数据结构实验报告

张文

2007-06-29

1 设计人员相关信息

1.1 设计者姓名、学号、班级：

张文
200532530040
2005级信息安全2班

1.2 设计日期：

2007年6月24日

1.3 上机环境：

WindowsXP SP2
Microsoft Visual C++ 6.0

2 程序设计相关信息

2.1 实验题目：

实验题目：11.10.10
设计一个将一组英文单词按字典序排列的基数排序算法。假设单词均由小写字母或空格构成，最长的单词有MaxLen个字母。

2.2 实验项目组成：

11-10-10.cpp
11-10-10.exe
data.txt

2.3 实验项目的程序结构（程序中的函数调用关系图）：

```
1    struct node_type
2    {
3        char *word;
4        struct node_type *next;
5    };
6
```

作为待生成链表的每个结点的类型。

程序函数列表

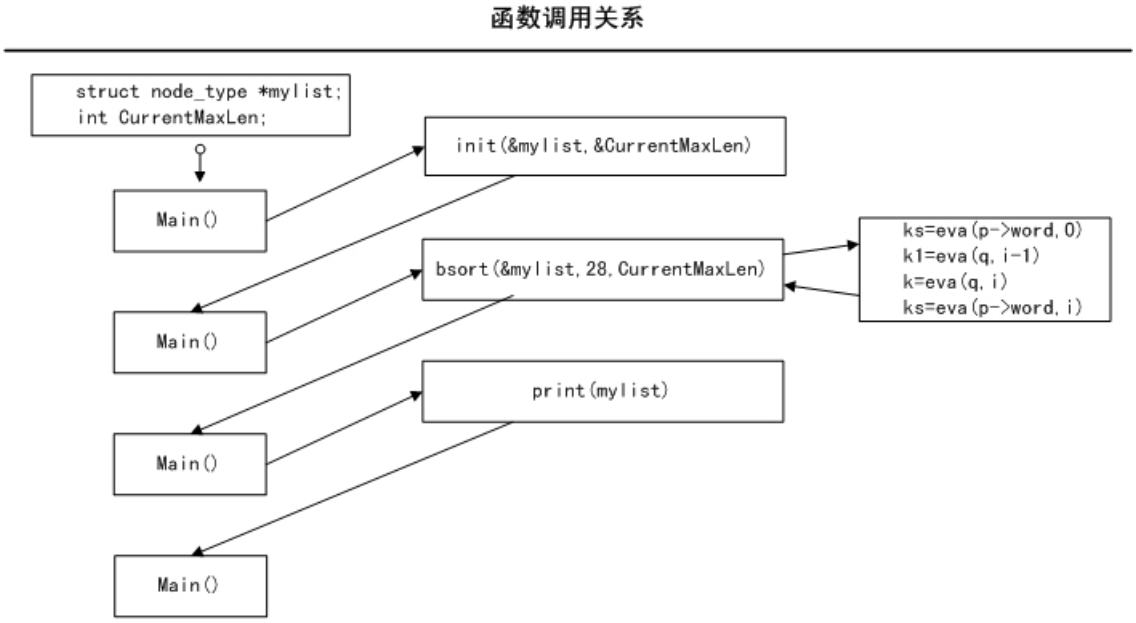
1. void init(struct node_type **ThisList,int *len)
   用途：建立一个链表，将从数据文件(data.txt)中读出待排序的单词，全部存储到这个链表中。并且在待排序的单词中，计算出最长的单词的长度。

2. void print(struct node_type* ThisNode)
   用途：输出已经已经按字典序排序好的链表中的单词。

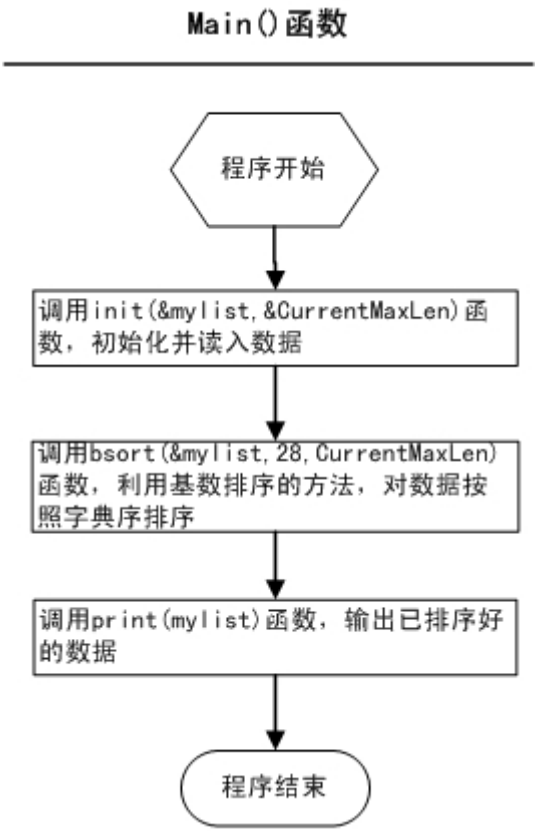
3. void bsort(struct node_type** ThisList,int r,int d)
   用途：使用基数排序法，对初始的单词链表进行排序。

4. int eva(char *str,int i)
   用途：计算待排序单词，在处理第i位时，该单词该放入的桶号。

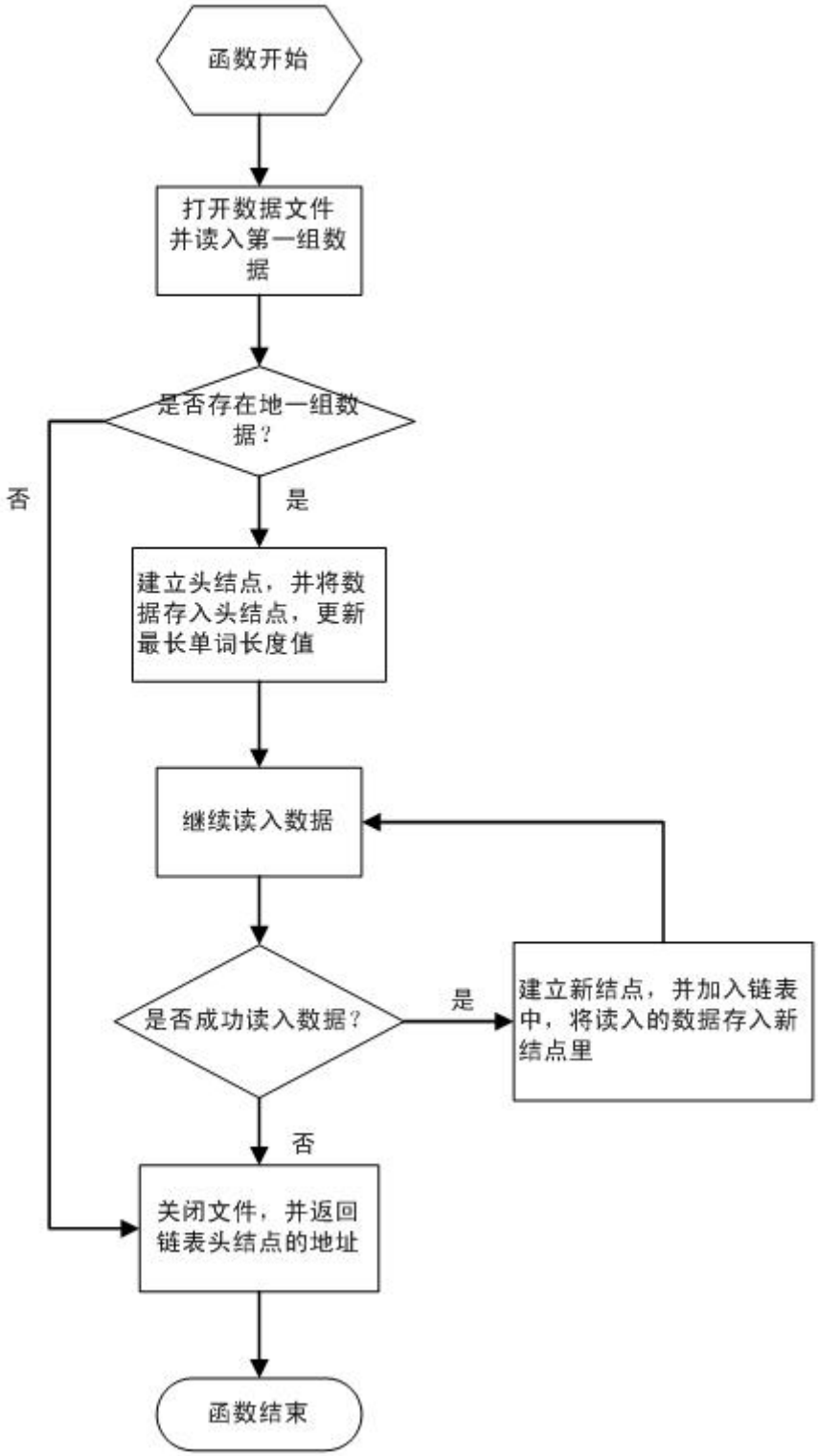
函数关系调用图


函数调用关系
struct node_type *mylist;
int CurrentMaxLen;
Main()
init(&mylist,&CurrentMaxLen)
Main()
bsort(&mylist,28,CurrentMaxLen)
ks=eva(p->word,0)
k1=eva(q,i-1)
k=eva(q,i)
ks=eva(p->word,i)
Main()
print(mylist)
Main()

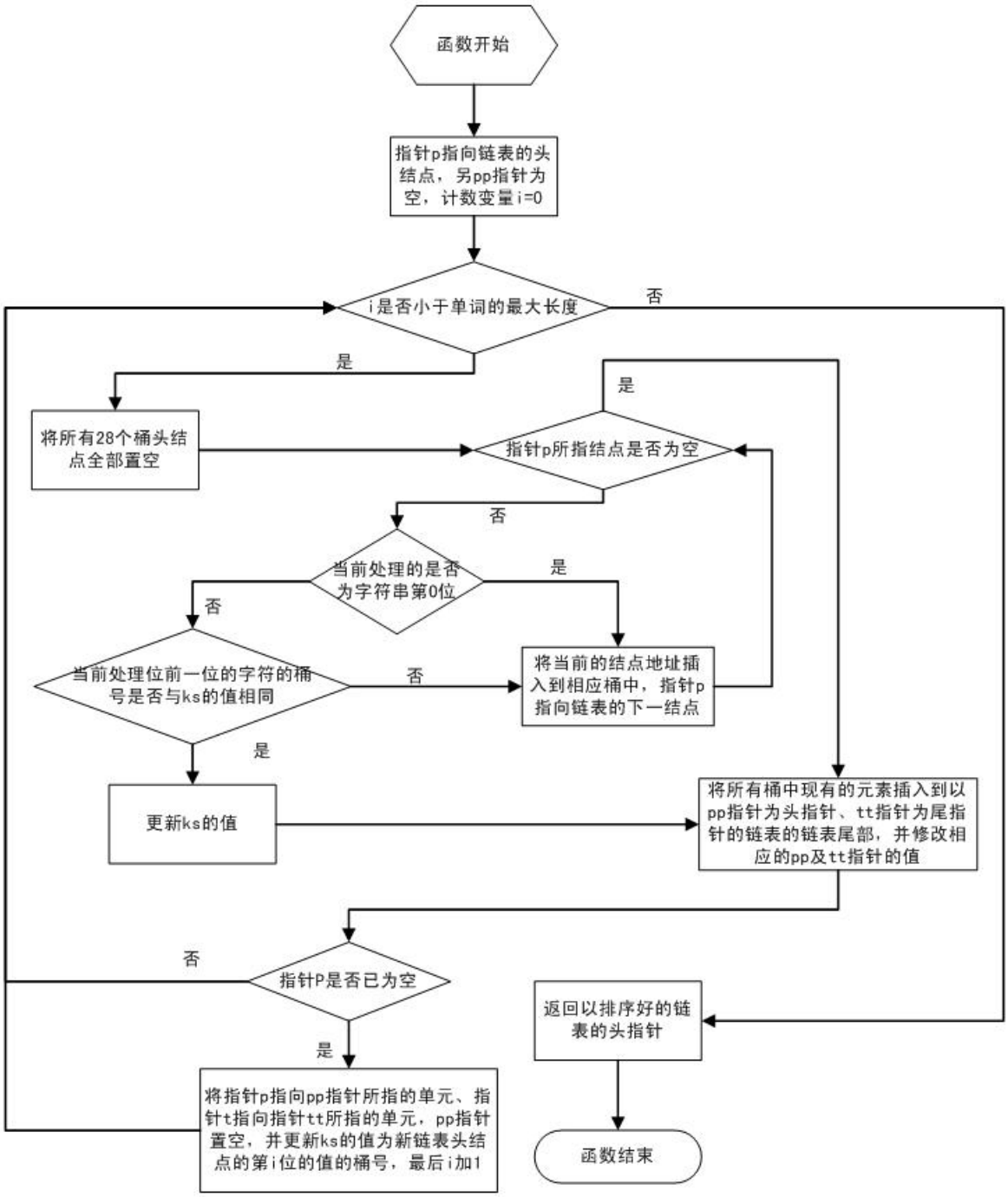
2.4 算法描述或流程图：


Main()函数
程序开始
调用init(&mylist,&CurrentMaxLen)函数，初始化并读入数据
调用bsort(&mylist,28,CurrentMaxLen)函数，利用基数排序的方法，对数据按照字典序排序
调用print(mylist)函数，输出已排序好的数据
程序结束

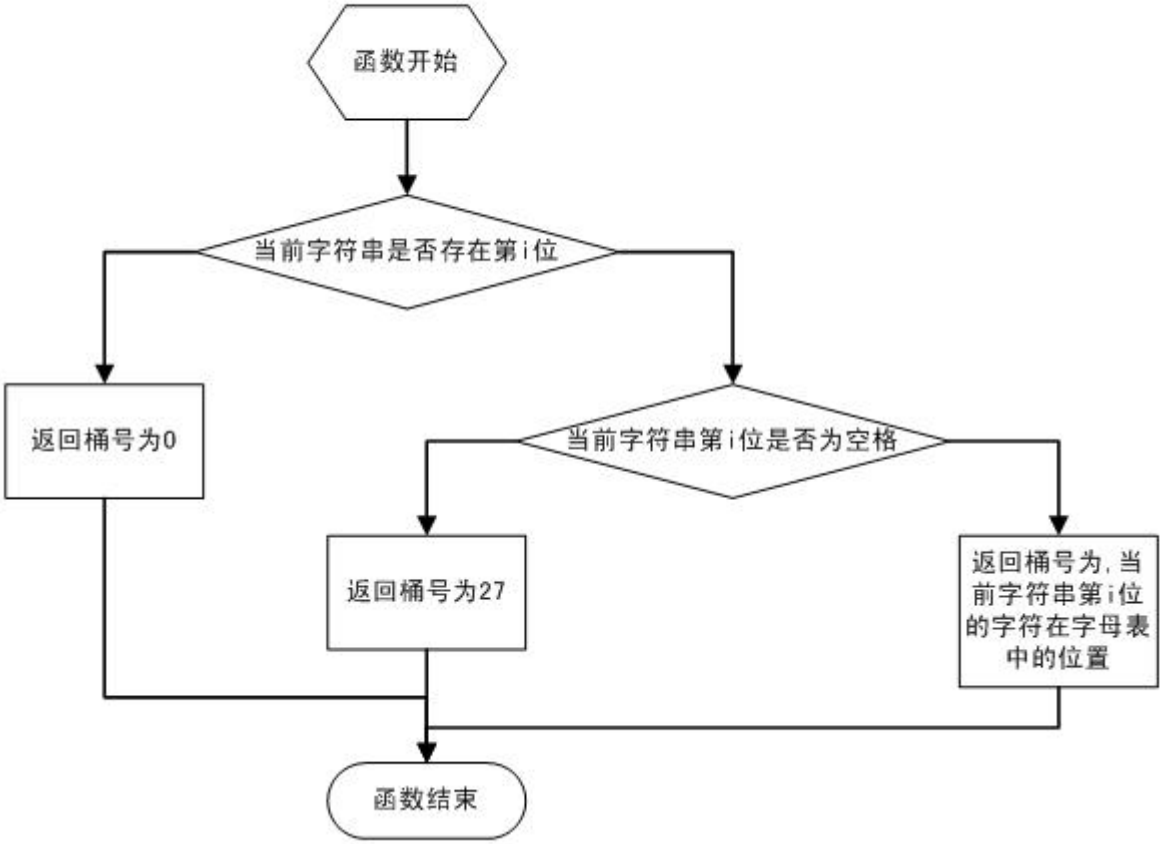
void init(struct node_type **,int *)函数



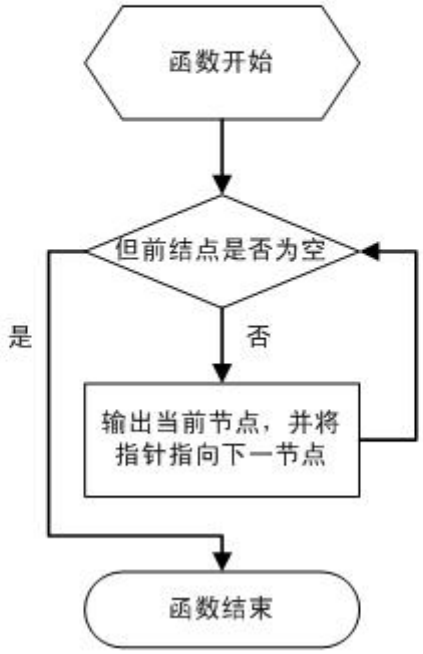
void bsort(struct node_type**, int, int)函数



int eva(char*,int)函数



void print(struct node_type*)函数



2.5 实验数据和实验结果：

1. **输入数据：**data.txt
   i
   you
   he
   she
   it
   a
   they
   ass
   aspace
   i am
   ia m
   输出数据：
   a
   aspace
   ass
   he
   i
   ia m
   it
   i am
   she
   they
   you

实验结果

通过这个实验，加深了我对基数排序的工作原理的理解，学习到了一些链表更为灵活的运用等等。

而通过实验报告的编写，也让我了解了TEX**这款闻名遐迩的排版软件，以及**Microsoft Office Visio**等软件。通过着许许多多知识的学习、联系、与运用，我的能力得到了一定提高。**

3 程序源代码

```
1  #include<stdio.h>
2  #include<stdlib.h>
3  #include<string.h>
4
5  #define MaxLen 20
6  #define MaxR 30
7
8  struct node_type
9  {
10         char *word;
11         struct node_type *next;
12 };
13
14 void init(struct node_type**,int *);
15 void print(struct node_type*);
16 void bsort(struct node_type**,int,int);
17 int eva(char*,int);
18
19 int main()
20 {
21         struct node_type *mylist;
22         int CurrentMaxLen;
23         init(&mylist,&CurrentMaxLen);
24         bsort(&mylist,28,CurrentMaxLen);
25         print(mylist);
```

```
26          return 0;
27  }
28
29  void init(struct node_type **ThisList,int *len)
30  {
31          char str[MaxLen];
32          int  TempLen,TempMaxLen;
33          struct node_type *head,*next,*pre;
34          FILE *fin;
35
36          fin=freopen("data.txt","r",stdin);
37          TempMaxLen=0;
38          if(gets(str)!=NULL)
39          {
40                  TempLen=strlen(str);
41                  head=(struct node_type *)malloc(sizeof(struct node_type));
42                  head->word=(char *)malloc(TempLen*sizeof(char));
43                  if(TempLen>TempMaxLen)TempMaxLen=TempLen;
44                  strcpy(head->word,str);
45                  head->next=NULL;
46                  pre=head;
47          }
48          else
49          {
50                  *ThisList=NULL;
51                  fclose(fin);
52                  *len=0;
```

```
53          }
54
55          while(gets(str)!=NULL)
56          {
57                  TempLen=strlen(str);
58                  next=(struct node_type *)malloc(sizeof(struct node_type));
59                  next->word=(char *)malloc(TempLen*sizeof(char));
60                  if(TempLen>TempMaxLen)TempMaxLen=TempLen;
61                  strcpy(next->word,str);
62                  next->next=NULL;
63                  pre->next=next;
64                  pre=next;
65          }
66          fclose(fin);
67          *ThisList=head;
68          *len=TempMaxLen;
69  }
70
71  void print(struct node_type* ThisNode)
72  {
73          struct node_type* next;
74          next=ThisNode;
75          while(next!=NULL)
76          {
77                  printf("%s\n",next->word);
78                  next=next->next;
79          }
```

```
80  }
81
82  int eva(char *str,int i)
83  {
84          int k;
85          if(strlen(str)<i+1)k=0;
86          else if(str[i]==' ')k=27;
87                  else k=str[i]-'a'+1;
88          return k;
89  }
90
91  void bsort(struct node_type** ThisList,int r,int d)
92  {
93          struct node_type *p,*t,*pp,*tt,*head[MaxR],*tail[MaxR];
94          int i,j,k,k1,ks;
95          char *q;
96
97          p=*ThisList;
98          pp=NULL;
99          i=0;
100         while(i<d)
101         {
102                 for(j=0;j<r;j++)
103                 {
104                         head[j]=NULL;
105                         tail[j]=NULL;
106                 }
```

```
107             while(p!=NULL)
108             {
109                     q=p->word;
110                     if(i>0)
111                     {
112                             k1=eva(q,i-1);
113                             if(k1!=ks)
114                             {
115                                     ks=k1;
116                                     break;
117                             }
118                     }
119                     k=eva(q,i);
120                     if(head[k]==NULL)
121                     {
122                             head[k]=p;
123                             tail [k]=p;
124                     }
125                     else
126                     {
127                             tail [k]->next=p;
128                             tail [k]=p;
129                     }
130                     p=p->next;
131             }
132             for(j=0;j<r;j++)
133                     if(head[j]!=NULL)
```

```
134                         {
135                             if(pp==NULL)
136                             {
137                                 pp=head[j];
138                                 tt=tail[j];
139                             }
140                             else
141                             {
142                                 tt->next=head[j];
143                                 tt=tail[j];
144                             }
145                         }
146                 tt->next=NULL;
147                 if(p==NULL)
148                 {
149                     p=pp;
150                     t=tt;
151                     pp=NULL;
152                     ks=eva(p->word,i);
153                     i++;
154                 }
155         }
156         *ThisList=p;
157 }
```