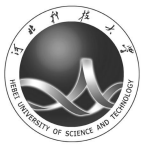


河北科技大学

课 程 设 计

学生姓名：刘敬超 学 号：2107090411

专业班级：软件工程 2102

课程名称：数据结构课程设计

学年学期：2022—2023 学年第 2 学期

指导教师：郑 广

2023 年 6 月

数据结构课程设计成绩评定表

<table>
<tr><td>学生姓名</td><td>刘敬超</td><td>学　号</td><td>2107090411</td><td>成绩</td><td></td></tr>
<tr><td>专业班级</td><td>软件工程 2102</td><td>起止时间</td><td colspan="3">2023/6/19~2023/6/23</td></tr>
<tr><td>设计题目</td><td colspan="5">顺序结构、动态链表结构下的一元多项式的加法、减法、乘法的实现。</td></tr>
<tr><td>指导教师评语</td><td colspan="5">学习态度出勤情况：　好 ☐　较好 ☐　一般 ☐　较差 ☐
课 题 工 作 量：　饱满 ☐　较大 ☐　合理 ☐　较小 ☐
综合运用知识能力：　好 ☐　较好 ☐　一般 ☐　较差 ☐
方 案 设 计 情况：　合理 ☐　较合理 ☐　基本合理☐　不合理 ☐
课题结果分析能力：　强 ☐　较强 ☐　一般 ☐　较差 ☐
设 计 实 现 情况：　全部 ☐　大部分 ☐　部分 ☐　未实现 ☐
设 计 报 告 内容：　详细☐　完整 ☐　较完整 ☐　不完整 ☐
设计报告文档格式：　规范 ☐　较规范 ☐　基本规范☐　不规范 ☐
独 立 动 手 能力：　强 ☐　较强 ☐　一般 ☐　较差 ☐

指导教师：_____

_____
年　月　日</td></tr>
</table>

目　录

1 需求分析说明

内部排序教学软件的总体功能要求：

本程序的主要目标是实现对两个一元多项式的加、减、乘操作，并满足以下基本功能要求：

能够判定多项式是否为稀疏多项式。

对两个一元多项式进行加、减、乘操作后，所得结果中无重复阶项和无零系数项。

输出的结果可以按照升幂和降幂排列。

以下是顺序结构下各功能模块的功能描述：

1. 主函数模块

实现多项式的初始化，调用各模块，实现各个功能。

2. 两多项式相加模块

将用户输入的两个一元多项式进行相加操作。

3. 两多项式相减模块

将用户输入的两个一元多项式进行相减操作。

4. 两多项式相乘模块

将用户输入的两个一元多项式进行相乘操作。

5. 输出模块

将程序的运行结果输出，可以指定输出哪个多项式，输出方式（控制台或文件），输出顺序（升幂或降幂）

以下是动态链表结构下各功能模块的功能描述：

1. 主函数模块：

实现多项式的初始化，调用各模块，实现各个功能。

2. 创建一元多项式子模块：

根据用户输入的数据信息，创建一元多项式。

3. 一元多项式转字符串子模块：

将一元多项式转换为对应的字符串表达。

4. 判断多项式是否稀疏子模块：

根据用户输入的一元多项式，判断该多项式是否为稀疏多项式。

5. 两多项式相加子模块：

将用户输入的两个一元多项式进行相加操作。

6. 两多项式相减子模块：

将用户输入的两个一元多项式进行相减操作。

7. 两多项式相乘子模块：

将用户输入的两个一元多项式进行相乘操作。

8. 合并同类项子模块：

将多项式中相同幂的系数合并在一起。

9. 升幂排列子模块：

将一个多项式按照升幂排列。

10. 降幂排列子模块：

将一个多项式按照降幂排列。

11. 输出子模块

将程序运行的结果输出到控制台和文本文件中。

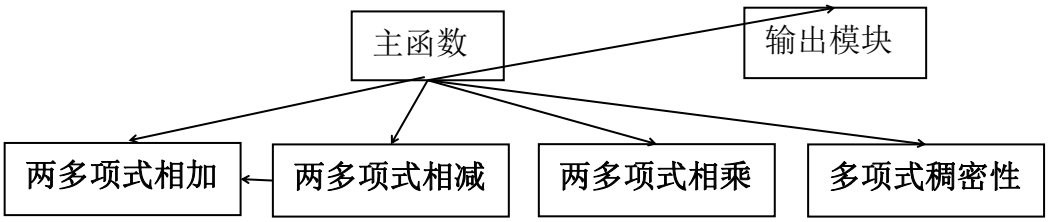
测试数据

第一个多项式：3x + 2x^2 + 5x^4 + 9x^9

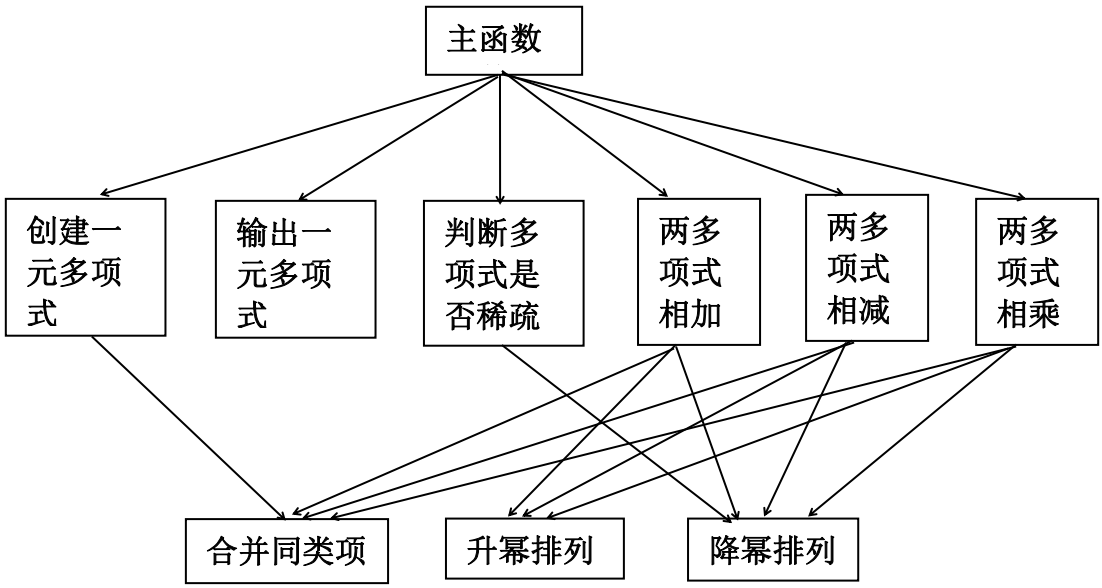
第二个多项式：x - 2x^2 +5x^3

2 概要设计说明

顺寻结构模块调用图如图一所示



动态链式结构模块调用图如图二所示。



图二

动态链式存储结构抽象数据类型一元多项式的定义如下：

```
struct term{
float coef;    //系数
int expn;      //指数
};
struct LNode{
term data;
struct LNode *next;
};
typedef LNode* polynomail;
polynomail creatpolyn(int m);//输入 m 项的系数和指数，建立表示一元多项式的有序链表
void polyntoString(polynomail P); //一元多项式 P 转字符串
polynomail addpolynup(polynomail pa,polynomail pb); //多项式的相加运算返回升幂结果
polynomail addpolyndown(polynomail pa,polynomail pb); //多项式的相加运算返回降幂结果
```

polynomail subpolynup(polynomail pa,polynomail pb); //完成多项式的相减运算返回升幂结果
polynomail subpolyndown(polynomail pa,polynomail pb); //完成多项式的相减运算返回降幂结果
polynomail mulpolynup(polynomail pa,polynomail pb);//完成多项式的相乘运算返回升幂结果

polynomail mulpolyndown(polynomail pa,polynomail pb);//完成多项式的相乘运算返回降幂结果
void arrangeup(polynomail pa); //对多项式 pa 升幂排列
void arrangedown(polynomail pa); //对多项式 pa 降幂排列
polynomail hebing(polynomail Head); //对多项式合并同类项
bool judge(polynomail Head); //判断多项式是否稀疏
outputTable(ostream& out, polynomail pa, polynomail pb, polynomail addpup, polynomail addpdown, polynomail subpup, polynomail subpdown, polynomail mulpup, polynomail mulpdown)// 将程序运行结果输出到控制台和文本文件

顺序存储结构的数据类型一元多项式的定义如下：

```
struct Term {
    int degree;        // 指数
    double coeff;      // 系数
};

// 一元多项式加法
vector<Term> add_poly(const vector<Term>& A, const vector<Term>& B)
// 一元多项式减法
vector<Term> subtract_poly(const vector<Term>& A, const vector<Term>& B)
// 一元多项式乘法
vector<Term> multiply_poly(const vector<Term>& A, const vector<Term>& B)
// 输出多项式
void print_poly(const vector<Term>& P, bool ascending = true, ostream& out = cout)
// 判断多项式是否稠密
bool is_dense(const vector<Term>& P, double density_threshold = 0.5)
```

3 详细设计说明

动态链式存储结构设计如下：

1. 主函数模块

首先调用 creatpolyn()函数创建一元多项式，再调用 printpolyn()函数输出一元多项式，然后调用 judge()函数判断多项式是否稀疏，调用 addpolynup()、printpolyn()输出两多项式升幂相加结果，调用 addpolyndown()、printpolyn()输出两多项式降幂相加结果，调用 subpolynup()、printpolyn()输出两多项式升幂相减结果，调 subpolyndown()、printpolyn()

输出两多项式降幂相减结果，调用 mulpolynup()、printpolyn()输出两多项式升幂相减结果，调用 mulpolyndown()、printpolyn()输出两多项式降幂相减结果。

2. 创建一元多项式子模块

该模块设计比较简单，创建一元多项式的过程相当于创建一个单链表，采用尾插法，输入多项式中一个项的系数和指数，然后就生成一个结点，建立起它的左指针，并用头结点指向它。用 for 循环语句来判断要输入的项数，并且如果多项式的项数大于一项时，调用 hebing()函数对一元多项式合并同类项。

3. 输出一元多项式子模块

（1）如果多项式的第一项的系数和指数都不等于 0，则输出：系数值 x^指数值，如果指数等于 0 并且系数不等于 0，则输出：系数值；

（2）如果不是多项式的第一项的时候，当其系数和指数都不等于 0 时，且系数大于 0 时，则输出：+系数值 x^指数值，当指数等于 0 且系数不等于 0 时，且系数大于 0 时，则输出：+系数值；

4. 判断多项式是否稀疏子模块

首先将一元多项式降幂排列，将头结点的指针用 p 表示，当 p 存在且指向 p 的下一个结点的指针存在的时候，如果当每一项都满足 p->data.zhishu-p->next->data.zhishu>1 时，则多项式是稀疏多项式，否则该多项式是稠密多项式。

5. 两多项式相加子模块

它从两个多项式的头部开始，两个多项式的某一项都不为空时，如果指数相等的话，系数就应该相加，相加的和不等于 0 的时候，用头插法建立一个新的结点。p 的指数小于 q 的指数的时候,就应该复制 q 结点到多项式中。p 的指数大于 q 的指数的时候，就应该复制 p 的结点到多项式中。当第二个多项式为空时，第一个多项式不为空时，将第一个多项式用新结点产生，当第一个多项式为空时，第二个多项式不为空时，将第二个多项式用新结点产生。

6. 两多项式相减子模块

它从两个多项式的头部开始，两个多项式的某一项都不为空时，如果指数相等的话，系数就应该相加，相加的和不等于 0 的时候，用头插法建立一个新的结点。p 的指数小于 q 的指数的时候,就应该复制 q 结点到多项式中。p 的指数大于 q 的指数的时候，就应该复制 p 的结点到多项式中。并且建立的结点的系数为原来的相反数。当第二个多项式为空时，第一个多项式不为空时，将第一个多项式用新结点产生，当第一个多项式为空时，第二个多项式不为空时，将第二个多项式用新结点产生，并且建立的结点的系数为原来的相反数。

7. 两多项式相乘子模块

第一个多项式的第一项分别与第二个多项式的各项相乘，相乘时系数相乘指数相加，然后第一个多项式的第二项与第二个多项式的各项相乘，以此类推，直到第一个多项式的最后一项与第二个多项式的各项相乘，用了 for 循环语句嵌套 for 循环语句实现此

算法。

8. 合并同类项子模块

第一个多项式的第一项与第二个多项式中指数相同的项相加，然后第一个多项式的第二项与第二个多项式中指数相同的项相加，以此类推，直到第一个最后一项与第二个多项式中指数相同的项相加，用了 for 循环语句嵌套 for 循环语句实现此算法。

9. 升幂排列子模块

对多项式的各项指数用了冒泡排序算法，冒泡排序的基本原理是对存放原始数据的数组，按从前往后的方向进行多次扫描，每次扫描称为一趟。当发现相邻的两个数据的次序不是递增的次序时，即将这两个数据进行互换。这样，较大的数据就会沉底，重复上面过程，直到整个多项式的每一项的指数成递增顺序排列。

10. 降幂排列子模块

对多项式的各项指数用了冒泡排序算法，冒泡排序的基本原理是对存放原始数据的数组，按从前往后的方向进行多次扫描，每次扫描称为一趟。当发现相邻的两个数据的次序不是递减的次序时，即将这两个数据进行互换。这样，较大的数据就会沉底，重复上面过程，直到整个多项式的每一项的指数成递减顺序排列。

动态链式存储结构设计如下：

1. 主函数模块

创建两个空的 vector<Term> P1 和 P2，然后读取用户输入的第一个多项式的每一项，将其存储在 P1 中，读取用户输入的第二个多项式的每一项，将其存储在 P2 中。接着调用 is_dense 函数判断这两个多项式是否为稠密多项式，并输出结果。

然后依次调用 add_poly、subtract_poly 和 multiply_poly 函数计算两个多项式的和、差和积，将结果分别存储在 sum、diff 和 product 中，然后使用 print_poly 函数输出升幂形式的结果和降幂形式的结果。

接下来打开文件 output.txt，将第一个多项式的信息和其升幂形式的结果输出到文件中，然后将第二个多项式的信息和其升幂形式的结果输出到文件中，最后将两个多项式的和、差和积的信息和升幂形式的结果输出到文件中。如果无法打开文件 output.txt，则输出错误信息并退出程序。

最后关闭文件 output.txt，返回 0，表示程序正常退出。

2. 多项式加法模块

该函数是一个一元多项式加法的实现，输入为两个多项式 A 和 B，输出它们的和 C。在函数开始时，创建一个空的 vector<Term> C，用于存储多项式的和。然后设置两个指针 i 和 j 分别指向多项式 A 和 B 的第一项，进入循环，当 i 和 j 分别小于 A 和 B 的长度时执行以下操作：

如果 A[i].degree == B[j].degree，则将 A[i].coeff 和 B[j].coeff 相加得到 coeff，如果 coeff 不为 0，则将 {A[i].degree, coeff} 加入 C 中，然后 i 和 j 都加 1。如果 A[i].degree < B[j].degree，则将 A[i] 加入 C 中，然后 i 加 1。如果 A[i].degree > B[j].degree，则将 B[j] 加入 C 中，然后 j 加 1。当其中一个指针超过了对应的多项式长度时，将另一个多项式剩余的项全部加入 C 中。最后返回 C，表示两个多项式的和。

3. 多项式减法模块

该函数是一个一元多项式减法的实现，输入为两个多项式 A 和 B，输出它们的差

C。在函数开始时，创建一个 vector<Term> neg_B，表示 B 的相反数，遍历 neg_B 中的每一项，将其系数取反。然后调用 add_poly 函数计算 A 和 neg_B 的和，得到 C。最后返回 C，表示两个多项式的差。

4. 多项式乘法模块

该函数是一个一元多项式乘法的实现，输入为两个多项式 A 和 B，输出它们的积 C。在函数开始时，创建一个 vector<Term> C，用于存储多项式的积。然后使用两个 for 循环遍历 A 和 B 中的每一项，分别计算它们的次数和系数的乘积，并将结果存储在 degree 和 coeff 中。接着使用一个 for 循环遍历 C 中的每一项，如果找到次数为 degree 的项，则将其系数加上 coeff。如果在 C 中没有找到次数为 degree 的项，并且 coeff 不为 0，则将 {degree, coeff} 加入 C 中。最后返回 C，表示两个多项式的积。

5. 多项式输出模块

该函数是一个输出一元多项式的实现，输入为一个一元多项式 P，以及两个可选参数：一个布尔值 ascending，表示是否按照次数升序排列，默认为 true；一个输出流对象 out，表示输出的目标，默认为标准输出流 cout。在函数开始时，创建一个 vector<Term> sorted_P，表示按照次数排序后的多项式 P，使用选择排序算法对 sorted_P 进行排序，排序方式由参数 ascending 决定。然后遍历 sorted_P 中的每一项，依次输出多项式的每一项，若某一项系数为 0，则跳过该项。如果不是第一项且该项系数为正数，则在系数前添加一个加号。如果该项系数不为 1 或者该项次数为 0，则输出该项系数。如果该项次数大于 0，则输出 x，如果该项次数大于 1，则输出一个上标符号和该项次数。最后换行输出。

6. 判断多项式是否稠密模块

该函数是一个判断一元多项式是否稠密的实现，输入为一个一元多项式 P，以及一个可选参数 density_threshold，表示密度阈值，默认为 0.5。

在函数开始时，声明两个变量 degree 和 term_count，分别表示最高次项次数和非零项数，初始值都为 -1。

然后使用一个 for 循环遍历 P 中的每一项，如果该项系数不为 0，则将 term_count 加 1，并检查该项次数是否大于 degree，如果大于，则将 degree 更新为该项次数。

接着计算多项式的密度 density，即 term_count 除以 degree+1。

最后判断 density 是否大于等于 density_threshold，如果是则返回 true，表示多项式稠密，否则返回 false，表示多项式稀疏。

4 调试分析

遇到的问题：

- 输入格式错误

在输入多项式时，用户可能会输入错误的格式，例如多项式的项数不正确、系数和指数不是数字等，这会导致程序无法正确解析多项式。可以在输入时，对用户输入进行校验，如果发现输入错误，可以提示用户重新输入。

- 命名空间错误

最开始运行代码时，出现'cout' : undeclared identifier 的错误，发现最开始没有定义 using namespace std，如果不定义这个，必须 std::cout、std::cin 这样写输入输出，如果定义后，cin、cout 前面的 std 可以省略。

- 输出格式错误

在输出最终的表格时，文本无法对其且出现换行的情况，是因为在输出多项式的函数中执行了 cout << \n;的语句。

5 用户使用说明

运行程序后出现如下图的界面：

表 1

输入多项式后会对多项式是否稀疏进行判断：

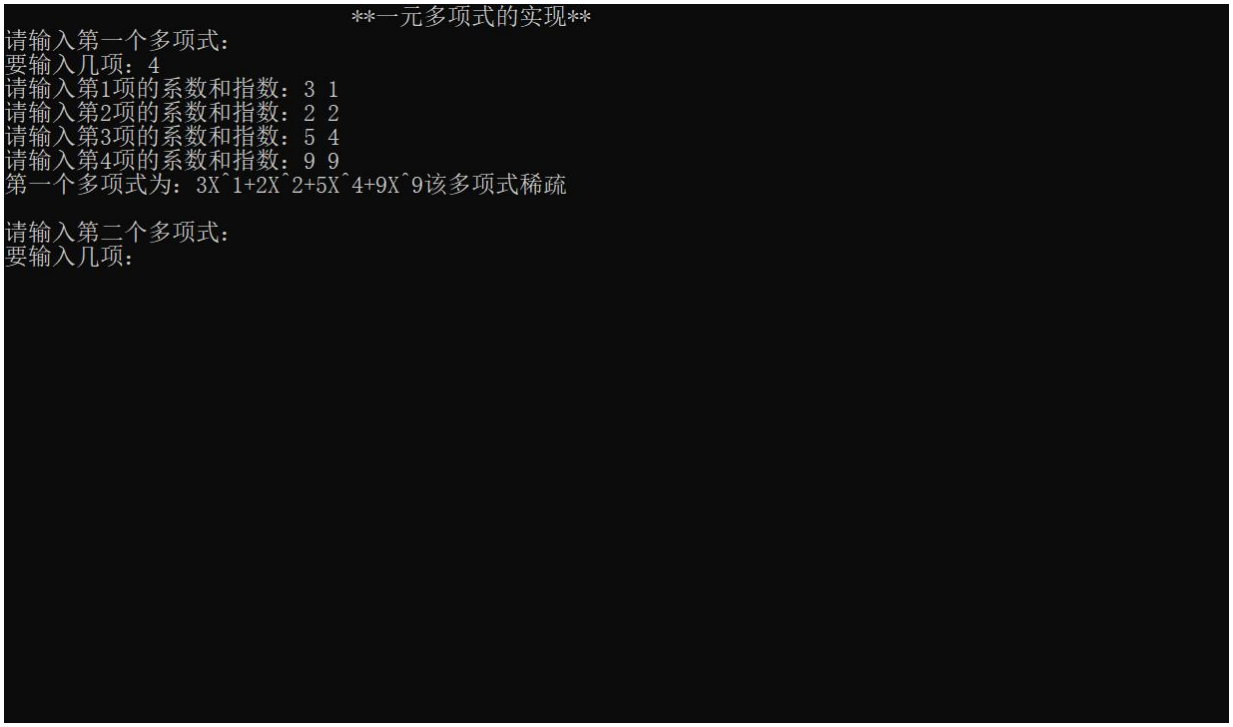


表 2

输入第二个多项式并判断是否稀疏，计算两个多项式的相加（升幂）、相加（降幂）、相减（升幂）、相减（降幂）、相乘（升幂）、相乘（降幂），并将结果同时输出到控制台和文件中

```
要输入几项：4
请输入第1项的系数和指数：3 1
请输入第2项的系数和指数：2 2
请输入第3项的系数和指数：5 4
请输入第4项的系数和指数：9 9
第一个多项式为：3X^1+2X^2+5X^4+9X^9该多项式稀疏

请输入第二个多项式：
要输入几项：3
请输入第1项的系数和指数：1 1
请输入第2项的系数和指数：-2 2
请输入第3项的系数和指数：5 3
第二个多项式为：1X^1-2X^2+5X^3该多项式稀疏

================================================
              操作结果
================================================
操作          升幂结果       降幂结果
------------------------------------------------
加法          4X^1+5X^3+5X^4+9X^99X^9+5X^4+5X^3+4X^1
------------------------------------------------
减法          2X^1+4X^2-5X^3+5X^4+9X^92X^1+4X^2-5X^3+5X^4+9X^9
------------------------------------------------
乘法          3X^2-4X^3+11X^4+15X^5-10X^6+25X^7+9X^10-18X^11+45X^123X^2-4X^3+11X^4+15X^5-10X^6+25X^7+9X^10-18X^11+45X^1
2
------------------------------------------------


--------------------------------
Process exited after 126.3 seconds with return value 0
请按任意键继续. . .
```

表 3

顺序结构的运行结果如下

```
请输入第一个多项式的系数和指数（以空格分割，输入 0 0 结束）：
3 1 2 2 5 4 9 9 0 0
请输入第二个多项式的系数和指数（以空格分割，输入 0 0 结束）：
1 1 -2 2 5 3 0 0
第一个多项式为：3x+2x^2+5x^4+9x^9
该多项式稀疏
第二个多项式为：x-2x^2+5x^3
该多项式稠密
两个多项式的和为：
4x+5x^3+5x^4+9x^9
9x^9+5x^4+5x^3+4x
两个多项式的差为：
2x+4x^2-5x^3+5x^4+9x^9
9x^9+5x^4-5x^3+4x^2+2x
两个多项式的积为：
3x^2-4x^3+11x^4+15x^5-10x^6+25x^7+9x^10-18x^11+45x^12
45x^12-18x^11+9x^10+25x^7-10x^6+15x^5+11x^4-4x^3+3x^2


--------------------------------
Process exited after 35.96 seconds with return value 0
请按任意键继续. . .
```

表 5

6 课程设计总结

通过这次数据结构课程设计，我深刻认识到一个优秀的程序所需要具备的特点，例如清晰明了、可读性好、结构合理、性能良好和容错性强等。本次课程设计让我更加熟悉和掌握了数据结构中的某些算法，也让我认识到编程的复杂性和耗时性。

虽然我完成了这个项目，但由于时间和数据结构知识的限制，我的程序的健壮性仍不够。在今后的学习中，我将会继续努力学习数据结构和算法知识，不断提升自己的编程能力和实践经验。我还会注重各方面能力的协调发展，选择一两门技术进行深入研究，成为一个既可以统筹全局，又有一定技术专长的优秀的程序开发人员。

在完成这个项目的过程中，我也意识到团队合作的重要性。虽然这个项目是我个人完成的，但是我也从同班同学中学到了很多东西，他们对我提出的问题进行耐心讲解，让我受益匪浅。感谢老师的指导和教诲，让我在这个项目中收获了很多。同时，我也要感谢同班同学的支持和帮助，让我在这个项目中感受到了团队的力量。

总之，这次数据结构课程设计是一次难得的机会，让我更加深入地了解了数据结构和算法，并提高了自己的编程能力和实践经验。在未来的学习和工作中，我将会继续努力，不断提升自己的技能和能力，为实现自己的梦想和目标做出更大的贡献。

7 测试结果

下面对几组数据进行测试。

第一个多项式：2x^2 + 3x + 1 第二个多项式：3x^3 - 2x^2 + x - 1		
操作	升幂	降幂
加法	3x + 2x^2 + 3x^3	3x^3 + 2x^2 + 3x
减法	3x - 2x^2 - 3x^3	-3x^3 + 4x^2 + 3x
乘法	-x + 7x^2 + 5x^3 + 2x^4 + 6x^5	6x^5 + 5x^4 + 7x^3 - 2x^2 + 4x - 1
第一个多项式：-2x^3 + 3x^2 + 5x - 1 第二个多项式：x^3 - 2x^2 + 3x - 4		
操作	升幂	降幂
加法	8x - x^2 - x^3	-x^3 + x^2 + 8x - 5
减法	-2x + 5x^2 + 3x^3	-3x^3 + 5x^2 + 2x + 3
乘法	-4 - 7x + 2x^2 + 5x^3 - 11x^4 + 7x^5 - 2x^6	-2x^6 + 7x^5 - 11x^4 + 17x^3 - 7x^2 + 11x - 4
第一个多项式：-x^4 + 3x^2 - 2x + 1 第二个多项式：2x^3 - 4x^2 + 5x - 1		
操作	升幂	降幂
加法	-x + 2x^3 - x^2 - x^4	-x^4 + 2x^3 - x^2 + 3x
减法	x - 2x^3 - x^4 + 7x^2	-x^4 - 2x^3 + 7x^2 - 3x + 2
乘法	7x - 5x^2 + 8x^3 - 2x^4 - 5x^5 + 10x^6 - 2x^7	-2x^7 + 10x^6 - 5x^5 - 2x^4 + 8x^3 - 7x^2 + 7x - 1

8 参考书目

[1] 严蔚敏，吴伟民. 数据结构（C 语言版）[M]. 清华大学出版社，2011.

[2] 邓俊辉. 数据结构与算法（C++语言版）[M]. 清华大学出版社，2012.

[3] Aditya Bhargava. 算法图解[M]. 人民邮电出版社, 2017.

[4] 维基百科. (2022 年 5 月 21 日). 稠密集 [网页]. Retrieved 2022 年 6 月 21 日, from https://zh.wikipedia.org/wiki/稠密集

[5] 维基百科. (2022 年 6 月 8 日). 多項式 [网页]. Retrieved 2022 年 6 月 21 日, from https://zh.wikipedia.org/zh-hans/%多项式

[6] Duye. (2016, 9 月 1). 数据结构之线性表（顺序存储结构和链式存储结构） [博客文章]. Retrieved 2022 年 6 月 21 日, from https://www.cnblogs.com/duye/p/6196571.html