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纤细，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、输入样列：

```
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、输出样列：

```
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：

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

struct Data
{
    char id[20];
    char name[20];
    char sex;
    double x;
    double y;
    double z;
    struct Data *next;
};

struct LinkList
{
    struct Data *head;
};

void Insert(struct LinkList *llst);
void List(struct LinkList *llst);
void Find(struct LinkList *llst);
void Change(struct LinkList *llst);

int main(void)
{
    //freopen("./in1.txt", "r", stdin);

    char op[20];
    struct LinkList llst;
    llst.head = NULL;

    while (scanf("%s", op), strcmp(op, "Quit") != 0)
    {
        if (strcmp(op, "Insert") == 0)
        {
            printf("Insert:\n");
            Insert(&llst);
```

```
        }
        else if (strcmp(op, "List") == 0)
        {
            printf("List:\n");
            List(&llst);
        }
        else if (strcmp(op, "Find") == 0)
        {
            printf("Find:\n");
            Find(&llst);
        }
        else if (strcmp(op, "Change") == 0)
        {
            printf("Change:\n");
            Change(&llst);
        }
        else if (strcmp(op, "Delete") == 0)
        {
            printf("Delete:\n");
        }
    }
    printf("Good bye!\n");

    return 0;
}

void Insert(struct LinkList *llst)
{
    struct Data * node = (struct Data *)malloc(sizeof(struct Data));
    scanf(
        "%s %s %c %lf %lf %lf",
        node -> id,
        node->name,
        &(node -> sex),
        &(node -> x),
        &(node -> y),
        &(node -> z)
    );
    node -> next = NULL;

    struct Data * p = llst -> head;
    struct Data * q = p;
    if (p == NULL)
    {
        llst -> head = node;
    }
    else
    {
        while (p != NULL)
        {
            q = p;
            p = p -> next;
        }
        q -> next = node;
    }

    printf(
        "%s %s %c %.1lf %.1lf %.1lf\n",
        node -> id,
        node -> name,
        node -> sex,
        node -> x,
        node -> y,
        node -> z
    );
}

void List(struct LinkList *llst)
{
    struct Data * p = llst -> head;
    while (p != NULL)
    {
        printf(
            "%s %s %c %.1lf %.1lf %.1lf\n",
            p -> id,
            p -> name,
            p -> sex,
            p -> x,
            p -> y,
            p -> z
        );
        p = p -> next;
    }
}

void Find(struct LinkList *llst)
{
    char id[20];
    scanf("%s", id);

    struct Data * p = llst -> head;
```

```
    while (p != NULL && strcmp(p -> id, id) != 0)
    {
        p = p -> next;
    }

    if (p != NULL)
    {
        printf(
            "%s %s %c %.1lf %.1lf %.1lf\n",
            p -> id,
            p -> name,
            p -> sex,
            p -> x,
            p -> y,
            p -> z
        );
    }
    else
    {
        printf("Failed\n");
    }
}

void Change(struct LinkList *llst)
{
    char id[20];
    scanf("%s", id);

    struct Data * p = llst -> head;
    while (p != NULL && strcmp(p -> id, id) != 0)
    {
        p = p -> next;
    }

    if (p != NULL)
    {
        scanf(
            "%s %c %lf %lf %lf",
            p -> name,
            &(p -> sex),
            &(p -> x),
            &(p -> y),
            &(p -> z)
        );

        printf(
            "%s %s %c %.1lf %.1lf %.1lf\n",
            p -> id,
            p -> name,
            p -> sex,
            p -> x,
            p -> y,
            p -> z
        );
    }
    else
    {
        printf("Failed\n");
    }
}
```

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