

# 温州大学计算机与人工智能学院

## 程序设计课程设计 实验报告

实验名称	动态链表：修改				
班 级	23大数据1	姓 名	徐王晶	学 号	23211870102
实验地点	南5B105	实验时间	2023-12-26,14:23:49	指导老师	虞铭财

### 一、问题编号：

2080

地址：<http://10.132.254.54/problem/2080/>

### 二、问题描述：

请设计一个简单的学生成绩管理系统，要求系统实现以下功能：  
学生信息包括学号、姓名、性别、语文、数学、英语。  
插入学生信息：  
Insert id name sex x y z 其中的参数分别为学号、姓名、性别、三门课的成绩，成绩为浮点数。  
输出所有学生信息：  
List  
按照插入的顺序输出所有学生的信息，每行一位学生的信息。每行的格式如下：  
id name sex x y z  
数据之间一个空格，成绩保留1位小数。  
查找学生信息：  
Find id  
查找学号为id的学生信息。  
修改学生信息：  
Change id newName, newSex, newX, newY, newZ  
把学号为id的学生信息修改为newName, newSex, newX, newY, newZ（学号保持不变）  
退出程序：  
Quit或者Exit

### 三、输入说明：

输入有多行，每行一条指令，指令格式如下：  
Insert id name sex x y z  
插入学生信息，分别为学号、姓名、性别和三门课的成绩。  
List  
输出所有学生信息。  
Find id  
查找学号为id的学生信息。  
Change id newName newSex newZ newY newZ  
把学号为id的学生信息修改为newName newSex newX newY newZ（学号保持不变）  
Quit或者Exit  
输出"Good bye!"后结束程序。

### 四、输出说明：

输出有多行，对应命令的输出如下：  
Insert id name sex x y z  
插入后在先在单独的一行中输出"Insert:"，然后在第二行中显示学生信息，数据之间用一个空格分开，成绩保留1位小数。  
List  
第一行输出"List:"，接下来按照插入的顺序输出所有学生的信息，每行一位学生的信息。每行的格式如下：  
id name sex x y z  
数据之间用一个空格分开，成绩保留1位小数。  
Find id  
第一行显示"Find:"，第二行显示格式如下：  
如果找到学号为id的学生，则在单独一行中显示学生信息，格式如List。否则在单独一行显示"Failed"。  
Change id newName newSex newX newY newZ  
第一行显示"Change:"。如果链表中不存在学号为id的学生，显示"Failed"。否则修改该学生信息并在单独一行中显示该生信息，显示格式如List命令。  
Quit或者Exit  
在单独一行中输出"Good bye!"后结束程序。

五、输入样例：

```
Insert 0911001 zhangsan F 87 78 65
Insert 0911002 zhaoliu F 97 90 55
Insert 0911003 Lisi F 77 72 55
Change 0911001 Zhangsan M 77 78 65
Change 0911004 Wangwu M 77 78 65
Insert 0911004 Wangwu F 68 56 95
Find 0911004
List
Quit
```

六、输出样例：

```
Insert:
0911001 zhangsan F 87.0 78.0 65.0
Insert:
0911002 zhaoliu F 97.0 90.0 55.0
Insert:
0911003 Lisi F 77.0 72.0 55.0
Change:
0911001 Zhangsan M 77.0 78.0 65.0
Change:
Failed
Insert:
0911004 Wangwu F 68.0 56.0 95.0
Find:
0911004 Wangwu F 68.0 56.0 95.0
List:
0911001 Zhangsan M 77.0 78.0 65.0
0911002 zhaoliu F 97.0 90.0 55.0
0911003 Lisi F 77.0 72.0 55.0
0911004 Wangwu F 68.0 56.0 95.0
Good bye!
```

七、解答内容：

所用语言：

源代码：

```
001. #include <stdio.h>
002. #include <string.h>
003. #include <stdlib.h>
004.
005. struct Data
006. {
007.     char id[20];
008.     char name[20];
009.     char sex;
010.     double x;
011.     double y;
012.     double z;
013.     struct Data *next;
014. };
015.
016. struct LinkList
017. {
018.     struct Data *head;
019. };
020.
021. void Insert(struct LinkList *llst);
022. void List(struct LinkList *llst);
023. void Find(struct LinkList *llst);
024. void Change(struct LinkList *llst);
025.
026. int main(void)
027. {
028.     //freopen("./in1.txt", "r", stdin);
029.
030.     char op[20];
031.     struct LinkList llst;
032.     llst.head = NULL;
033.
034.     while (scanf("%s", op), strcmp(op, "Quit") != 0)
035.     {
036.         if (strcmp(op, "Insert") == 0)
037.         {
038.             printf("Insert:\n");
039.             Insert(&llst);
```

```

040.     }
041.     else if (strcmp(op, "List") == 0)
042.     {
043.         printf("List:\n");
044.         List(&llst);
045.     }
046.     else if (strcmp(op, "Find") == 0)
047.     {
048.         printf("Find:\n");
049.         Find(&llst);
050.     }
051.     else if (strcmp(op, "Change") == 0)
052.     {
053.         printf("Change:\n");
054.         Change(&llst);
055.     }
056.     else if (strcmp(op, "Delete") == 0)
057.     {
058.         printf("Delete:\n");
059.     }
060. }
061. printf("Good bye!\n");
062.
063. return 0;
064. }
065.
066. void Insert(struct LinkList *llst)
067. {
068.     struct Data * node = (struct Data *)malloc(sizeof(struct Data));
069.     scanf(
070.         "%s %s %c %lf %lf %lf",
071.         node -> id,
072.         node->name,
073.         &(node -> sex),
074.         &(node -> x),
075.         &(node -> y),
076.         &(node -> z)
077.     );
078.     node -> next = NULL;
079.
080.     struct Data * p = llst -> head;
081.     struct Data * q = p;
082.     if (p == NULL)
083.     {
084.         llst -> head = node;
085.     }
086.     else
087.     {
088.         while (p != NULL)
089.         {
090.             q = p;
091.             p = p -> next;
092.         }
093.         q -> next = node;
094.     }
095.
096.     printf(
097.         "%s %s %c %.11f %.11f %.11f\n",
098.         node -> id,
099.         node -> name,
100.         node -> sex,
101.         node -> x,
102.         node -> y,
103.         node -> z
104.     );
105. }
106.
107. void List(struct LinkList *llst)
108. {
109.     struct Data * p = llst -> head;
110.     while (p != NULL)
111.     {
112.         printf(
113.             "%s %s %c %.11f %.11f %.11f\n",
114.             p -> id,
115.             p -> name,
116.             p -> sex,
117.             p -> x,
118.             p -> y,
119.             p -> z
120.         );
121.         p = p -> next;
122.     }
123. }
124.
125. void Find(struct LinkList *llst)
126. {
127.     char id[20];
128.     scanf("%s", id);
129.
130.     struct Data * p = llst -> head;

```

```

131.     while (p != NULL && strcmp(p -> id, id) != 0)
132.     {
133.         p = p -> next;
134.     }
135.
136.     if (p != NULL)
137.     {
138.         printf(
139.             "%s %s %c %.11f %.11f %.11f\n",
140.             p -> id,
141.             p -> name,
142.             p -> sex,
143.             p -> x,
144.             p -> y,
145.             p -> z
146.         );
147.     }
148.     else
149.     {
150.         printf("Failed\n");
151.     }
152. }
153.
154. void Change(struct LinkList *llst)
155. {
156.     char id[20];
157.     scanf("%s", id);
158.
159.     struct Data * p = llst -> head;
160.     while (p != NULL && strcmp(p -> id, id) != 0)
161.     {
162.         p = p -> next;
163.     }
164.
165.     if (p != NULL)
166.     {
167.         scanf(
168.             "%s %c %lf %lf %lf",
169.             p -> name,
170.             &(p -> sex),
171.             &(p -> x),
172.             &(p -> y),
173.             &(p -> z)
174.         );
175.
176.         printf(
177.             "%s %s %c %.11f %.11f %.11f\n",
178.             p -> id,
179.             p -> name,
180.             p -> sex,
181.             p -> x,
182.             p -> y,
183.             p -> z
184.         );
185.     }
186.     else
187.     {
188.         printf("Failed\n");
189.     }
190. }

```

## 八、判题结果

**AC - 答案正确**

判题结果补充说明：

test id:4021,result:AC, usedtime:0MS, usedmem:864KB,score:100