项相乘时从左到右计算）
- `Cost(a ^ b) =`
  - 若a是单变量因子，`Cost(a ^ b) = 1`
  - 若a是表达式因子`(c)`，`Cost(a ^ b) = max(Cost(c), 2) ^ max(b,1)`
- `Cost(+a) = Cost(-a) = Cost(a) + 1`

如果提交的数据不满足上述数据限制，则该数据将被系统拒绝，且不会用来对同屋其他被测程序进行测试。

规则简介

以下仅为简要介绍。如需深入了解互测的具体规则及细节，请**务必参阅课程所提供的规则文件，并以该文件作为最终依据。**

（一）互测安排

- 前置条件：互测屋分配完成（强测结束后）。

（二）互测屋机制

1. 分组规则：
    - 互测屋分为A、B、C三档，依据强测结果（功能分 + 性能分）分组。
    - 进入A档需通过[A_criterion, 100%]测试用例；B档需通过[B_criterion, A_criterion)；C档需通过[C_criterion, B_criterion)。
2. 测试机制：
    - 有效样例（非invalid）用于测试屋内所有代码，但不包括提交者本人（禁止自刀）。
3. 信息公开机制：
    - 身份信息：互测结束一周后公开。
    - 测试信息：互测期间定期公布有效样例、有价值样例及发现的bug总数；互测结束后公布成功hack的样例信息。
4. 互测积分：
    - base分：满分10分，具体计算方式见规则文件。
    - bonus分：满分30分，发现bug数超过30时按30分计算，同质bug按1个计算。
5. 恶意hack惩罚机制：
    - 对同质bug进行过多hack将受到惩罚，具体惩罚措施根据hack次数从扣分到取消作业得分不等。

（三）测试用例提交细则

1. 提交内容：测试用例输入 + 期望输出，期望输出需与评测机标程输出一致，否则视为无效。
2. 测试用例分类：
    - invalid：不符合规范或期望输出与标程不一致。
    - naive：有效但未发现bug。
    - valuable：有效且发现至少一个bug。
3. 提交限制：两次提交之间有冷却时间（CD），且每次作业有输入数据量限制。
4. 同质测试用例：
    - 多个测试用例攻击同一bug视为同质，bonus分按一个bug计算。
    - 具体分数计算根据bug修复情况调整。

（四）互测过程中的交互

1. 举报：
    - 互测期间接受实名举报作弊行为，包括代码标记、抄袭等，一经核实将严肃处理。
2. bug申诉：
    - 互测结束后两天内可向助教申诉，申诉需附录屏和详细原因，申诉失败累计达到5次后将失去申诉机会。
3. 答疑：
    - 互测期间所有提问应在讨论区进行，助教不接受私人答疑，最终答案以课程组公告为准。

三、样例

#	输入	输出(省略标签)	说明
1	1	1	根据表达式定义可得。
2	x	x	根据表达式定义可得。
3	x+2*x	x+2*x	未合并同类项，但表达式依然等价。
4	x+2*x+x^2	3*x+x^2	在表达式等价的基础上合并同类项。
5	-3*(x)	-3*x	拆括号后得到符合输出格式的表达式。
6	-3*(x-1)	-3*x--3*1	拆括号后得到符合输出格式的表达式。
7	(x-1)^2	x^2-2*x+1	拆括号后得到符合输出格式的表达式。
8	(x+2*x^(2+1))^0	1	0 次幂的值为 1。
9	(x+1)^3+1	x^(3+3*x)^2+3*x+2	拆括号后得到符合输出格式的表达式。
10	(x*x*3*x)^2*x	x*x*3*x*x*3*x*x	拆括号后得到符合输出格式的表达式。
11	(x+1)*(x+2)	x^2+x+2*x+2	把两个不嵌套的括号展开得到结果。
12	x*(x-(x+1))	Wrong Format!（无需输出）	嵌套括号不会出现在本次作业中，本次作业不需考虑。

注意：由于本作业可被判定为正确的答案不唯一，以上样例的输出**仅保证正确性，但并不一定为性能最优解。**

第六部分：设计建议

- 在 Java程序中，不建议使用静态数组。推荐使用 `ArrayList` 、 `HashMap` 、 `HashSet` 等容器来高效率管理数据对象。
- 在处理输入解析时，可以考虑采用**递归下降解析法**或是正则表达式作为工具。递归下降方法已在先导课程的作业中得到了详尽的介绍与充分的实践，**建议同学们继续沿用**这一方法。而对正则表达式相关的 API 可以了解 `Pattern` 和 `Matcher` 类。
- 针对形式化定义的表达式结构，可以考虑针对定义中的表达式、项、因子等单独建立相应的类，并通过容器来管理类中的数据元素，从而把整个表达式构建为树形结构。
- 本次作业只需考虑单层括号的展开，但建议**提前考虑**如果下一次作业出现多层括号会对架构产生什么样的影响和如何迭代优化。

第七部分：提示与警示

一、提示

- Java 内的原生整数类型有 `long` 和 `int`，长度分别为 64 位和 32 位，遇到整数过大的问题，可以使用BigInteger存储。
- 要善于利用java语言内置的工具（工具函数或者工具类等），不要重复造轮子!

- 我们鼓励大家通过 Baidu、Google、Stack Overflow 等方式自行学习和解决问题。比如: 程序运行异常的原因很容易通过搜索引擎了解明白。
- 请注意一个往年遇到比较多的问题，使用`Integer.parseInt(String input)`函数抛出了`java.lang.NumberFormatException`异常，一般是因为传入的`input`字符串其中含有非数字字符，以至于函数无法将之转换为数字。
- 如果还有更多的问题，请到讨论区提问。但是**请善用讨论区**，并在此之前认真阅读包括但不限于课程要求文档、指导书、搜索引擎结果等的内容。关于如何提问。

二、警示

- 如果在互测中发现其他人的代码疑似存在**抄袭**等行为，可向课程组举报，课程组感谢同学们为 OO 课程建设所作出的贡献。