# ­­­­­­­­­­Project 1技术报告

姓名：欧穗新 学号：16340173 班级：教务三班

## 实验目的：

实现一个多项式计算器

## 二、实验环境：

Linux下（g++编译）

## 分析与设计：

## 3.1需求分析：

需要实现的基本功能有：

1. 在任何时候能够返回主菜单界面
2. 存储多项式
3. 多项式加法
4. 多项式减法
5. 多项式求值
6. 查看现有的所有多项式

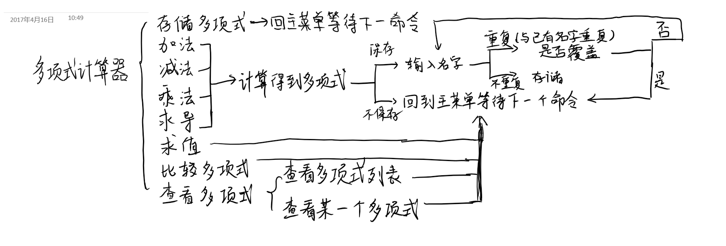
扩展功能有：

1. 多项式与多项式相乘
2. 判断两个多项式是否相等
3. 对某个多项式求导

自己实现的额外功能：

1. 根据名字查找特定多项式
2. 文档记录程序的多项式

## 3.2实现思路：



本次project是使用map容器实现的

一、

首先是一个全局的map变量，用来存储所有的多项式（该map的每一个变量就是一个多项式），它的key是每一个多项式变量的名字，它的value是一个多项式类；

二、

多项式类的数据是一个map，这个map的一个元素是多项式的一项，key是这一项的指数位的值value是这一项的系数的值（这样就可以很方便的**通过作为key的指数**得到**指数相等的项**进行**系数相加，**从而**实现加减**和**合并同类项**）

多项式类的一系列函数则实现了几乎所有功能。

1. 所有函数的具体实现方式：

功能1：用户输入一个多项式（以“name = （1,2）（2,1）”的形式）

函数解析出name部分，使用map.find(name)，如果没有找到，直接使用map【name】 = （1,2）（2,1）插入；如果找到，说明之前已经有这个多项式了，询问是否覆盖，若是，则覆盖，否则要求重新输入

功能2：用户要输入两个多项式，函数解析得到具体多项式以后使用

newmap【i1.first】+= i1.second; newmap【i2.first】+= i2.second（i1，i2分别是第一个和第二个多项式的每一个元素）

来实现把这两个多项式的每一项都加入到newmap

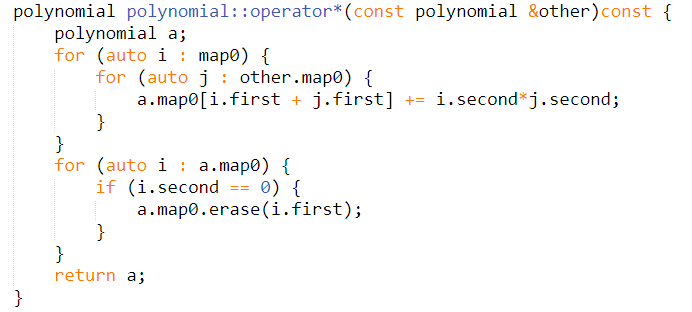
功能3：类似功能2，用户要输入两个多项式，函数解析得到具体多项式以后使用

newmap【i1.first】+= i1.second; newmap【i2.first】-= i2.second（i1，i2分别是第一个和第二个多项式的每一个元素）

先把第一个多项式的每一项加入到newmap，然后减去第二个多项式的每一项

功能4：类似2，3：用户要输入两个多项式，函数解析得到具体多项式以后

运用一个二层的遍历，把两个多项式的每一项相乘然后相加

像这样！

功能5：就是用户要输入多项式，函数解析得到具体多项式以后直接每一项分别做求导计算就OK了

功能6：就是就是用户要输入多项式和一个值，把值带入到多项式进行计算就好了

功能7：用户要输入两个多项式，两个多项式的每一项的系数，指数都进行比较，若都相等则打印“equal”，否则不

功能8：可以输入-1，然后就遍历全局map打印所有存储的多项式，

也可以输入数字，然后输入n个变量的名字查找对应的多项式（使用map.find（name））

再有就是存档函数：就是利用map的遍历存入文档，

然后读函数：就用“map【name】= 多项式”插入

**关于任何地方输入“Q”都可以回到主菜单**，因为我的所有输入都是用一个函数

Input\_and\_check()来处理的。所以可以首先就判断是不是输入了Q，如果是，就返回信息，使得调用程序退回到主菜单

总之，详见代码，，，，，，

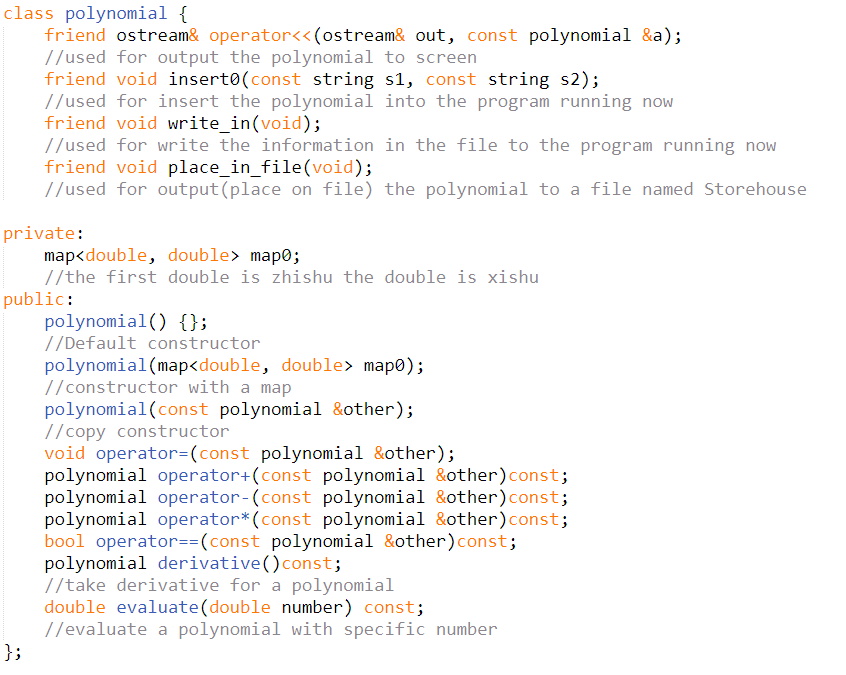
## 3.3数据设计：

**拥有一个map类型的全局变量**，关键词是string name（一个多项式名字），值是多项式；

用来存储当前程序中所有的多项式。

自定义的**多项式类：**

代码如下：



**解释：**

1. **数据**：拥有一个map关联容器，拥有多个pair<double, double>,每一个都表示多项式的一项，多个项组合，从而表示一个多项式（并且在map内部，所有的项是升序排列，所以只要用反向迭代器就可以降序输出多项式）

它的关键词是一个double类型的变量，用来存储多项式某一项的指数；

它的值也是一个double类型的变量，用来存储多项式某一项的系数；

1. **函数**：

具有三个**构造函数**：默认构造函数；以一个map容器为参数的构造函数；拷贝构造函数；

**重载**了“+”、“-”、“\*”、“<<”，“=”（赋值运算符）、“==”（判断两个多项式类型的变量是否相等）；

**具有求导、求值函数**（接受一个double变量作为参数来对多项式求值）；

另外有**四个友元函数**：

一个是对<<的重载，用于向屏幕输出传入的多项式变量；

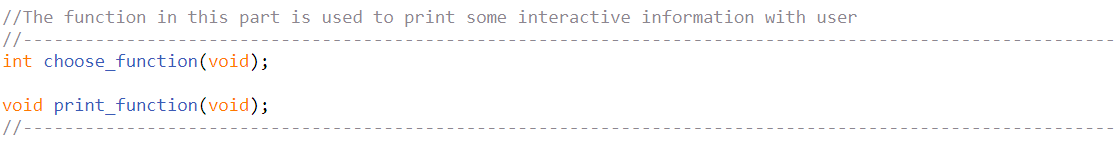
一个是用于接受一个多项式名字（string）和一个多项式类做为参数，把该多项式存储到保存所有多项式的map容器；

一个用于在程序开始之前将文件保存的内容输入到存储所有多项式的map容器中；

一个用于在程序退出之前将存储所有多项式的map容器的值全部保存到文件中；

## 3.4函数设计

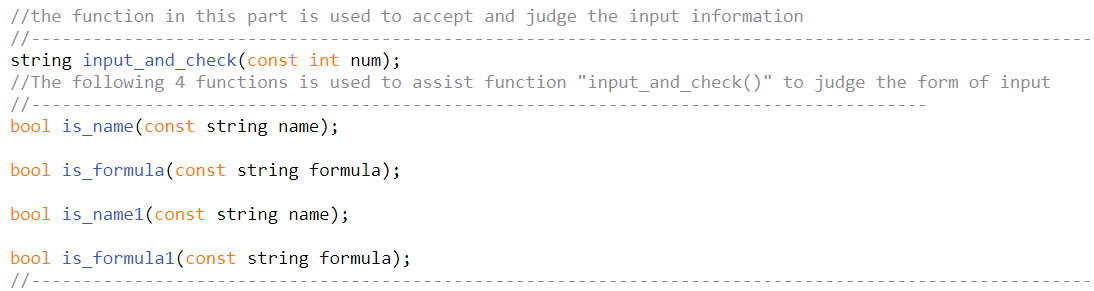
**一、首先是与用户交互的函数：**



第一个函数用于给用户选择需要的功能，返回一个int，如果该返回值是0，则退出程序，否则程序回到主菜单，继续由用户进行功能选择。

第二个函数用来打印所有的功能供用户选择

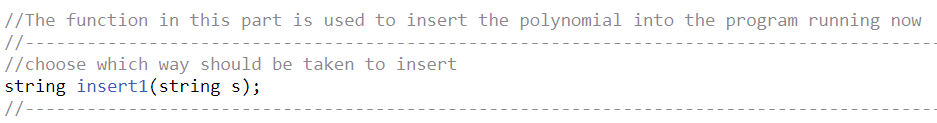
**二、然后是接受并检查用户输入的函数**



**第一个函数**接受一个int做为参数，可以接受用户输入，并且根据int是多少来选择判断方法，并且将判断结果存储到一个string里面，返回给调用程序进行解析。

**后面四个**函数的都是用来辅助第一个函数进行判断的

**三、将一个多项式插入到用来保存所有的多项式的容器中的函数**

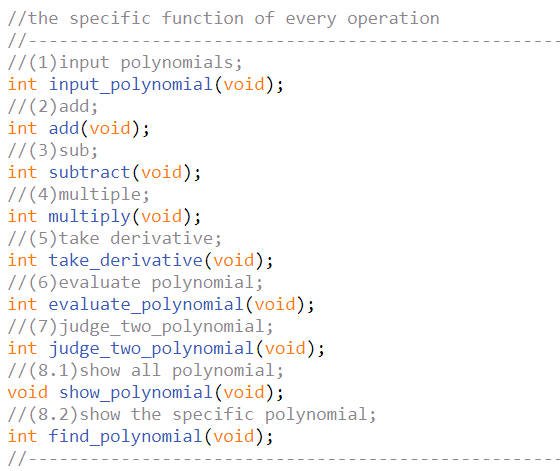


这个函数接受一个含有多项式所有信息的string（如“name = （1,2）（2,1）（1,0）”），

对string内容进行解析，然后分成name部分和式子部分存储到全局容器中

返回一个string变量，这个string变量是刚刚插入的多项式的名字

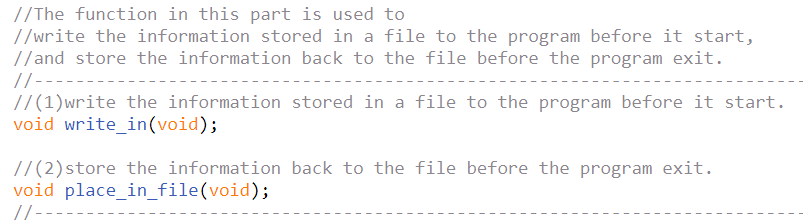
**四、计算器功能具体实现的函数：**



这些函数的功能就像它们的名字一样直白，，，，，，

返回的int参数如果不为零，表示程序继续，否则返回主菜单

**五、存档函数：**



这些函数的功能就如注释所说，用来存档和读档保存的多项式的

## 3.5最终功能

1.基本功能

需要实现的基本功能有：

1. 在任何时候能够返回主菜单界面
2. 存储多项式
3. 多项式加法
4. 多项式减法
5. 多项式求值
6. 查看现有的所有多项式

2.扩展功能

扩展功能有：

1. 多项式与多项式相乘
2. 判断两个多项式是否相等
3. 对某个多项式求导

自己实现的额外功能：

1. 根据名字查找特定多项式
2. 文档记录程序的多项式

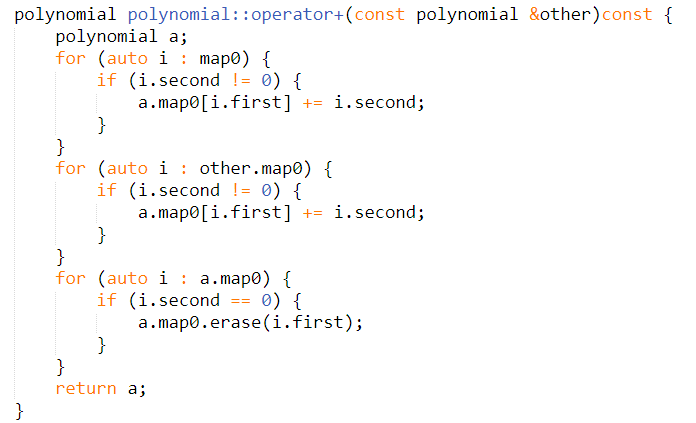
## 3.6不足与改进

自我感觉良好，无论是**健壮性**还是**可用性（会有针对错误输入的特定提示）**都不错

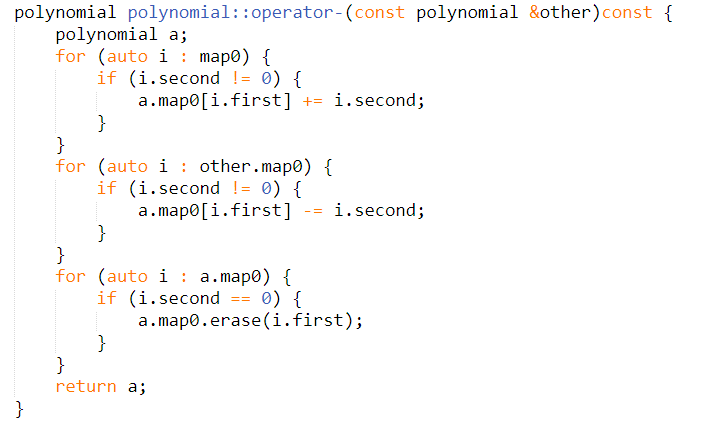
如果说有什么不足，就是界面可能不是很好看（但是如果非要设计一个用框框起来的界面就需要用户把界面调大，感觉不是很友好，而且一旦终端界面不够大，就是一团乱，所以就没有做）

## 重要代码

这是对‘+’的重载：



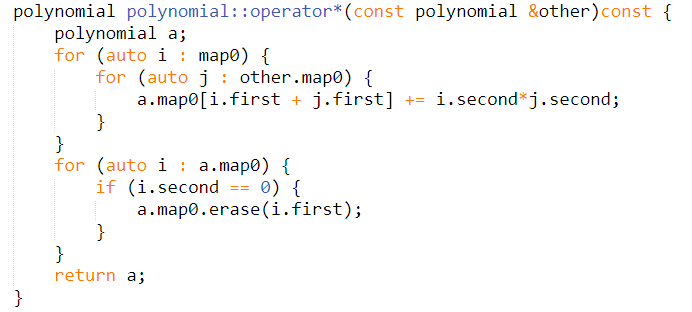
这是对‘-’的重载：



显然相似度很高，因为只是符号不同

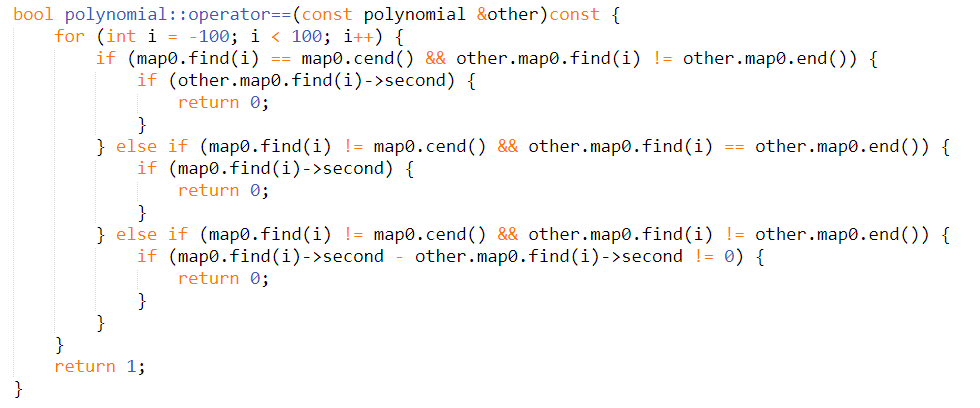
都是先把某一个多项式遍历所有项载入新的多项式类，然后把另一个遍历所有项载入

这是对‘\*’的重载：



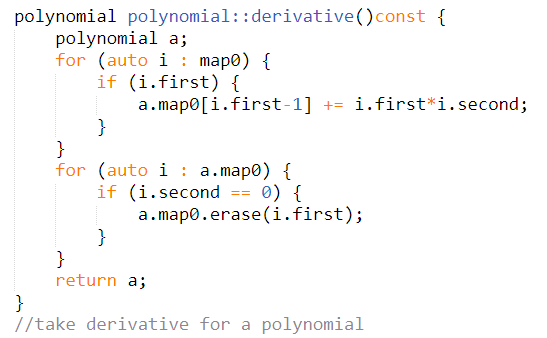
把所有项分别相乘再相加，载入新的多项式类

这是对==的重载：



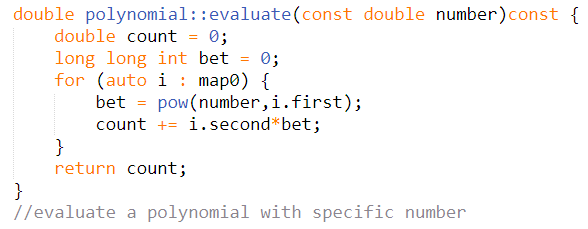
通过作差后与0比较判断某一项是不是相等，如果每一项都想等，就想等。

这是求导函数：



就是正常的求导，然后倒入新的多项式变量

这是求值函数：



就是带入一个值进行计算，，，，，，

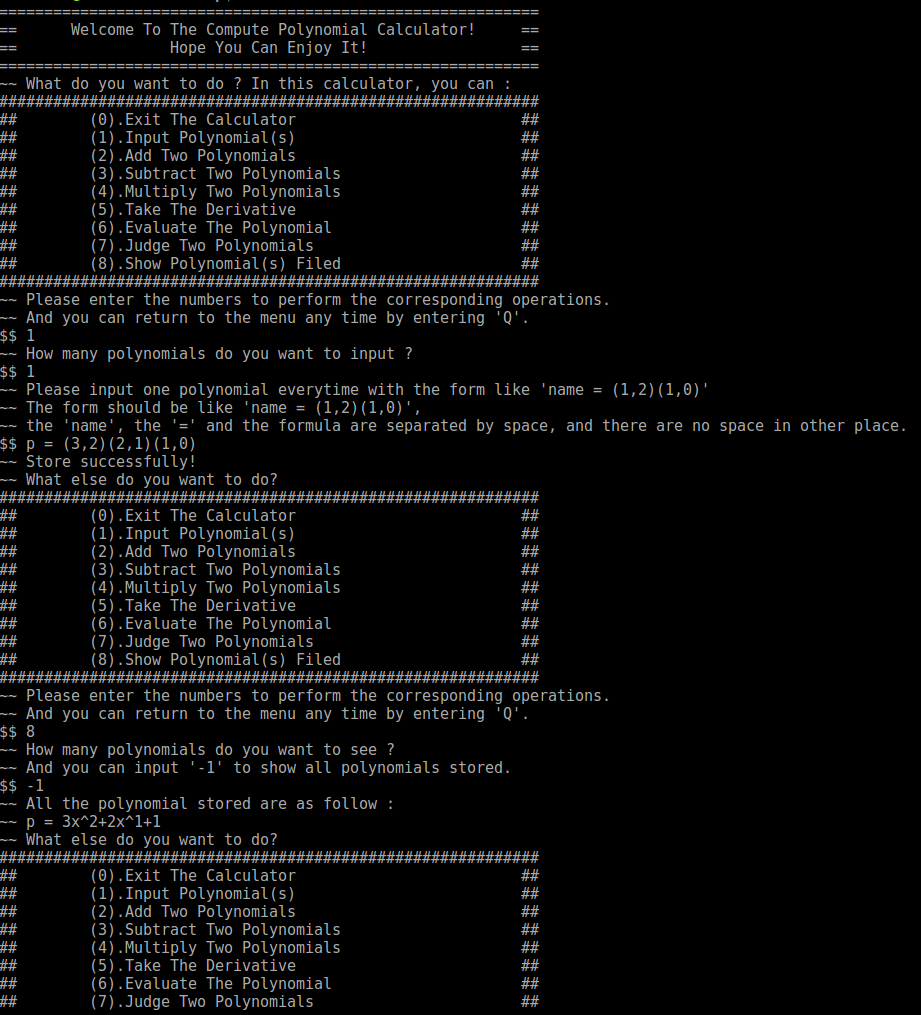
## 五、实验结果（截图测试数据及其结果）

1.测试意图&测试结果

测试意图：

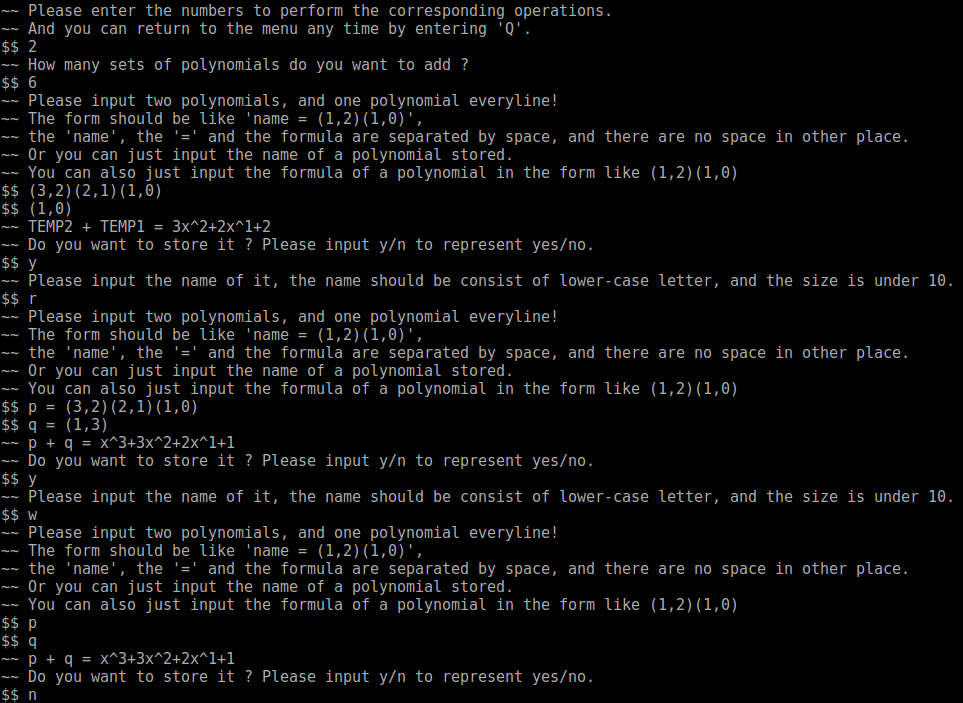
1. 测试所有功能是不是都实现了

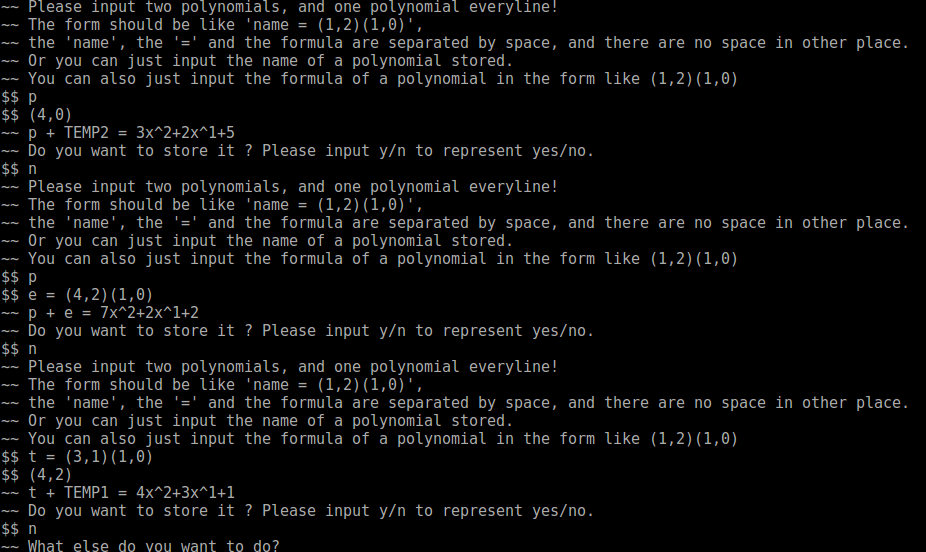
功能1和8的测试：



先保存了一个多项式，然后使用8的部分功能查看了它

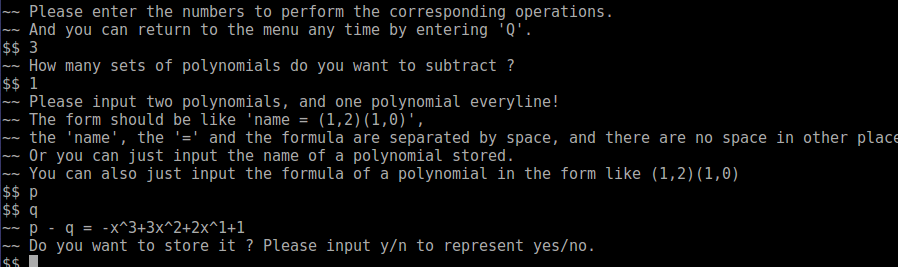
再对第二个功能进行测试：



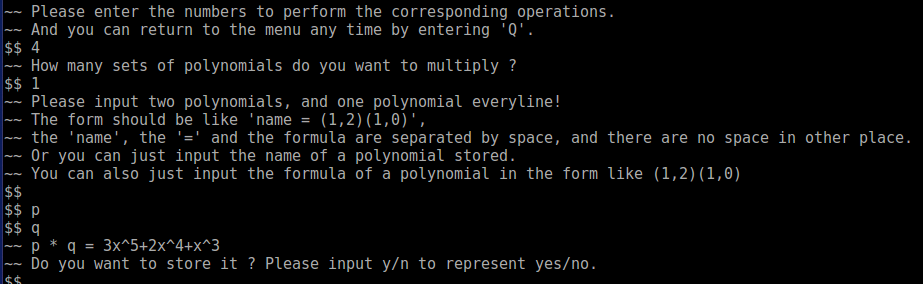


可以从上面看到，6种输入方式全部是可以的，并且结果是对的

对第三个功能测试是类似的，仅仅附上图：

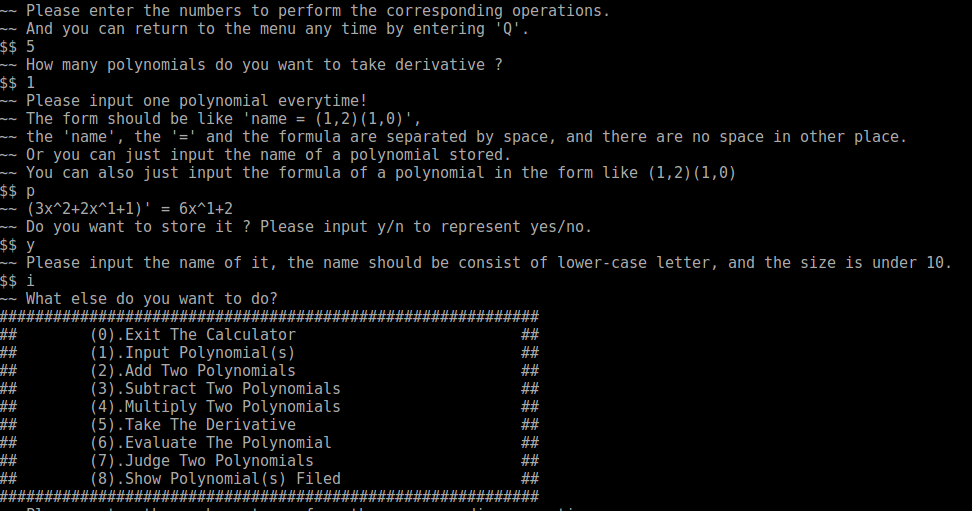


对第四个功能测试：



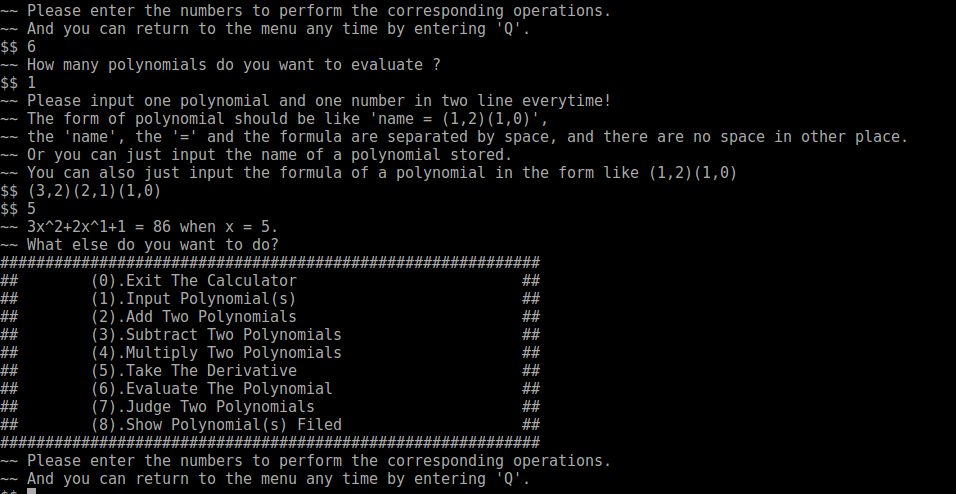
这是对乘法的测试，可以看出，是对的

再对第五个功能进行测试：



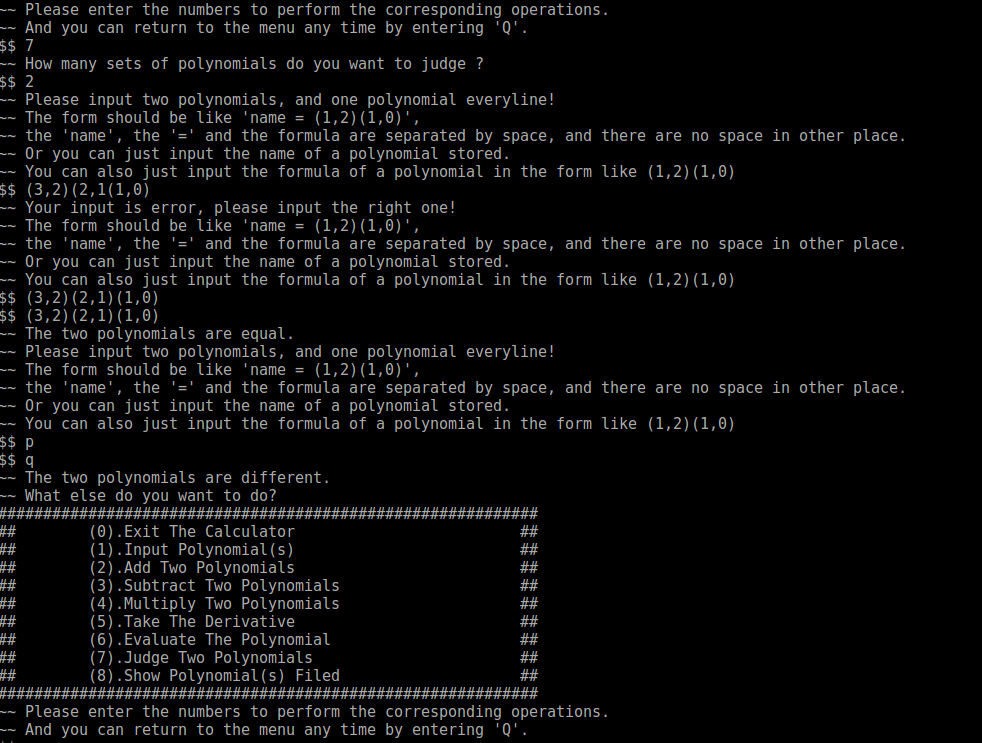
显然结果是对的

再对第六个功能进行测试：



显然也是对的

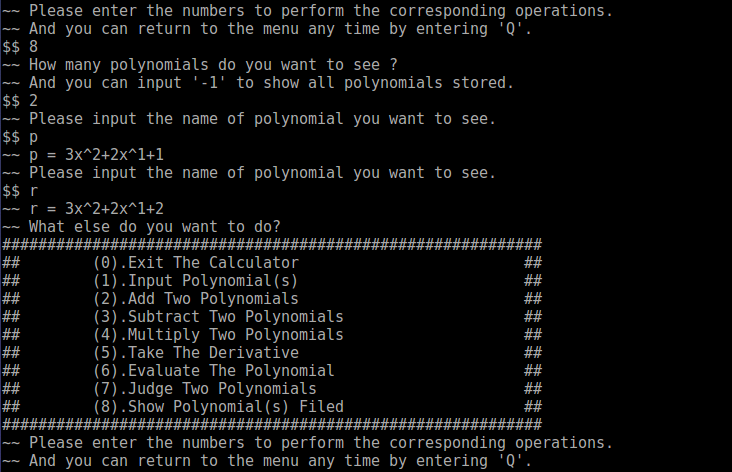
再对第七个功能进行测试：



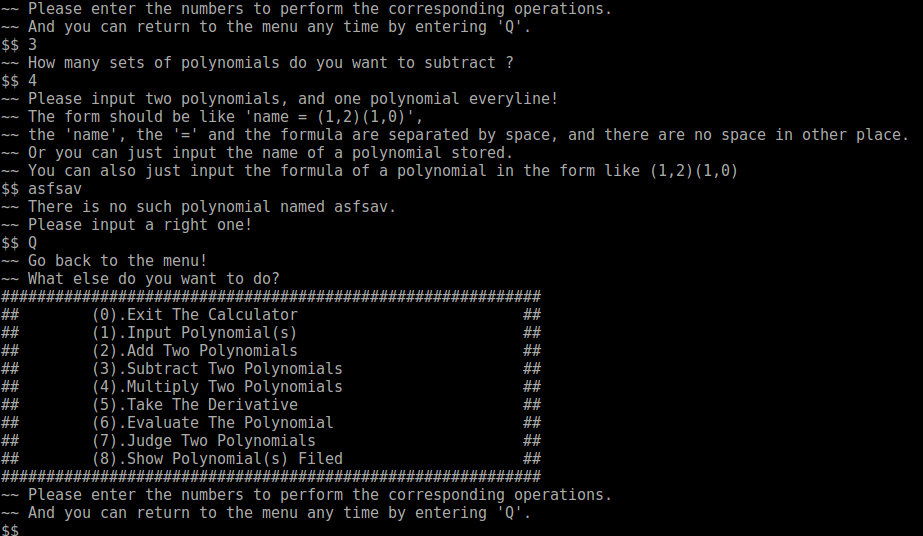
我们看到第一组两个相等，第二组不相等

同时我们也看到输错是会报错并且提示正确输入的

最后是第八个功能的查找功能的测试：



还有就是任何地方输入Q返回主菜单：



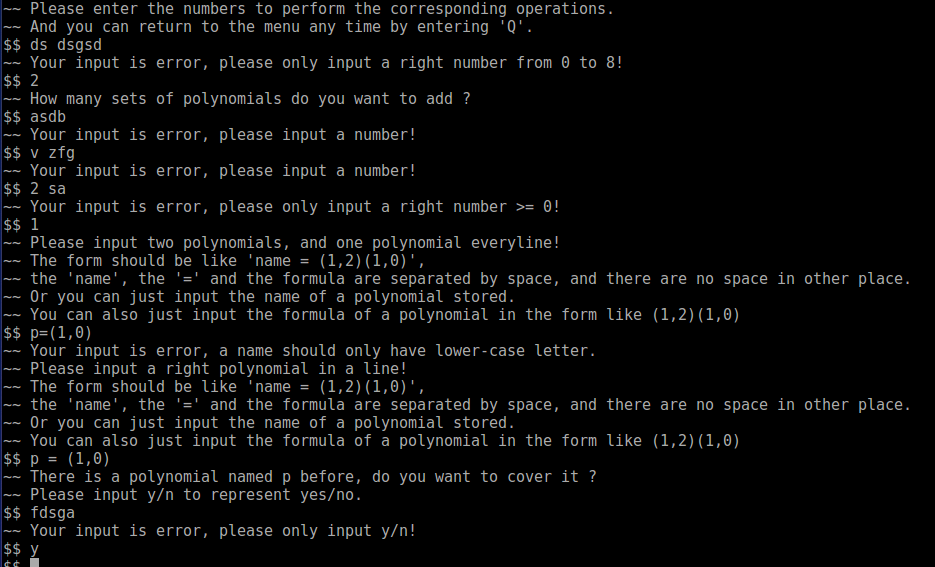
这里就利用Q返回主菜单

另外，可以有负数项和小数项：

![C:\Users\14735\Documents\Tencent Files\1473595322\FileRecv\MobileFile\Image\6%8MDSWBDC6H4_AD](]IF%I.png](data:image/png;base64,)

而且也是对的

1. 测试错误输入的报错：



可以看出错误输入是都有报错并且提示的

2.已经发现并改正的错误

感觉都是细节错误，比如变量中途被覆盖，导致出错之类，所以不值一提，，，，，，

错误原因

改正方法

改正后结果

## 六．心得体会

对比上次的project，这次的project好像没有什么额外发挥空间（想不出太多的额外功能）

但是却并不代表这次的就简单，相反，我觉得这次的输入处理更加难。

一、上次只需要处理一些常规的**输入**----string，int，double就可以了

这次则是由一个多项式的式子需要处理，考虑到中间有空格，有括号，括号里面还有double、有‘+’、‘-’；

二、上次的**数据**要简单一些，个数看起来多一些，其实是更少，（因为这一次的多项式计算，多项式的项数是可以有很多项的），而且都是简单的int，double，string。并且数据之间不需要考虑顺序，只需要存储就好了。

但是这一次则是存储多个多项式，每一个多项式的项是可以增加，这就意味着数据个数比上一次更多，同时还需要考虑到多项式每一个项的降序排列。

1. 这次的**输出处理**也不简单：由于需要人性化的输出----类似于3x^2+x^2-1

这就意味着需要考虑每一项系数的正负，是不是1等等因素

1. 这次需要的**类的运算**很多，二上一次基本上不需要用到运算，只需要能够存储、删除就好了。

**所以，综上所述，感觉这次的project是上一个的超集，，，，，，**

**至于收获，还是有的：**

**比如string的处理，正则表达式的运用**

**比如map容器的运用，直接使用map【key】进行访问/添加元素（简直超级方便，，，）**

**比如反向迭代器的使用**

**比如锻炼了打码的速度熟练度**

**等等等等。**

**最后，谢谢ta看我的破代码/调试我的破程序orz**

--------------------------- 阿里嘎多啊 ----------------------------