华中科技大学

课 程 实 验 报 告

课程名称： 数据结构实验

实验名称： 基于链表结构线性表的操作实现

专业班级：_____

学　　号：_____

姓　　名：_____

指导教师：_____

报告日期：_____

网络空间安全学院

目 录

1 链式循环队列的设计

1.1 需求分析

1.1.1 功能需求

以链表作为物理结构，使用 C 语言编程实现循环队列的基本功能，可参考约瑟夫环的设计。依据循环队列的功能，实现对输入/输出数据进行入队/出队操作，能够处理满队/空队等边界情况。

1.1.2 输入输出需求

使用链表建立容量为 8 的循环空队列，下标依次记为 0,1,2...,7,指定起始位置（队首）下标 i，读入数据后从起始位置 i 顺序存储整数序列，队列满队(读入 8 个元素)之后停止读入；输出时，如果输出个数大于存储最大容量 8 则输出"Error"，否则如果输出个数大于已存储元素个数(即会发生空队列输出)则输出"Fault"；如果队列已满则先输出"Full"，然后从起始位置开始顺序输出 k 个数据，否则直接顺序输出 k 个数据。

输入形式：非负整数序列 n,i,k,e1,e2,...,en。(n 为读入元素个数,i 为队首下标,k 为输出个数)

输出形式：非负整数序列"Full",e1',e2',...,ek'或"Error"或"Fault"。（"Full"只有在队满时才会输出，后面是循环队列中存储的 k 元素）

1.2 总体设计

整个程序分为建立循环队列并初始化和入队/出队处理两个部分。

1.3 算法设计

链式队列入队/出队系统如图 1-1 所示。

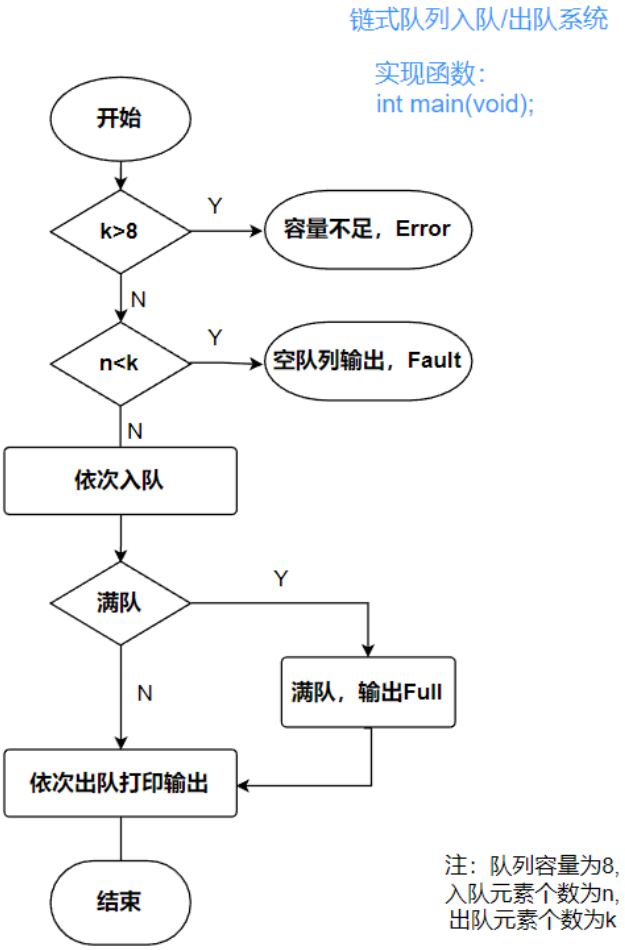


图 1-1 链式队列入队/出队系统

1.4 系统实现

本程序全程在 Microsoft Visual Studio 2022 上编写、编译、调试、运行，并最终在 Educoder 平台上运行通过。

主要函数以及功能如表 1-1 所示。

函数名	主要功能
int main()	实现队列入队和出队功能
void init()	完成链式队列的创建与初始化

表 1-1 主要函数及功能

1.5 系统测试

支持 Educoder 平台的所有可见测试用例与隐藏测试用例，均通过，如图 1-2 所示。

图 1-2 通过所有测试

1.6 结果分析

成功通过所有的给定测试用例，表明该链式循环队列操作程序设计成功实现使用需求。（若存在测试用例没有通过，试分析原因，下同）

1.7 实验小结

略。（在实验中遇到的问题，以及解决方案等，下同）

2 链式栈建栈与操作

2.1 需求分析

2.1.1 功能需求

以链表作为物理结构，使用 C 语言编程实现栈的基本功能。依据栈的功能，实现对输入/输出数据进行入栈/出栈操作，能够处理溢出等边界情况。

2.1.2 输入输出需求

建立链式栈(长度=5)来存储多项式序列并合并，之后逆序输出。

输入形式：

按幂次升序输入多项式序列

c1 e1 c2 e2...cn en ; c1' e1' c2' e2'...cn' en'

输出形式：

将两多项式合并，按幂次降序输出多项式序列。

注：

两多项式序列中没有相等的幂次（即不存在 1<=i<=n，1<=j<=n'，使得 ei=ej'）。其中 ci,ci'为整数，代表系数；ei,ei'为整数，代表指数；';'为分隔符。

2.2 总体设计

整个程序分为字符串读取系统，链表合并系统 2 个部分。

字符串读取系统将读入的字符串数据转化为数值存入链表的结点中，同时实现对于中断标志‘;’的判断。

链表合并系统将两个多项式链表进行按顺序比较合并，实现幂次降序排列。

2.3 算法设计

字符串读取系统，链表合并系统分别如图 2-1,2-2 所示。

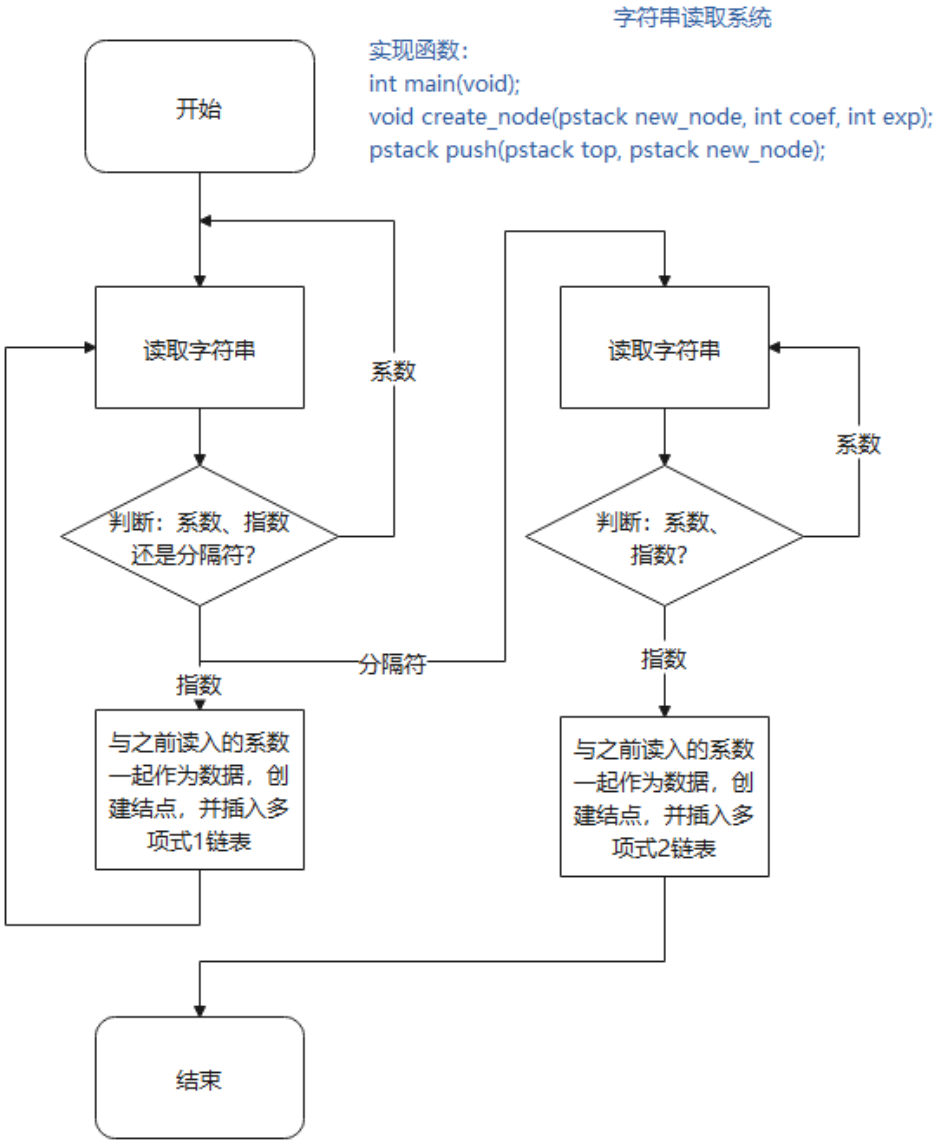

字符串读取系统
实现函数:
int main(void);
void create_node(pstack new_node, int coef, int exp);
pstack push(pstack top, pstack new_node);
开始
读取字符串
判断：系数、指数还是分隔符?
系数
指数
分隔符
与之前读入的系数一起作为数据，创建结点，并插入多项式1链表
读取字符串
判断：系数、指数?
系数
指数
与之前读入的系数一起作为数据，创建结点，并插入多项式2链表
结束

图 2-1 字符串读取系统

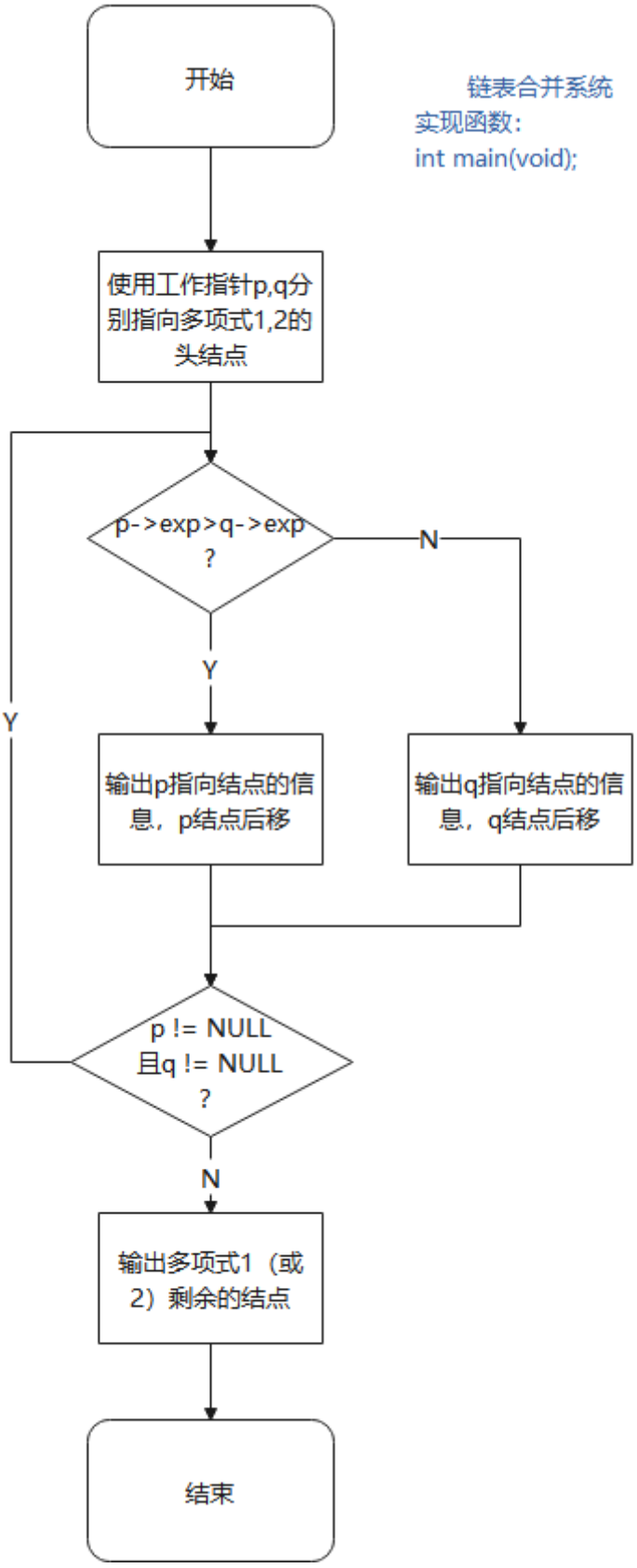

开始
链表合并系统
实现函数：
int main(void);
使用工作指针p,q分别指向多项式1,2的头结点
p->exp>q->exp ?
N
Y
Y
输出p指向结点的信息，p结点后移
输出q指向结点的信息，q结点后移
p != NULL 且q != NULL ?
N
输出多项式1（或2）剩余的结点
结束

图 2-2 链表合并系统

2.4 系统实现

本程序全程在 Microsoft Visual Studio 2022 上编写、编译、调试、运行，并最终在 Educoder 平台上运行通过。

主要函数以及功能如表 2-1 所示。

函数名	主要功能
int main()	实现读取多项式、合并多项式、输出多项式功能
void create_node(pstack new_node, int coef, int exp)	为多项式的一项创建结点
pstack push(pstack top, pstack new_node)	完成入栈操作

表 2-1 主要函数及功能

2.5 系统测试

支持 Educoder 平台的所有可见测试用例与隐藏测试用例，均通过，如图 2-3 所示。

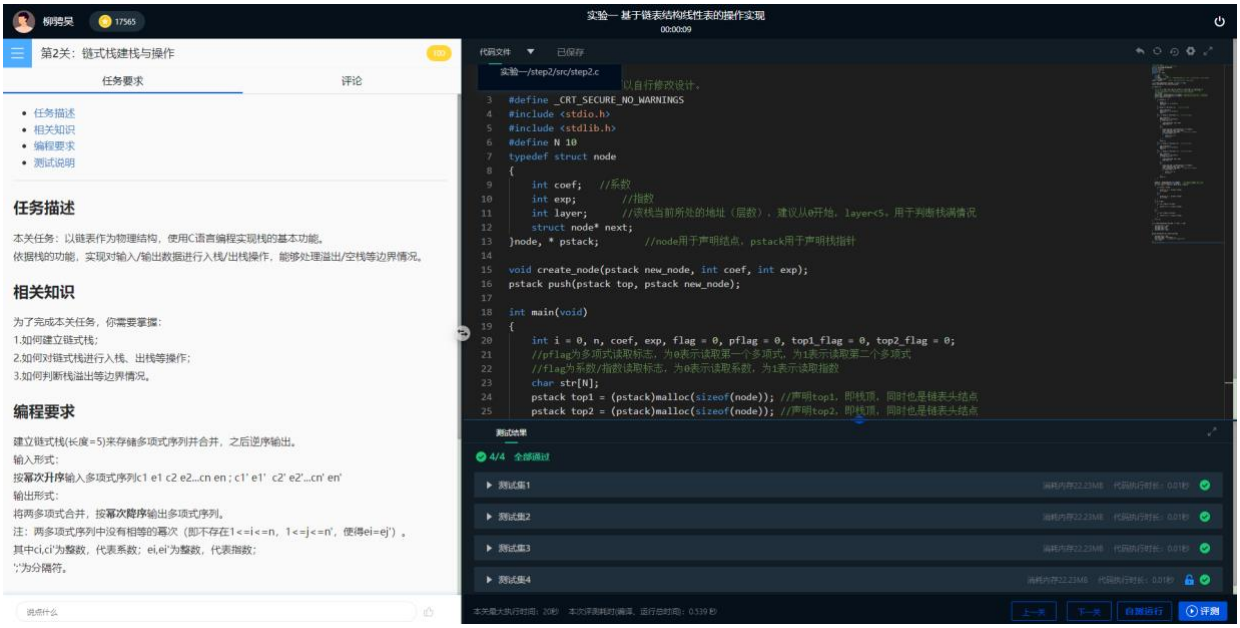


图 2-3 测试通过

2.6 结果分析

成功通过所有的给定测试用例，表明该链式栈操作程序设计成功实现使用

需求。

2.7 实验小结

略。

3 一元稀疏多项式加减运算

3.1 需求分析

3.1.1 功能需求

对两个一元稀疏多项式进行加减运算，运算结果按照幂次从高到低依次输出多项式的幂次和系数。

3.1.2 输入输出需求

对两个一元稀疏多项式进行加减运算。

输入形式：

c1 e1 c2 e2..... cn1 en1 ± c1' e1' c2' e2'.....cn2' en2'；ci，ci'为浮点数，分别为两个多项式第 i 项的系数；ei，ei'为整数，分别为两个多项式第 i 项的指数，指数乱序排列；±为加号或减号。

输出形式：

将运算后的结果多项式按幂次降序排列，依次输出结果多项式的系数和指数。

3.2 总体设计

整个程序分为字符串读取系统，多项式链表排序系统，多项式链表加减系统 3 个部分。

字符串读取系统将读入的字符串数据转化为数值存入链表的结点中，同时实现对于加减符号‘±’的判断。

多项式链表排序系统分别将两个多项式按幂次降序排序，为之后的加减运算做准备。

多项式链表加减系统对两个多项式进行加减。

3.3 算法设计

字符串读取系统，多项式链表排序系统，多项式链表加减系统分别如图 3-1,3-2,3-3 所示。

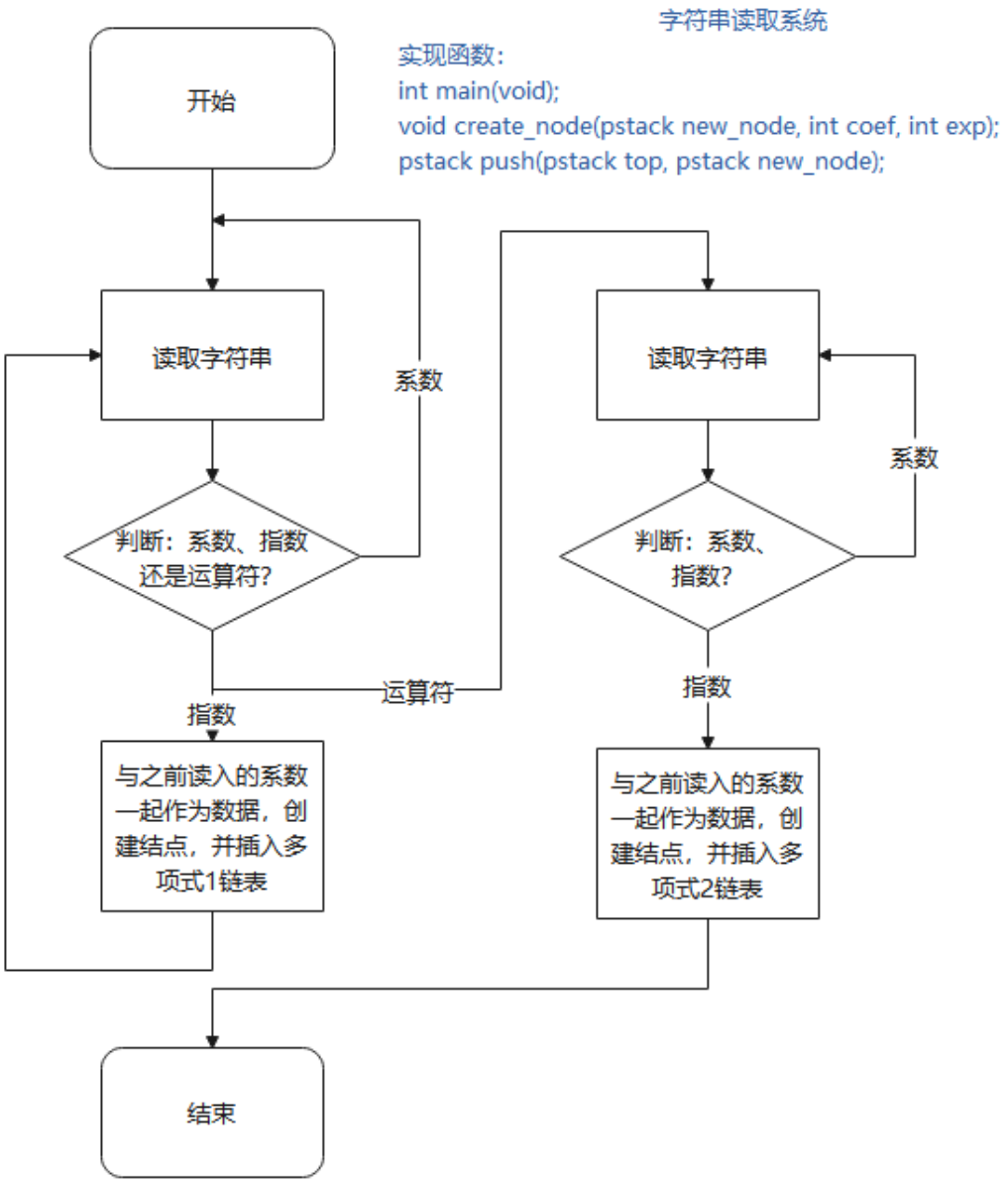
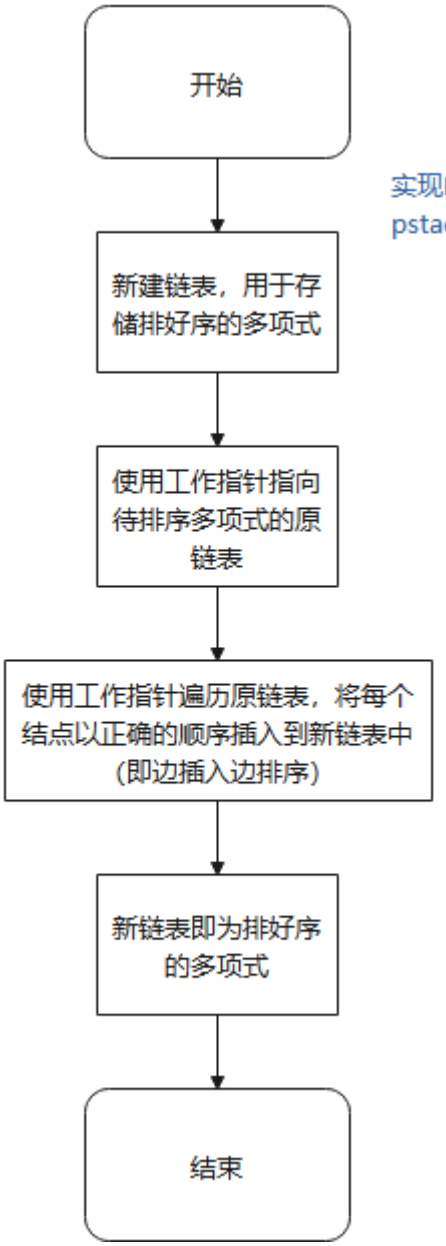

字符串读取系统
实现函数：
int main(void);
void create_node(pstack new_node, int coef, int exp);
pstack push(pstack top, pstack new_node);
开始
读取字符串
判断：系数、指数还是运算符?
系数
运算符
指数
与之前读入的系数一起作为数据，创建结点，并插入多项式1链表
读取字符串
判断：系数、指数?
系数
指数
与之前读入的系数一起作为数据，创建结点，并插入多项式2链表
结束

图 3-1 字符串读取系统



多项式链表排序系统
实现函数：
pstack poly_sort(pstack top);

图 3-2 多项式链表排序系统

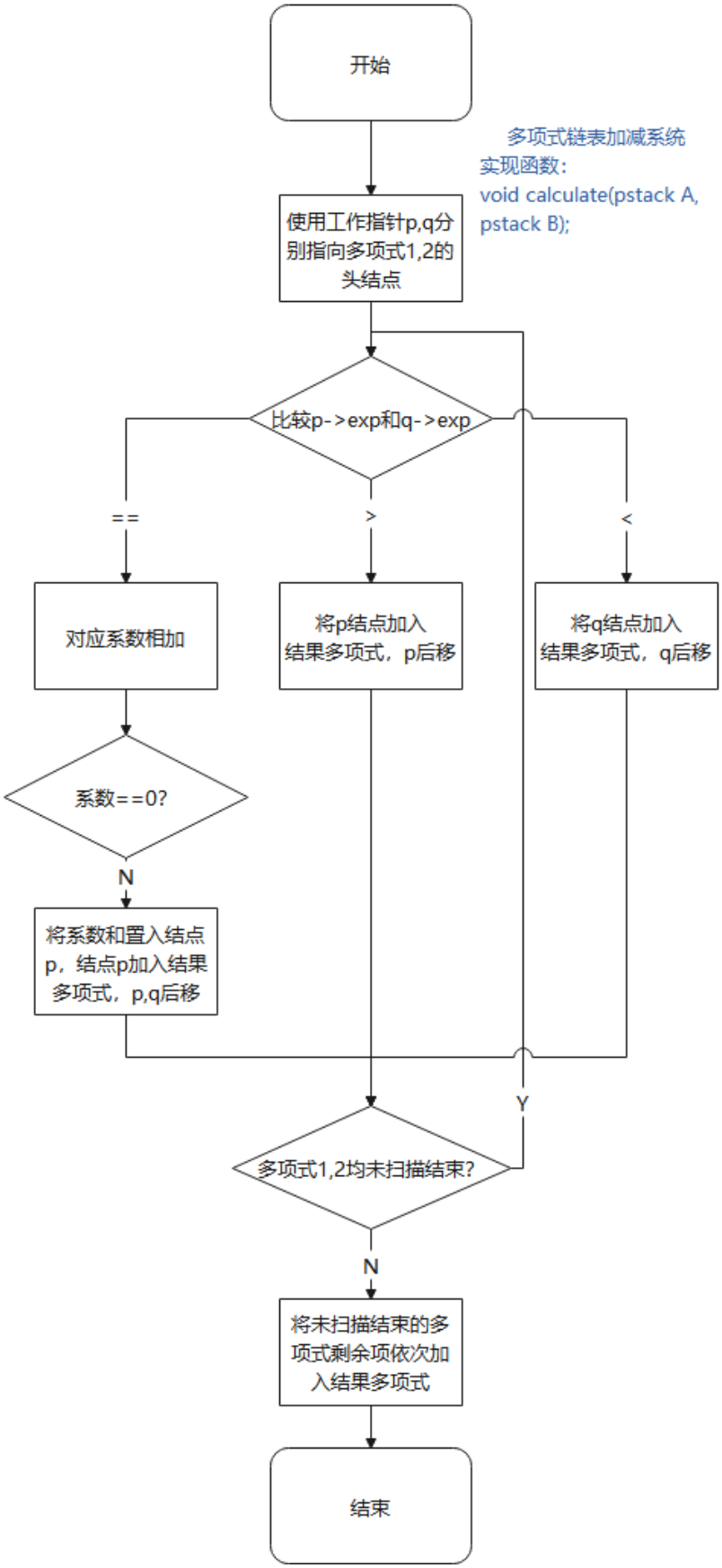


图 3-3 多项式链表加减系统

3.4 系统实现

本程序全程在 Microsoft Visual Studio 2022 上编写、编译、调试、运行，并最终在 Educoder 平台上运行通过。

主要函数以及功能如表 3-1 所示。

函数名	主要功能
int main()	实现读取多项式、输出多项式功能
pstack poly_sort(pstack top)	实现对多项式进行排序的功能
void calculate(pstack A, pstack B);	实现对两个多项式进行加法运算的功能
void display(pstack A)	实现按 ci，ei 格式输出多项式各项
void create_node(pstack new_node, int coef, int exp)	实现创建结点的功能
pstack push(pstack top, pstack new_node)	实现结点入栈的功能

表 3-1 主要函数及功能

3.5 系统测试

支持 Educoder 平台的所有可见测试用例与隐藏测试用例，均通过，如图 3-4 所示。

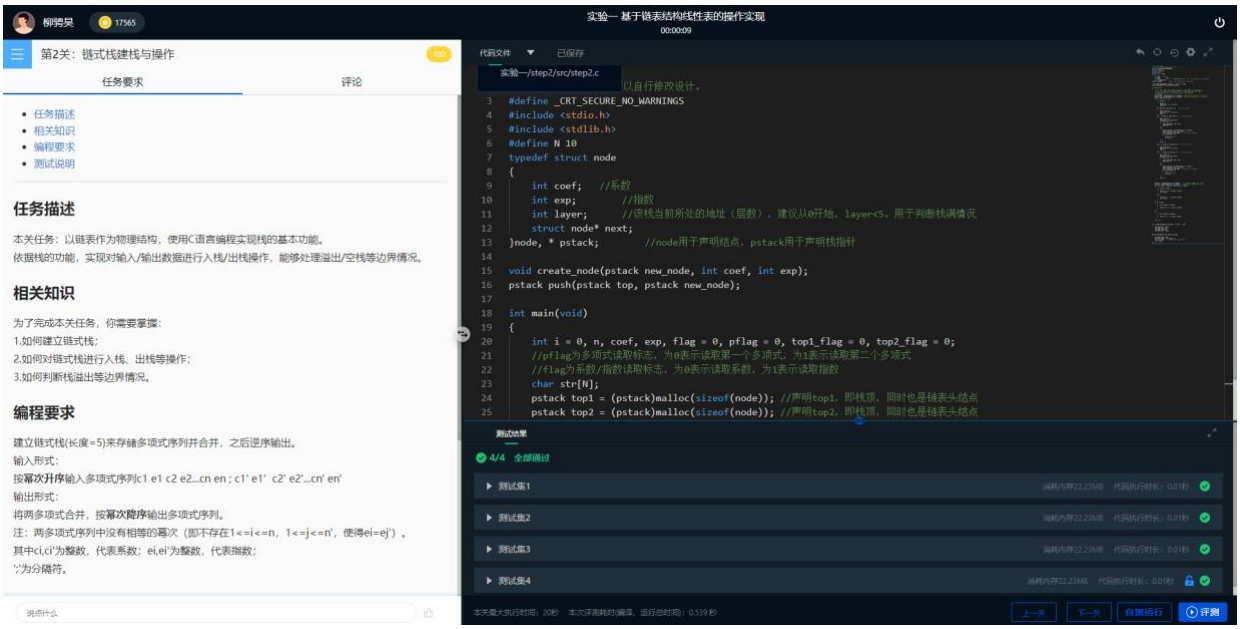


图 3-4 测试通过

3.6 结果分析

成功通过所有的给定测试用例，表明该一元稀疏多项式加减运算器成功实现使用需求。

3.7 实验小结

略。

4 一元稀疏多项式运算的升级功能

4.1 需求分析

4.1.1 功能需求

对一元稀疏多项式的加减运算提供升级功能，满足以下几点：

可以实现自然语言格式的输入输出；可以处理输入项的幂次有重复的情况；可以处理输入项的次序是按幂次乱序的情况。

4.1.2 输入输出需求

对两个一元稀疏多项式进行加减运算。

输入形式：(c1x^e1±c2x^e2±...±cnx^en1)±(c1'x^e1'±c2'x^e2'±...±cn'x^en2')

ci，ci'为浮点数，分别为两个多项式第 i 项的系数；

ei，ei'为整数，分别为两个多项式第 i 项的指数；

多项式按幂次乱序排列。

输出形式： c1x^e1±c2x^e2±...±cnx^en1 按幂次降序排列

注意，输出形式需满足常见的数学多项式的书写形式。 比如如下特殊情况：

当系数为±1 且指数不为 0 时，1 省略；

当指数为 0 时，x 和指数省略；

当系数为 0 时，该项省略；

加一个负数，需要表示成减一个正数；

等等。

4.2 总体设计

整个程序分为字符串匹配转换系统、多项式链表排序系统，多项式链表加减系统、字符串输出系统 4 个部分。

字符串匹配转换系统将用户输入的自然表达式转换为链表，每个链表保存一项的信息，从表头至表尾呈降幂排列（如果输入不为降幂会进行自动调整，如果输入中存在两项次数相同会进行自动合并，如果输入为 0 则表为空）。

字符串输出系统将结果多项式的链表进行输出。

4.3 算法设计

字符串匹配转换系统与字符串输出系统如图 4-1,4-2 所示，多项式链表排序系统和多项式链表加减系统已在第三关报告中给出，此处不再次给出。

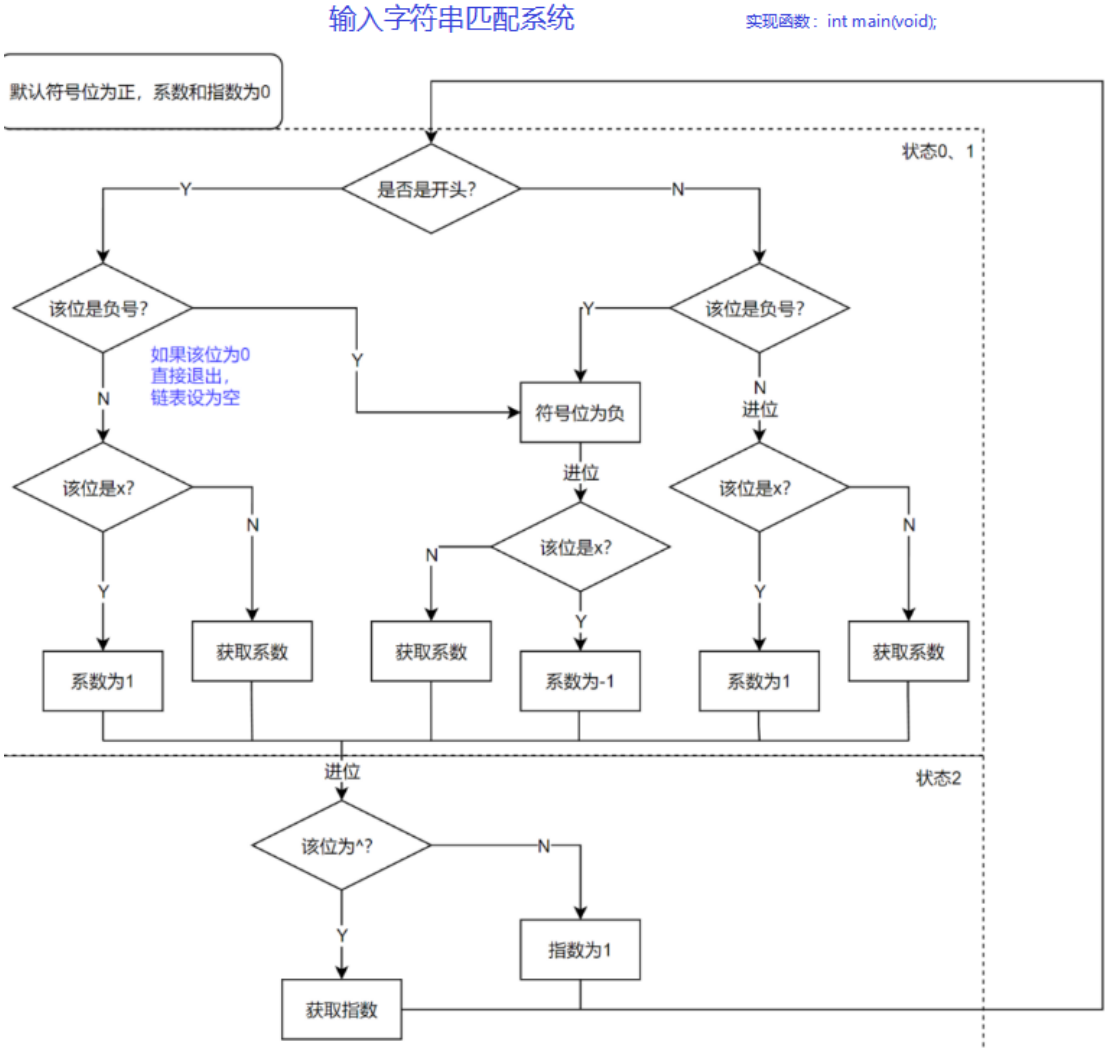


图 4-1 输入字符串匹配系统

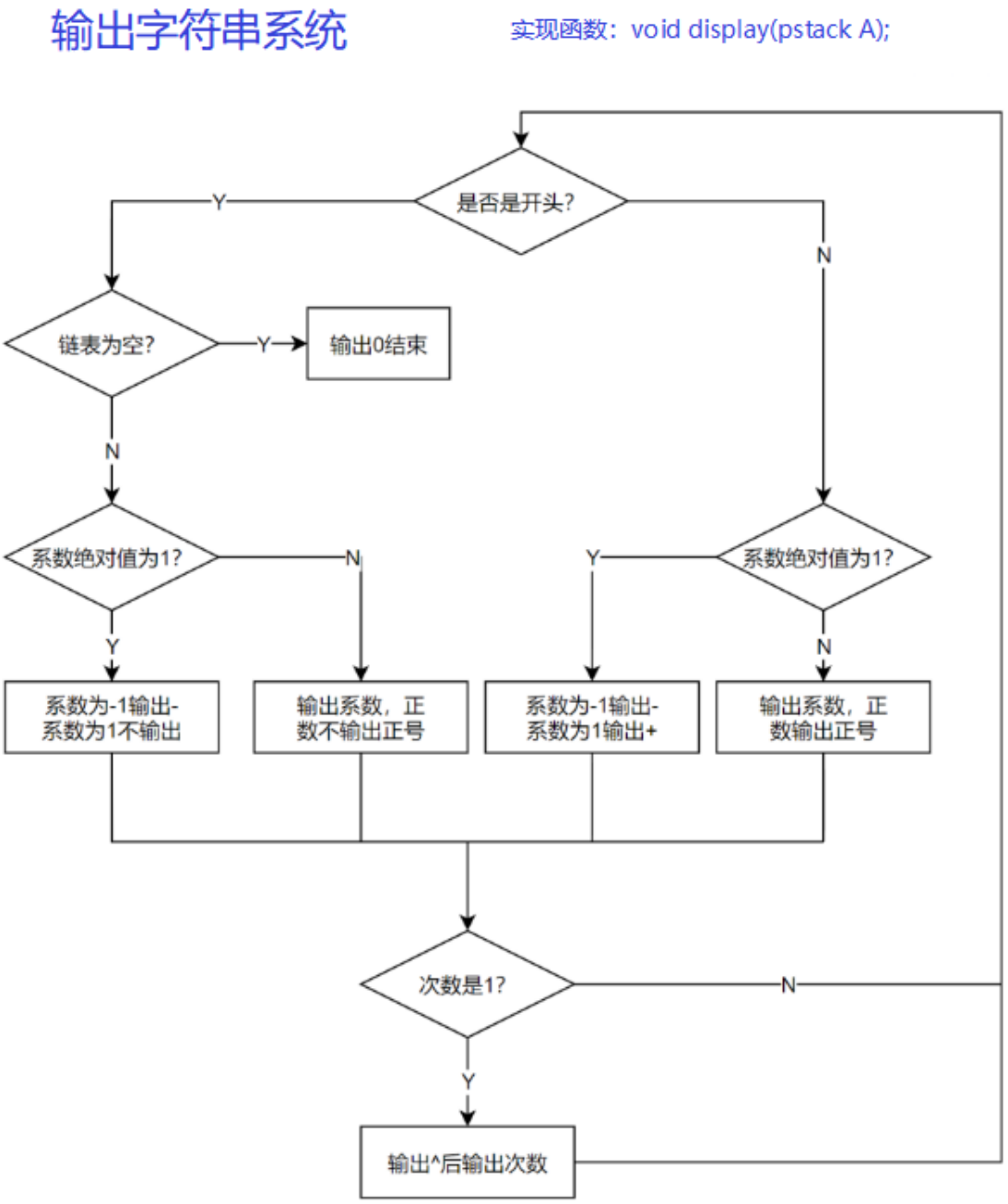


图 4-2 输出字符串系统

4.4 系统实现

本程序全程在 Microsoft Visual Studio 2022 上编写、编译、调试、运行，并最终在 Educoder 平台上运行通过。

主要函数以及功能如表 4-1 所示。

函数名	主要功能
int main()	实现按自然语言格式读取多项式的功能，并调

	用其他系统中的函数
pstack poly_sort(pstack top)	实现对多项式进行排序的功能
void calculate(pstack A, pstack B);	实现对两个多项式进行加法运算的功能
void display(pstack A)	实现自然语言格式输出多项式各项
void create_node(pstack new_node, int coef, int exp)	实现创建结点的功能
pstack push(pstack top, pstack new_node)	实现结点入栈的功能

表 4-1 主要函数及功能

4.5 系统测试

支持 Educoder 平台的所有可见测试用例与隐藏测试用例，均通过，如图 4-3 所示。

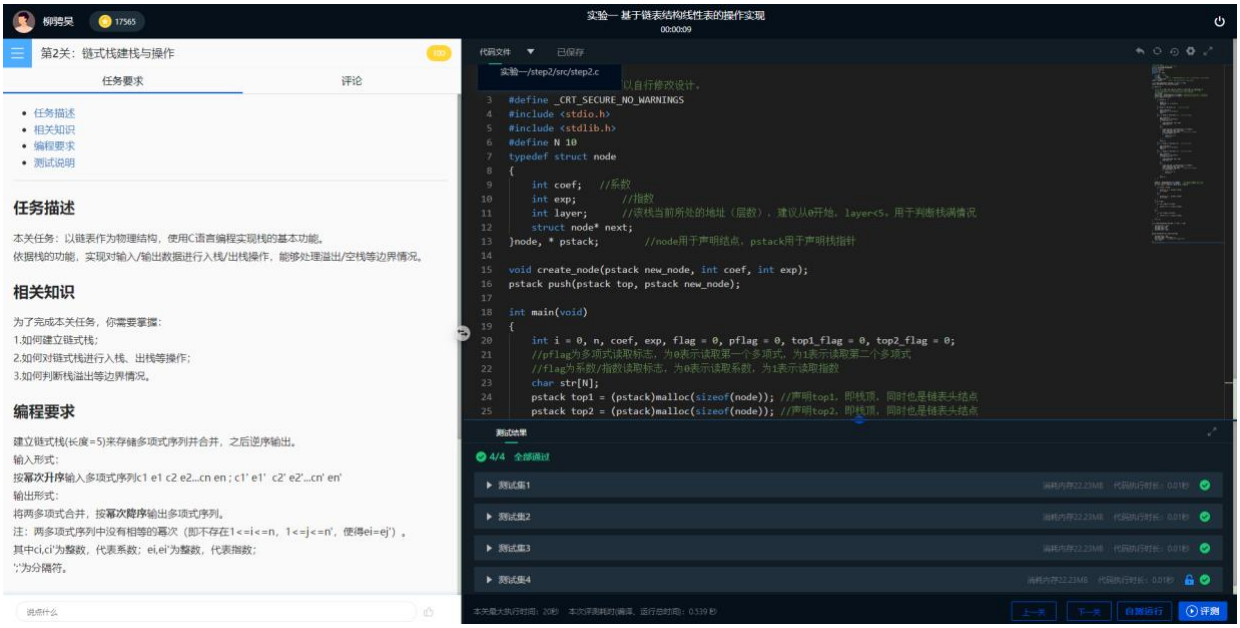


图 4-3 测试通过

4.6 结果分析

成功通过所有的给定测试用例，表明该一元稀疏多项式加减运算器的升级功能需求成功实现。

4.7 实验小结

略。

参考文献

[1] 严蔚敏等. 数据结构(C 语言版). 清华大学出版社

[2] Larry Nyhoff. ADTs, Data Structures, and Problem Solving with C++. Second Edition, Calvin College, 2005

[3] 严蔚敏等.数据结构题集(C 语言版). 清华大学出版社

附录 基于链式存储结构线性表实现的源程序

略。