



面向对象设计与构造第二次作业



第一部分：训练目标



通过对表达式结构进行建模，完成多项式的括号展开与函数调用、化简，进一步体会层次化设计的思想。

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

本次作业中需要完成任务为：读入一系列自定义函数的定义以及一个包含幂函数、三角函数、自定义函数调用的表达式，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' 。其中， E' 中不再含有自定义函数，且只包含必要的括号（必要括号的定义见公测说明-正确性判定）。

第四部分：迭代内容概览

在第一次作业基础上，本次迭代作业增加了以下几点：

- 本次作业支持嵌套多层括号。
- 本次作业新增三角函数因子，三角函数括号内部包含任意因子。
- 本次作业新增自定义函数因子，但自定义函数的函数表达式中不会调用其他函数。

第五部分：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、19260817 等。
- 因子
 - 变量因子
 - 幂函数
 - 一般形式 由自变量 x, y, z ，指数符号 $**$ 和指数组成，指数为一个非负带符号整数，如： $x ** +2$ ， $y ** 02$ ， $z ** 2$ 。
 - 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 ， y ， z 。



- **省略形式** 当指数为 1 的时候，可以采用省略形式，省略指数符号 `**` 和指数部分，如：`sin(x)`。
- 本指导书范围内的“三角函数”**仅包含 `sin` 和 `cos`**。

○ **自定义函数（新增）**

- 自定义函数的**定义**形如 `f(x, y, z) = 函数表达式`，比如 `f(y) = y**2`，`g(x, y) = sin(x)*cos(y)`，`h(x, y, z) = x + y + z`。
- `f`、`g`、`h` 是函数的**函数名**。在本次作业中，保证函数名**只使用 `f`，`g`，`h`**，且**不出现同名函数的重复定义**（因此每次最多只有 3 个自定义函数）。更具体的约束信息请看第六部分中的数据限制部分。
- `x`、`y`、`z` 为函数的**形参**。在本次作业中，**形参个数为 1~3 个**。形参**只使用 `x`，`y`，`z`**，且同一函数定义中不会出现重复使用的形参。
- 函数表达式为一个关于形参的表达式。函数表达式的一般形式参见**形式化定义**。
- 自定义函数的**调用**形如 `f(因子, 因子, 因子)`，比如 `f(x**2)`，`g(sin(x), cos(x))`，`h(1, 0, -1)`。
- 因子 为函数调用时的**实参**，包含任意一种因子。
- **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，且指数为符号不是 `-` 的整数，例如 `(x**2 + 2*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)* sin(x**2) ** 06 - cos(x) * 3 * sin(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 ** 1113`。
- **空白字符** 在本次作业中，空白字符仅包含空格 `<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，注意带符号整数本身的符号与整数之间也不允许包含空白字符。
- 指数运算符内不允许包含空白字符，如 `* *` 不合法。
- 函数保留字内不允许包含空白字符，即 `sin`，`cos` 关键字内不可以含有空白字符。

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 | 三角函数 | 自定义函数**调用**
- 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 → 自变量 [空白项 指数]
- 自变量 → 'x' | 'y' | 'z'
- 三角函数 → 'sin' 空白项 '(' 空白项 因子 空白项 ')' [空白项 指数] | 'cos' 空白项 '(' 空白项 因子 空白项 ')' [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)



- 空白字符 \rightarrow (空格) | \t
- 加減 \rightarrow '+' | '-'



自定义函数相关(相关限制见“公测数据限制”)



- 自定义函数**定义** → 自定义函数名 空白项 '(' 空白项 自变量 空白项 '[' 空白项 自变量 空白项 '[' 空白项 自变量 空白项 ']' 空白项 ')' 空白项 '=' 空白项 函数表达式



- 自定义函数调用 → 自定义函数名 空白项 '(' 空白项 因子 空白项 ',' 空白项 因子 空白项 '[' 空白项 因子 空白项 ']' ')'



- 自定义函数名 \rightarrow 'f' | 'g' | 'h'
- 函数表达式 \rightarrow 表达式 (注: 本次作业中函数表达式保证不会调用自己或其他自定义函数)

形式化表述中 $\{\}[\cdot]()$ 符号的含义已在第一次作业指导书中说明, 不再赘述。

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**第六部分：输入/输出说明**的**数据限制部分**。

第六部分：输入/输出说明

一、公测说明

输入格式

本次作业的输入数据包含若干行:

- 第一行为一个整数 n ($0 \leq n \leq 3$)，表示自定义函数定义的个数。
- 第 2 到第 $n + 1$ 行，每行为一行字符串，表示一个自定义函数的定义。
- 第 $n + 2$ 行，一行字符串，表示待展开表达式。

输出格式

输出展开括号之后，不再含有自定义函数，且只包含**必要的括号**的表达式。（必要括号的定义见**公测说明-正确性判定**）。

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**。
- **自定义函数定义**满足以下限制：
 - 不会出现重名函数。
 - 函数表达式中不允许调用其他自定义函数或自己（即不允许递归调用）。（注：**但函数调用时实参可以使用自己或其他的函数，即下面的例子**）
 - 函数定义时 $f(x,y) = g((x-1))+1$,不合法。
 - 函数调用时 $f(f((2*x),y),g((z-1)))$,合法。
 - 函数形参不能重复出现，即无需考虑 $f(x,x)=x**2+x$ 这类情况

— 忍數白(1) 寺由山頂的赤良如以須方形全由右白(1)



个单个自定义函数定义的有效长度至多为 150 个字符。其中有效长度指的是去除掉所有空白符后剩余的字符总数。

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，正确性判定为 80 分，性能判定部分为 20 分，二者之和为总分。

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，则性能分部分为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，需要**展开所有括号**且与保持原表达式**恒等**。
 - 展开所有括号**的定义：对原输入表达式 E 做**恒等变形**，得到新表达式 E' 。其中， E' 中不再含有自定义函数，且只包含**必要的括号**。
 - 三角函数调用时必要的一层括号： $\sin()$ 与 $\cos()$ 。
 - 三角函数**对应的嵌套因子为不带指数的表达式因子**时，该表达式因子两侧必要的一层括号： $\sin((x+x))$ 与 $\cos((x*x))$ 。（注意是“不带指数”的表达式因子，如果是 $\sin((x+1)**2)$,这**并不符合必要括号**的定义，你必须将其展开为 $\sin((x**2+2*x+1))$ 这种类似的形式才是合法的)
 - 同样，例如 $\sin(1)$ 与 $\sin((1))$ 均为展开形式，但 $\sin(((1)))$ 不是，因为后者除了函数调用和三角嵌套表达式因子的一层括号外，还包括了表达式内嵌套表达式的括号
 - 本次作业中对于恒等的定义：设 $f(x)$ 的定义域为 D_1 ， D_1 包含于 R ， $g(x)$ 的定义域为 D_2 ， D_2 包含于 R ，对任意 $x \in D_1 \cap D_2$ ， $f(x) = g(x)$ 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：

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

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%
1.3	10.9%
1.4	4.5%
1.5	0.0%

当测试时，你可以通过定义输入数据帮助得到正确的输出，以组数据去做单元测试问题。当测试问题中的其他问题的程序。输入数据必须符合上述的文法规则。提交的输出只需要包含一行，即输出的正确表达式。

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**。
- 自定义函数限制与公测相同，见上文公测数据限制。
- 对于规则“指数 \rightarrow ****** 空白项 $[+]$ 允许前导零的整数”，我们本次要求**输入数据的指数不能超过 8**。
- 最终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）
- 自定义函数定义的有效长度至多30个字符，有效长度定义同上
- 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 $\text{Cost}(\text{Expr}) \leq 5000$ ，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

- $\text{Cost}(\text{常数}) = \max(1, \text{len}(\text{常数}))$ （常数的前导零不算在其长度内）
- $\text{Cost}(x) = \text{Cost}(y) = \text{Cost}(z) = 1$
- $\text{Cost}(a + b) = \text{Cost}(a - b) = \text{Cost}(a) + \text{Cost}(b)$
- $\text{Cost}(a * b) = \text{Cost}(a) * \text{Cost}(b)$ （多项相乘时从左到右计算）
- $\text{Cost}(\sin(a)) = \text{Cost}(\cos(a)) = \text{Cost}(a) + 1$
- $\text{Cost}(a ** b) =$
 - 若a是单变量因子， $\text{Cost}(a ** b) = 1$
 - 若a是表达式因子 (c)， $\text{Cost}(a ** b) = \max(\text{Cost}(c), 2) ^ \max(b, 1)$
 - 若a是三角函数因子， $\text{Cost}(a ** b) = 2 ^ b + \text{Cost}(a)$
- $\text{Cost}(+a) = \text{Cost}(-a) = \text{Cost}(a) + 1$
- $\text{Cost}(h) = \text{Cost}(h') * 2$ ，h是自定义函数调用，其中 h' 是将调用 h 的参数作为表达式因子代入后，所得到的表达式。同时注意h的实参代价不能超过阈值500。

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。

三、样例

#	输入	输出	说明
1	1 f(x)=x**2 (x+f(x))*x	x**2+x**3	带入 f(x)之后得到(x+x**2)*x，展开括号可得结果
2	0 (x+1*(x+2*(x+3)))	4*x+6	括号嵌套层数没有限制
3	1 f(z)=z*(sin(z)) sin(((x+1)**2-1))+cos((-y-1)+f(0)))	sin((x**2+2*x))+cos((-y+1))	三角函数内部可以包含任何因子
4	1 f(x)=x*(sin(x)+cos(x)) x-(f(y)+x)	-y*sin(y)-y*cos(y)	带入 f(x)之后展开可得到结果



	$f(\sin(x)) + \cos(z)$		
6	1 $f(x,y,z) = x + y^{**2} + z^{**3}$ $f(\sin(x)^{**2}, \cos(x), f(1,1,1))$	$\sin(x)^{**2} + \cos(x)^{**2} + 27$	自定义函数调用的实参可以调用自定义函数(包括自己)
7	2 $f(x,y) = x + y$ $g(x,y) = x * y$ $x * f(\sin(x), \cos(x)) + g(\sin(x), \cos(x))$	$x * \sin(x) + x * \cos(x) + \sin(x) * \cos(x)$	多个自定义函数

第七部分：设计建议

- 在 Java 内，不建议使用静态数组。推荐使用 `ArrayList`、`HashMap`、`HashSet` 一类的数据结构，快速管理和调配手中无序的数据。
- 对于使用一般输入手动解析的同学，处理字符串时可以考虑使用**正则表达式**，相关的 API 可以了解 `Pattern` 和 `Matcher` 类；另外也可以考虑使用**递归下降**等方法进行解析，相关教程我们会放在实验或者训练中。
- 这次作业看上去似乎很难，其实找对了方法后并不难。关键思想是“**化整为零**”，可以这样考虑：
 - 按照文法所定义的层次化结构来建立类（如表达式、项、因子）
 - 对于每一种运算规则（乘法、加减法），可以分别单独建立类，也可以将运算规则作为层次化结构类的功能部分。
 - 对于自定义函数，可以先将其 **定义表达式** 展开，将展开后的结果代入进行计算。

第八部分：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。
- 不要重复造轮子！不要重复造轮子！不要重复造轮子！重要的事情说三遍**
- 我们鼓励大家通过 Baidu、Google、Stack Overflow 等方式自行学习和解决问题。
- 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。

