**设计文档**

**项目结构**

全部文件包括**main.cpp，Graph.cpp，Graph.h，Node.cpp，Node.h，Cut.h**以及一个**makefile**。其中，两个Graph文件针对整个图的构建与处理（计算图采用二叉树的形式，对于一元运算，左儿子为参数，右儿子为空指针；对于二元运算，左儿子右儿子分别为两个参数；对于Cond运算，左儿子为参数（判断参数），右儿子为空指针），两个Node文件针对单个节点的构建与处理，Cut文件存储了处理单个语句的切块函数（按空格将语句切开并存入数组，**注意：数组arr是存储在一个名为Temp的结构体里的，在Graph.h里面**）。

**main.cpp**

分三部分处理。（每一句需要调用一次切块函数）

第一部分输入n个初始变量，有三种形式：占位符、常量和变量（后两者其实无本质不同，只是类型名称记录得不一样而已），直接分别调用Node的构造函数即可。

第二部分输入m个节点操作语句，每一句操作一次节点，调用buildgraph函数即可。

第三部分输入q个赋值计算语句，每一句输出一次（SETANSWER和SETCONSTANT除外），调用Graph类中的eval即可。

**Node.h里存储的Node类**

ty代表节点操作类型，存储的可以是加减乘除号、四种进阶运算符号、P（代表占位符）、C（代表常数）、V（代表变量）以COND。

name是这个节点的名称，字符串类型。

value是这个节点当前的值，没有赋值的话值为MAX。

left和right指向左儿子和右儿子。

两个output是COND时的两个可能的输出（字符串类型）。

**Node.cpp**

**Node**类代表基本的指针，实际上**Constant**，**Variable**，以及**Placeholder**类均继承它（只是因为赋值原因而没有把Variable写成继承关系）。Node.cpp里存储了这几个类的一些需要用到的构造函数。

**Graph.h里存储的Graph类**

relation是存储了节点名称与节点对应关系的一个map，所有与寻找节点相关的操作都要用到relation。

error1、2、3分别用于记录三种报错信息，若为true则输出报错信息。

answer存储了cout的结果，哪一行输出就记入对应位置（float类型）。

NodeConstruct是构建新节点的函数，参数是节点类型、节点名称、节点值。

eval是主函数中调用的计算赋值与输出的函数。

calculate是从二叉树的顶部递归计算赋值的函数（eval里调用），而Cal是其特化版本，只负责处理二元运算。

buildgraph是建图函数，负责构建节点和构建节点之间的关系，其中调用了NodeConstruct函数用于单独构建新节点。其中还主要处理了：1.节点覆盖的情况；2.节点重名的情况（即单个语句中出现同一名称多次）。

reset是还原函数——在每次计算过后，节点的value和print情况都会变，因此要调用之以实现二者的还原。

setconstant函数和setanswer函数用于处理SETCONSTANT和SETANSWER两种运算。

Constructnewrelation是用于处理二元运算关系的函数。当没有出现节点覆盖与重名且是二元简单运算时，调用之实现关系建立。第一个参数是节点名称，第二个参数是节点类型，后两个参数是左右儿子。