沈阳航空航天大学

**课 程 设 计 报 告**

课程设计名称：数据结构课程设计

课程设计题目：希尔排序的具体实现

|  |  |
| --- | --- |
| 学 院 | 计算机学院 |
| 专 业 | 计算机科学与技术 |
| 班 级 | 计算机2003 |
| 学 号 | 203401010319 |
| 姓 名 | 路世显 |
| 指导教师 | 李飞 |

2021年12月24日

**目 录**

[1 题目介绍 1](#_Toc19817)

[1.1 问题描述 1](#_Toc866)

[1.1.1 问题背景 1](#_Toc26858)

[1.1.2 主要任务 1](#_Toc12456)

[1.2 问题分析 1](#_Toc1120)

[2 系统总体设计 2](#_Toc3216)

[3 数据结构与算法设计 3](#_Toc7533)

[3.1 数据结构设计 3](#_Toc5181)

[3.2 算法设计 3](#_Toc647)

[4 系统运行与测试 6](#_Toc4777)

[4.1 调试及调试分析 6](#_Toc6070)

[4.2 测试用例 6](#_Toc21732)

[5 总 结 11](#_Toc12873)

[参考文献 12](#_Toc11646)

[附 录 （程序清单） 13](#_Toc19236)

1. 题目介绍
   1. 问题描述

首先，如何将一组整数按照输入的方式保存下来并且能够随时调用，即确定自己本系统所使用的数据结构。其次，要对整体的系统有框架一样的设计，保证每一步都是可知的。最后，要完美的掌握希尔排序的算法逻辑和C语言的具体实现。

* + 1. 问题背景

希尔排序按其设计者希尔（Donald Shell）的名字命名，该算法由希尔1959年公布。一些老版本教科书和参考手册把该算法命名为Shell-Metzner，即包含Marlene Metzner Norton的名字，但是根据Metzner本人的说法，“我没有为这种算法做任何事，我的名字不应该出现在算法的名字中。”

希望通过使学生编写希尔排序的具体步骤，使学生更加深刻地体会希尔排序的真实运作算法。

* + 1. 主要任务

1. 输入一组整数序列并以链表的方式存储下来。
2. 直接进行希尔排序并将结果展示出来。
3. 进行希尔排序并进行逐步的演示。
   1. 问题分析

首先，设计数据结构时，考虑到直接使用数组会导致存储空间的浪费，所以采用了链表作为存储的数据结构。其次，想要将希尔排序的每一步都具体的演示出来，则更需要对希尔排序的每一步了如指掌，知道按照什么分组，按照什么对每一个分组进行排序，套用了选择排序，并且同时掌握选择排序的运行原理和算法设计。最后，实现如何使用链表式的存储结构来进行希尔排序的算法的编写。

1. 系统总体设计

通过问题分析和老师指导后，设计系统总体结构，将整个系统划分为4个模块，分别是输入保存模块，直接排序不演示，排序并演示与退出。

输入模块采用了一个单链表作为存储结构，对输入的一组整数序列进行保存，并且使用一个结构体指针数组来指向链表的每一个节点方便之后的对数组排序的算法的编写。直接排序不演示与排序并演示模块的核心算法是一样的，都是通过指针数组的调用来实现数据域的排序。差别在于直接排序不演示则是不会将每一步的结果打印在屏幕上。退出模块则是控制系统的进行和结束。

1. 数据结构与算法设计
   1. 数据结构设计

一个地址不一定连续的单向链表：

typedef struct node

{

int num;

node \*next;

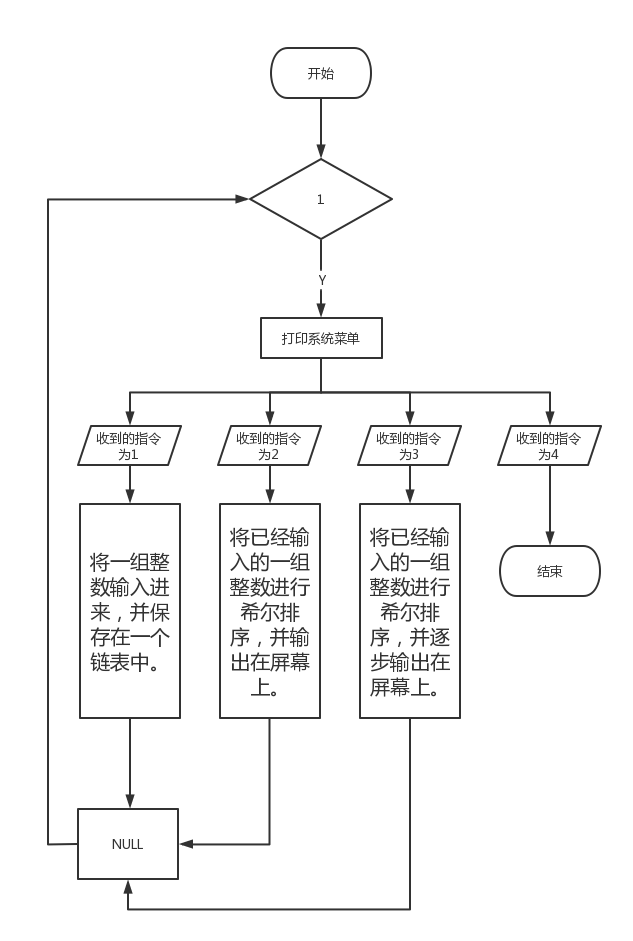
}node;

一个存储node\*地址的连续地址的数组。

node \*\*a=new node[length];

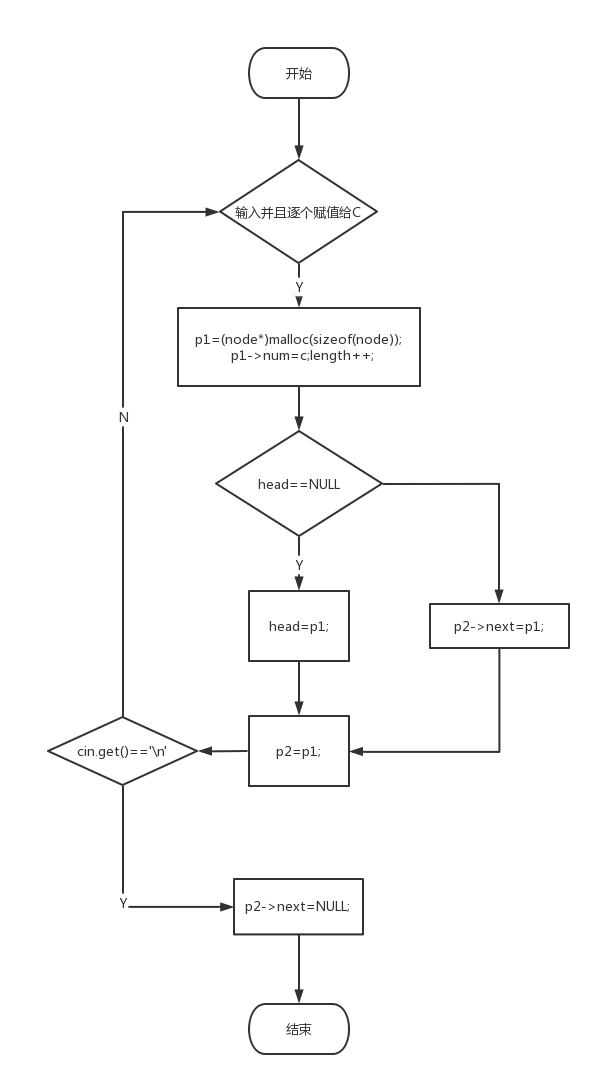
* 1. 算法设计

总体框图：



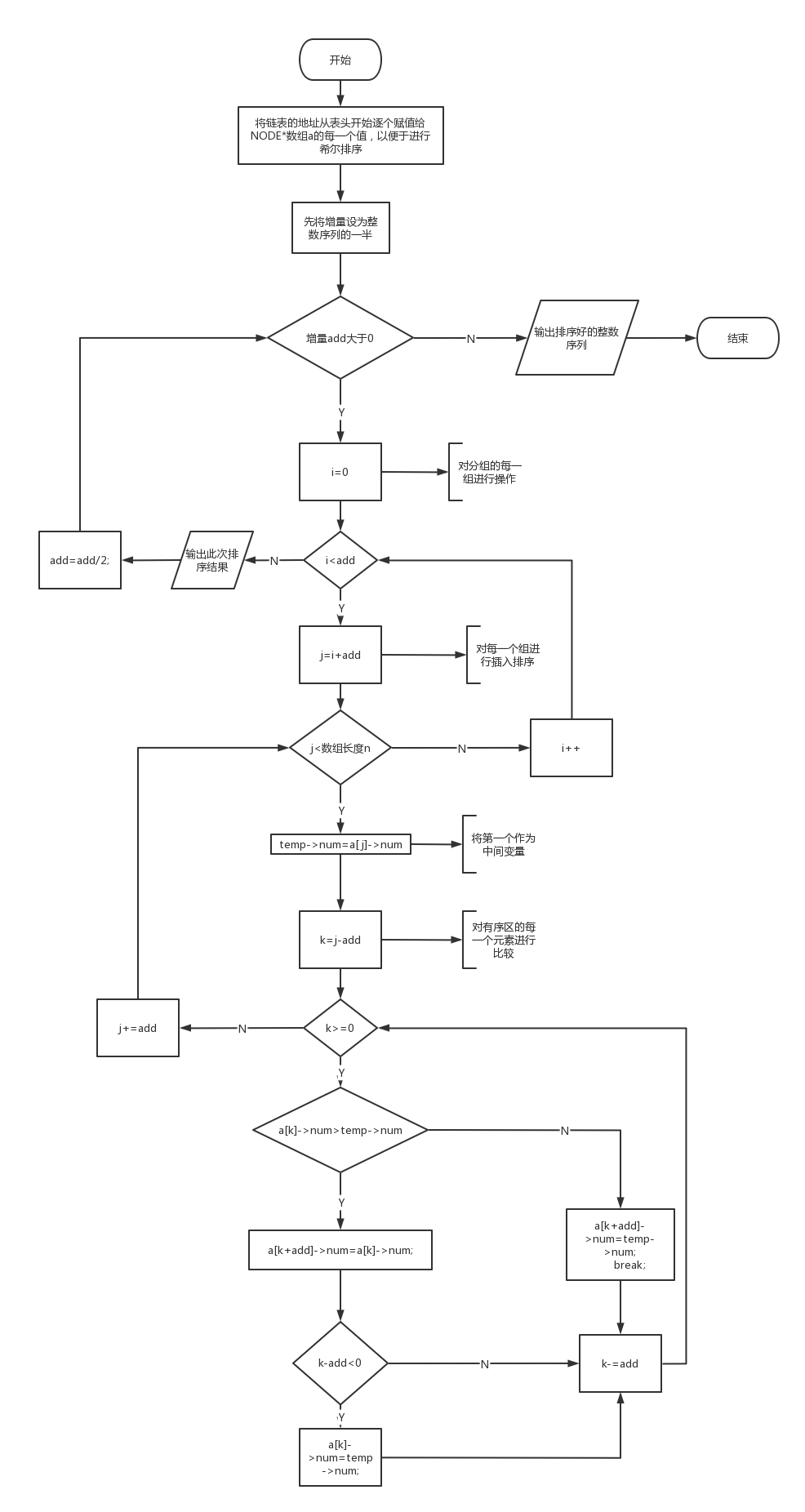
**图3.1 输入流程图**

从键盘输入一组整数序列，并且存储在一个链表中：



**图3.2 输入流程图**

链希尔排序的具体演示：



**图3.3 希尔排序具体实现流程图**

1. 系统运行与测试
   1. 调试及调试分析
2. 链表的排序问题

问题描述：直接在链表的存储位置上进行线性表的插入，删除

解决方法：自己建立了一个指向链表结构体的指针数组，将从头节点到尾节点的地址存储在这个数组中，直接通过在该数组上的调用比较来直接改变链表存储位置的值，从而实现链表结构的排序。

1. 系统设计问题

问题描述：作为初学者，对于系统的建立还停留在做题层面。

解决方法：带着电脑来到老师的办公室，亲自询问老师，经过老师的指导，得知了一个健康的系统是需要划分成各个模块的，使得源程序的代码不至于像是一团，对于后续的修改和优化提供了便捷。

* 1. 测试用例

1. 本身就是顺序的一组整数进行测试：

测试用例：1 2 3 4 5 6 7 8 9

预期输出：1 2 3 4 5 6 7 8 9

实际输出：

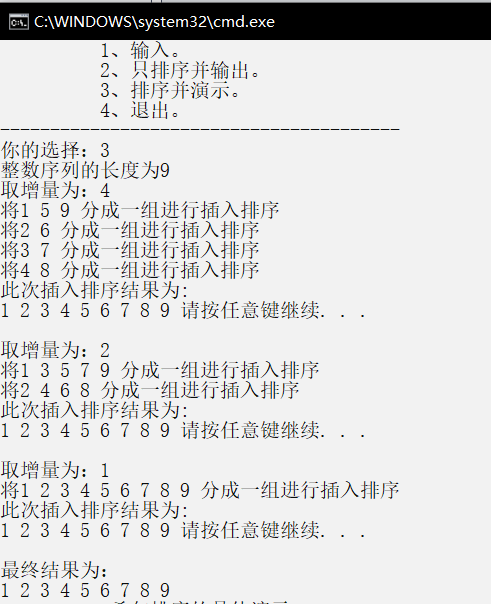


图4.2.1 测试用例1

1. 一组平常的整数序列：

测试用例：9 5 6 2 4 8 3 7 11

预期输出：2 3 4 5 6 7 8 9 11

实际输出：

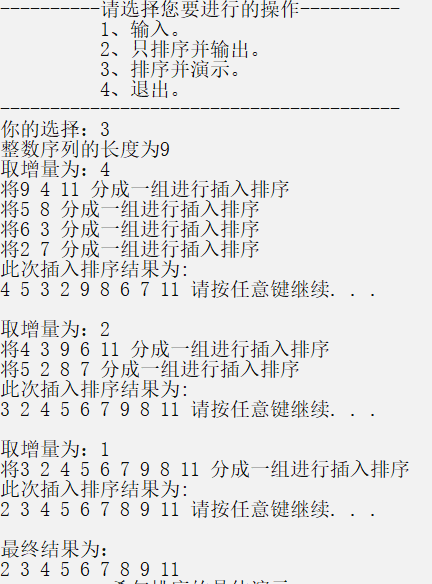


图4.2.2 测试用例2

1. 一组带有重复数字的整数序列

测试用例：2 5 6 3 4 8 6 1 9

预期输出：1 2 3 4 5 6 6 8 9

实际输出：

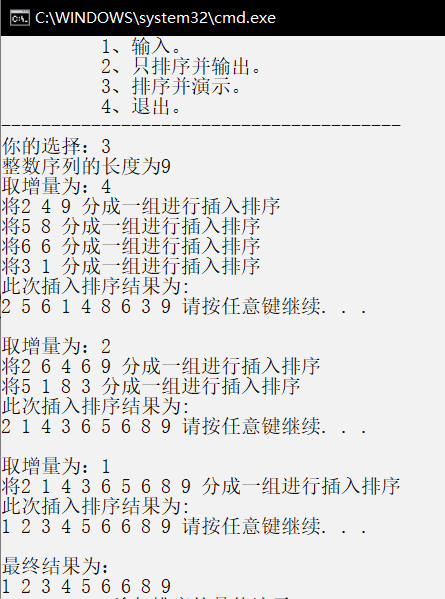


图4.2.3 测试用例3

1. 一组只有一个数的整数序列

测试用例：1

预期输出：1

实际输出：

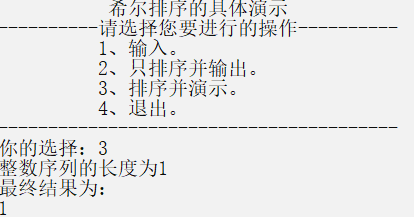


图4.2.4 测试用例4

**表4.1 系统测试**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试项 | 测试用例 | 预期输出 | 实际输出 |
| 一组整数序列 | 1 2 3 4 5 6 7 8 9 10 | 1 2 3 4 5 6 7 8 9 10 | 1 2 3 4 5 6 7 8 9 10 |
| 一组整数序列 | 9 5 6 2 4 8 3 7 11 | 2 3 4 5 6 7 8 9 11 | 2 3 4 5 6 7 8 9 11 |
| 一组整数序列 | 2 5 6 3 4 8 6 1 9 | 1 2 3 4 5 6 6 8 9 | 1 2 3 4 5 6 6 8 9 |
| 一组整数序列 | 1 | 1 | 1 |

1. 总 结

通过这次对希尔排序的具体步骤的体现的实验，使得我在编写的过程中对于排序的算法认知更加清晰，同时在查找并理解希尔排序的时候对十大经典排序算法都有了一定的认知。对于课程设计有了更深层次的认知，了解到了课程设计与平时作业的不同，设计一个系统的复杂度明显要比做出一项简单的作业大。通过阅读文献和查阅资料，不断地修改并检查自身程序的不足之处，既是对自身程序编写能力的提升和体现，也是对自身是否能够找出一个系统的bug的锻炼和培养。使得我们将来在工作的时候能够更好的做出令甲方满意的系统。

# **参考文献**

1. 陈正冲. C语言深度解剖. 北京: 北京航空航天大学出版社, 2010，4
2. 严蔚敏.吴伟民.数据结构.北京:清华大学出版社,1997.4
3. 钱能.C++程序设计教程.北京:清华大学出版社,2019.7
4. 谭浩强.C程序设计(第五版).北京:清华大学出版社，2017，8
5. 和克智.C++程序设计（第二版).西安:西安西安交通大学出版社，1999，8

# **附 录 （程序清单）**

#include<stdio.h>

#include<stdlib.h>

#include<iostream>

using namespace std;

typedef struct node

{

int num;

node \*next;

}node;

int dddd=0;

node \*create(node \*head,int &length)

{

if(dddd==1)

{

cout<<"您ú本?身Θ?已?经-输?入?过y一?次?整?数簓序ò列，?再ù次?输?入?将?会á导?致?上?次?数簓据Y的?丢a失骸?，?是?否?继ì续?（辍?是?，?输?入?1/否?，?输?入?0）?：阰"<<endl;

int tt;

cin>>tt;

switch(tt)

{

case 0:return head;

case 1:head=NULL;length=0;break;

}

}

cout<<"请?输?入?需è要癮排?序ò的?整?数簓序ò列（辍?以?回?车μ结á束?）?：阰"<<endl;

int c;

node \*p1,\*p2=NULL;

while(cin>>c)

{

p1=(node\*)malloc(sizeof(node));

p1->num=c;

length++;

if(head==NULL)

head=p1;

else

p2->next=p1;

p2=p1;

if(cin.get()=='\n')break;

}

p2->next=NULL;

if(head!=NULL)

dddd=1;

return head;

}

void print(node \*head,int &length)

{

if(head==NULL)

{

cout<<"您ú还1没?有瓺输?入?一?组哩?整?数簓序ò列，?这a里?推?荐?您ú先è去ǎ?."<<endl;

return ;

}

node \*\*a,\*p,\*temp=new node,\*head2=NULL,\*p1,\*p2=NULL;

int i=0,add=length/2,j,k;

a=new node\*[length];

for(p=head;p!=NULL;p=p->next)

{

p1=(node\*)malloc(sizeof(node));

p1->num=p->num;

if(head2==NULL)

head2=p1;

else

p2->next=p1;

p2=p1;

}p2->next=NULL;

for(p=head2;p!=NULL;p=p->next)

{

a[i]=p;

i++;

}

while(add>0)

{

for(i=0;i<add;i++)

{

for(j=i+add;j<length;j+=add)

{

temp->num=a[j]->num;

for(k=j-add;k>=0;k-=add)

{

if(a[k]->num>temp->num)

{

a[k+add]->num=a[k]->num;

if(k-add<0)

{

a[k]->num=temp->num;

}

}

else

{

a[k+add]->num=temp->num;

break;

}

}

}

}

add=add/2;

}

for(p=head2;p!=NULL;p=p->next)

{

cout<<p->num<<' ';

}

cout<<endl;

}

void show(node \*head,int &length)

{

if(head==NULL)

{

cout<<"您ú还1没?有瓺输?入?一?组哩?整?数簓序ò列，?这a里?推?荐?您ú先è去ǎ?."<<endl;

return ;

}

node \*\*a,\*p,\*temp=new node;

int i=0,add=length/2,j,k;

a=new node\*[length];

for(p=head;p!=NULL;p=p->next)

{

a[i]=p;

i++;

}

cout<<"整?数簓序ò列的?长¤度è为a"<<length;

while(add>0)

{

cout<<endl<<"取?增?量?为a：阰"<<add<<endl;

for(i=0;i<add;i++)

{

cout<<"将?";

for(int d=i;d<length;d+=add)

{

cout<<a[d]->num<<' ';

}

cout<<"分?成é一?组哩?进?行D插?入?排?序ò"<<endl;

}

for(i=0;i<add;i++)

{

for(j=i+add;j<length;j+=add)

{

temp->num=a[j]->num;

for(k=j-add;k>=0;k-=add)

{

if(a[k]->num>temp->num)

{

a[k+add]->num=a[k]->num;

if(k-add<0)

{

a[k]->num=temp->num;

}

}

else

{

a[k+add]->num=temp->num;

break;

}

}

}

}

cout<<"此?次?插?入?排?序ò结á果?为a:"<<endl;

for(p=head;p!=NULL;p=p->next)

{

cout<<p->num<<' ';

}

add=add/2;

system("pause");

}

cout<<endl<<"最?终?结á果?为a：阰"<<endl;

for(p=head;p!=NULL;p=p->next)

{

cout<<p->num<<' ';

}

cout<<endl;

}

int main()

{

int n,length=0;

node \*head=NULL;

while(1)

{

cout<<" 希￡尔?排?序ò的?具?体?演Y示? "<<endl

<<"----------请?选?择?您ú要癮进?行D的?操ù作痢?---------"<<endl

<<" 1、￠输?入?。￡ "<<endl

<<" 2、￠只?排?序ò并￠输?出?。￡ "<<endl

<<" 3、￠排?序ò并￠演Y示?。￡ "<<endl

<<" 4、￠退?出?。￡ "<<endl

<<"----------------------------------------"<<endl

<<"你?的?选?择?：阰";

cin>>n;

if(n==1)

head=create(head,length);

if(n==2)

print(head,length);

if(n==3)

show(head,length);

if(n==4)

break;

}

return 0;

}