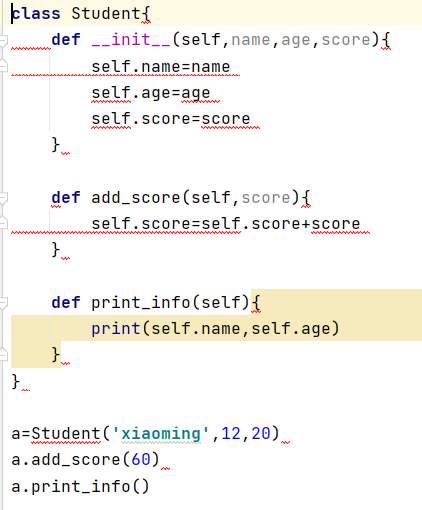
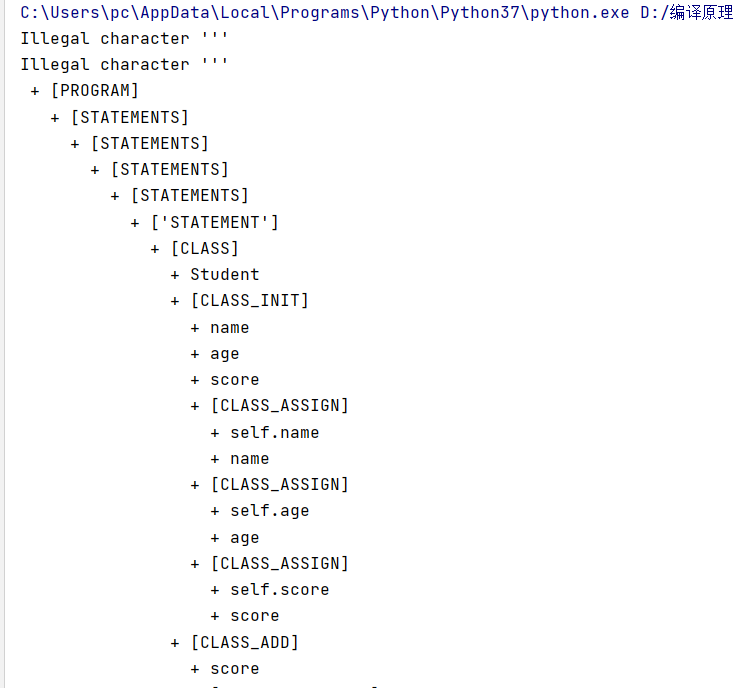
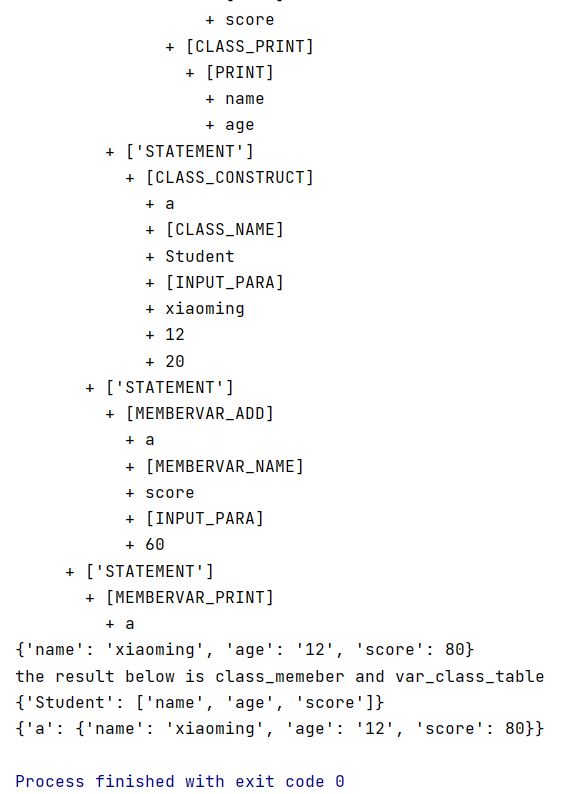
**输入输出说明：**

本次实验的解析样例已经放在**stu.py**文件中，（stu.py文件需放在和main.py放在一个目录下）执行本程序则只需要运行**main.py文件即可**。

**解析样例如下：**



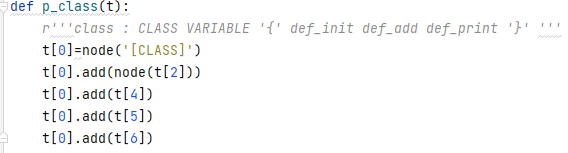
**程序输出结果如下：**

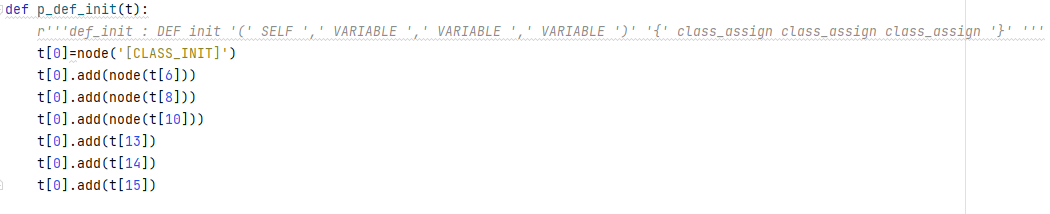
**设计思路：**

类的分析与之前不在于有构造函数，自己的成员变量，所以要针对这两样进行分析

首先将类名 初始化函数 改变成员变量函数 打印成员变量函数加入语法树

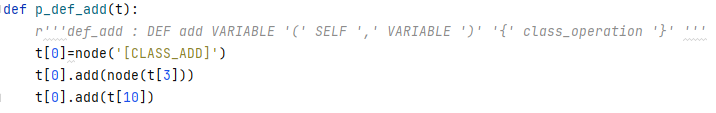


因为需要解析的Student类需要三个成员变量所以需要有三个节点分别将其加入语法树中

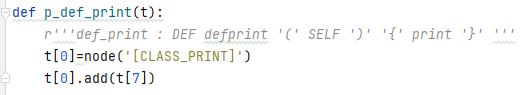


改变成员变量值函数 默认add\_var var即为要改变的变量

把改变的VAR和操作函数加入语法树



因为类的打印与常规打印不同所以要构建单独对类成员的打印



调用成员函数部分

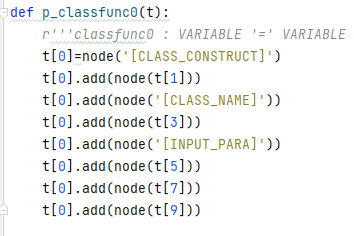
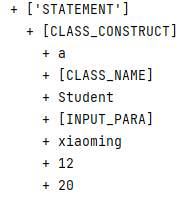
此处也考虑了三种情况：

调用构造函数

对于测试样例中的各种类的用法进行如下分析：

记录哪个变量调用了构造函数 将调用的类名 和输入的变量参数记录入语法树

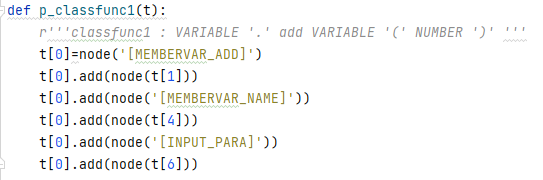




改变成员变量函数

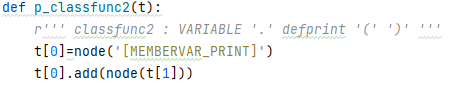
记录要改变的变量名 成员变量名 改变的数值





打印函数

记录要打印的变量名

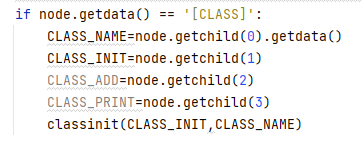


语法制导翻译部分

进行了一定的简化，反复使用历史代码会产生问题

检测到类函数

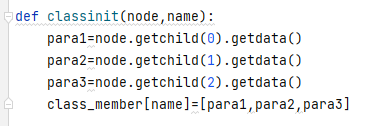
记录各个节点，调用初始化节点开始初始化。



初始化函数

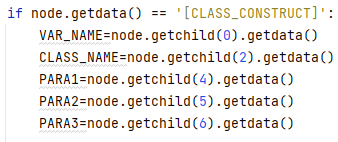
如图向class\_member中写入给定类所含的成员变量





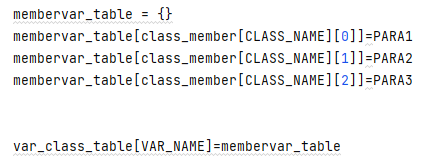
调用构造函数

获取变量名和类名 以及输入的参数名



调用成员的类名和输入参数构造键值对生成hashmap

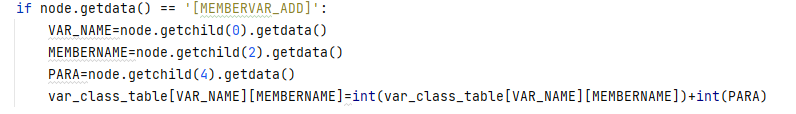
并将其和变量名相匹配 放入var\_class\_table中

改变成员变量的函数

获取变量名 成员变量名 改变量

改变var\_class\_table中键值对的值



打印函数 输出对应变量名的table

