Good

2020/3/18

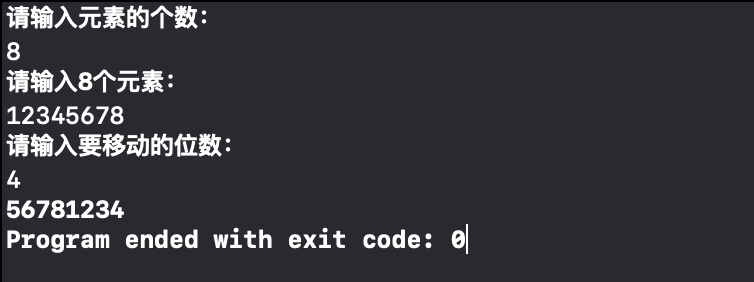
数据结构作业

姓名：叶先志 学号：18040400011 班级：1818039

第17题√

代码：

1. #include<stdio.h>
2. #define MAX 1024
3. **typedef** **char** datatype;
4. **typedef** **struct**
5. {
6. **int** last;
7. datatype data[MAX];
8. }sequenlist;
10. **void** moves(sequenlist \*L,**int** k)         //将q中元素移动k位
11. {
12. **int** i;
13. datatype temp;
14. **while**(k--)
15. {
16. temp=L->data[L->last];
17. **for**(i=L->last-1;i>=0;i--)
18. {
19. L->data[i+1]=L->data[i];
20. }
21. L->data[0]=temp;
22. }
23. }
25. **int** main()
26. {
27. sequenlist \*p,a;
28. **int** k,i,n;
29. p=&a;
30. printf("请输入元素的个数：\n");
31. scanf("%d",&n);
32. getchar();
33. p->last=n-1;
34. printf("请输入%d个元素：\n",n);
35. **for**(i=0;i<n;i++)
36. {
37. scanf("%c",&p->data[i]);
38. }
39. printf("请输入要移动的位数：\n");
40. scanf("%d",&k);
41. moves(p,k);
42. **for**(i=0;i<n;i++)
43. {
44. printf("%c",p->data[i]);
45. }
46. printf("\n");
47. **return** 0;
48. }

程序效果：

时间复杂度：O（k\*n）

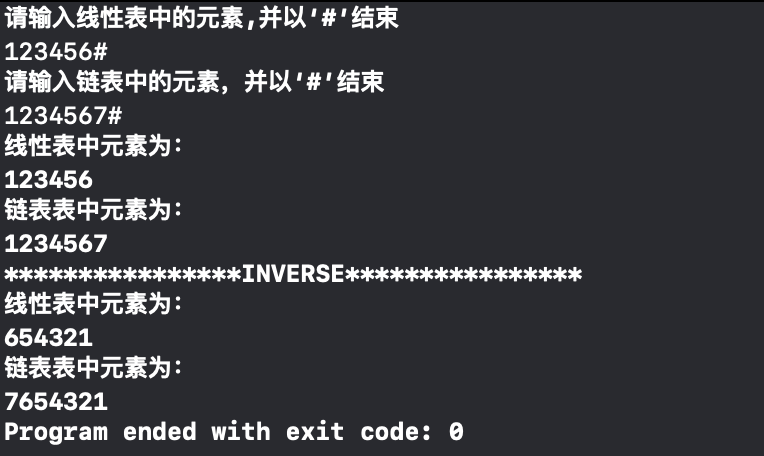
其中k为移动位数，n为元素个数

第21题√

代码：

1. #include <stdio.h>
2. #include <stdlib.h>
3. #define MAXSIZE 1024
4. **typedef** **char** datatype;
5. **typedef** **struct**
6. {
7. datatype data[MAXSIZE];
8. **int** last;
9. }sequenlist;
10. **typedef** **struct** node
11. {
12. datatype data;
13. **struct** node \*next;
14. }linklist;
16. sequenlist \*creatse()           //创建顺序表，last表示最后一个元素的i值
17. {
18. sequenlist \*L;
19. datatype ch;
20. L=(sequenlist\*)malloc(**sizeof**(sequenlist));
21. L->last=-1;
22. printf("请输入线性表中的元素,并以‘#’结束\n");
23. **for**(**int** i=0;i<MAXSIZE;i++)
24. {
25. scanf("%c",&ch);
26. **if**(ch=='#')
27. **break**;
28. L->data[i]=ch;
29. L->last++;
30. }
31. **if**(L->last==MAXSIZE-1)
32. printf("线性表已满\n");
33. **return** L;
34. }
36. linklist \*creatli()             //创建链表（有表头）
37. {
38. linklist \*L,\*s,\*p;
39. datatype ch;
40. L=(linklist\*)malloc(**sizeof**(linklist));
41. p=L;
42. printf("请输入链表中的元素，并以‘#’结束\n");
43. scanf("%c",&ch);
44. **while**(ch!='#')
45. {
46. s=(linklist\*)malloc(**sizeof**(linklist));
47. s->data=ch;
48. p->next=s;
49. p=s;
50. scanf("%c",&ch);
51. }
52. p->next=NULL;
53. **return** L;
54. }
56. **void** showse(sequenlist \*L)      //打印顺序表
57. {
58. sequenlist \*p;
59. p=L;
60. printf("线性表中元素为：\n");
61. **for**(**int** i=0;i<=L->last;i++)
62. {
63. printf("%c",L->data[i]);
64. }
65. printf("\n");
66. }
68. **void** showli(linklist \*L)        //打印链表
69. {
70. linklist \*p;
71. p=L->next;
72. printf("链表表中元素为：\n");
73. **while**(p!=NULL)
74. {
75. printf("%c",p->data);
76. p=p->next;
77. }
78. printf("\n");
79. }
81. **void** inversese(sequenlist \*L)       //逆转顺序表
82. {
83. datatype temp;
84. **int** n=(L->last-1)/2;
85. **for**(**int** i=0;i<=n;i++)
86. {
87. temp=L->data[i];
88. L->data[i]=L->data[L->last-i];
89. L->data[L->last-i]=temp;
90. }
91. }
93. **void** inverseli(linklist \*L)         //逆转链表
94. {
95. linklist \*p,\*q,\*r;
96. **if**(L->next&&L->next->next)
97. {
98. p=L->next;
99. q=p->next;
100. p->next=NULL;
101. r=p;
102. **while**(q)
103. {
104. p=q;
105. q=q->next;
106. p->next=r;
107. r=p;
108. }
109. L->next=r;
110. }
111. **else**
112. printf("链表元素个数小于2，无需逆转\n");
113. }
115. **int** main()
116. {
117. sequenlist \*se;
118. linklist \*li;
119. se=creatse();
120. getchar();
121. li=creatli();
122. showse(se);
123. showli(li);
124. inversese(se);
125. inverseli(li);
126. printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*INVERSE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");
127. showse(se);
128. showli(li);
129. **return** 0;
130. }

程序效果：



第27题√

代码：

1. #include <stdio.h>
2. #include <stdlib.h>
3. **typedef** **char** datatype;
4. **typedef** **struct** node
5. {
6. datatype data;
7. **struct** node \*next;
8. }linklist;
10. linklist \*creatli()             //创建链表
11. {
12. linklist \*L,\*p,\*s;
13. datatype ch;
14. L=(linklist\*)malloc(**sizeof**(linklist));
15. p=L;
16. printf("请输入链表中的元素，并以‘#’结束\n");
17. scanf("%c",&ch);
18. **while**(ch!='#')
19. {
20. s=(linklist\*)malloc(**sizeof**(linklist));
21. s->data=ch;
22. p->next=s;
23. p=s;
24. scanf("%c",&ch);
25. }
26. p->next=NULL;
27. **return** L;
28. }
30. **void** showli(linklist \*L)            //打印链表
31. {
32. linklist \*p;
33. p=L->next;
34. printf("链表中的内容为：\n");
35. **while**(p)
36. {
37. printf("%c",p->data);
38. p=p->next;
39. }
40. printf("\n");
41. }
43. linklist \*unionli(linklist \*A,linklist \*B)          //连接链表
44. {
45. linklist \*pa,\*pb,\*p,\*q,\*s;
46. pa=A->next;
47. pb=B->next;
48. q=NULL;
49. **while**(pa&&pb)
50. {
51. **if**(pa->data<=pb->data)
52. {
53. p=pa;
54. pa=p->next;
55. }
56. **else**
57. {
58. p=pb;
59. pb=p->next;
60. }
61. p->next=q;
62. q=p;
63. }
64. **if**(pa==NULL)
65. p=pb;
66. **else**
67. p=pa;
68. s=p->next;
69. **while**(s)
70. {
71. p->next=q;
72. q=p;
73. p=s;
74. s=p->next;
75. }
76. p->next=q;
77. A->next=p;
78. free(B);
79. **return** A;
80. }
82. **int** main()
83. {
84. linklist \*A,\*B,\*C;
85. printf("链表A的创建，");
86. A=creatli();
87. showli(A);
88. getchar();
89. printf("链表B的创建，");
90. B=creatli();
91. showli(B);
92. printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*连接链表\*\*\*\*\*\*\*\*\*\*\*\*\*\n");
93. C=unionli(A,B);
94. showli(C);
95. **return** 0;
96. }

程序效果：

图片包含 屏幕截图

描述已自动生成

第31题√

代码：

1. #include <stdio.h>
2. #include <stdlib.h>
3. **typedef** **char** datatype;
4. **typedef** **struct** node
5. {
6. datatype data;
7. **struct** node \*next;
8. }linklist;
10. linklist \*Creat()           //创建链表（带头节点）
11. {
12. **char** ch;
13. linklist \*head,\*s,\*r;
14. head=(linklist\*)malloc(**sizeof**(linklist));
15. r=head;
16. printf("请输入链表中的字符，并以‘#’结束：\n");
17. **while**(1)
18. {
19. scanf("%c",&ch);
20. **if**(ch=='#')
21. **break**;
22. s=(linklist\*)malloc(**sizeof**(linklist));
23. s->data=ch;
24. r->next=s;
25. r=s;
26. }
27. r->next=NULL;
28. **return** head;
29. }
31. linklist \*sortnum(linklist \* L)             //整理出数字并返回头节点
32. {
33. linklist \*numhead,\*p,\*r,\*p1;
34. numhead=(linklist\*)malloc(**sizeof**(linklist));
35. p1=numhead;
36. p=L->next;
37. r=L;
38. **while**(p!=NULL)
39. {
40. **if**(p->data>=48&&p->data<=57)
41. {
42. p1->next=p;
43. p1=p;
44. r->next=p->next;
45. p=r->next;
46. p1->next=NULL;
47. }
48. **else**
49. {
50. r=p;
51. p=r->next;
52. }
53. }
54. **return** numhead;
55. }
57. linklist \*sortmot(linklist \* L)             //整理出字母并返回头节点
58. {
59. linklist \*mothead,\*p,\*r,\*p1;
60. mothead=(linklist\*)malloc(**sizeof**(linklist));
61. p1=mothead;
62. p=L->next;
63. r=L;
64. **while**(p!=NULL)
65. {
66. **if**((p->data>=65&&p->data<=90)||(p->data>=97&&p->data<=122))
67. {
68. p1->next=p;
69. p1=p;
70. r->next=p->next;
71. p=r->next;
72. p1->next=NULL;
73. }
74. **else**
75. {
76. r=p;
77. p=r->next;
78. }
79. }
80. **return** mothead;
81. }
83. linklist \*sortsym(linklist \* L)             //整理出符号并返回头节点
84. {
85. linklist \*symhead,\*p,\*r,\*p1;
86. symhead=(linklist\*)malloc(**sizeof**(linklist));
87. p1=symhead;
88. p=L->next;
89. r=L;
90. **while**(p!=NULL)
91. {
92. **if**((p->data>=65&&p->data<=90)||(p->data>=97&&p->data<=122)||(p->data>=48&&p->data<=57))
93. {
94. r=p;
95. p=r->next;
96. }
97. **else**
98. {
99. p1->next=p;
100. p1=p;
101. r->next=p->next;
102. p=r->next;
103. p1->next=NULL;
104. }
105. }
106. **return** symhead;
107. }
109. **void** showlist(linklist \*L)          //打印链表内容
110. {
111. linklist \*p;
112. p=L->next;
113. **while**(p!=NULL)
114. {
115. printf("%c",p->data);
116. p=p->next;
117. }
118. printf("\n");
119. }
121. **int** main()
122. {
123. linklist \*L,\*numlist,\*motlist,\*symlist;
124. L=Creat();
125. printf("表中元素为：\n");
126. showlist(L);
128. numlist=sortnum(L);
129. printf("表中数字有：\n");
130. showlist(numlist);
132. motlist=sortmot(L);
133. printf("表中字母有：\n");
134. showlist(motlist);
136. symlist=sortsym(L);
137. printf("表中字符有：\n");
138. showlist(symlist);
140. **return** 0;
141. }

程序效果：

图片包含 屏幕截图

描述已自动生成