#include <stdio.h>

#include <stdlib.h>

#include <string.h>

// 定义学生成绩信息结构体

typedef struct Student {

char name[50];

int score;

struct Student \*next;

} Student;

// 初始化线性表（单链表）

void InitList(Student \*\*head) {

\*head = NULL; // 将头指针置为空，表示链表为空

}

// 插入学生信息

void InsertStudent(Student \*\*head, char name[], int score) {

Student \*newNode = (Student \*)malloc(sizeof(Student)); // 分配新节点内存

if (!newNode) {

perror("Memory allocation failed"); // 内存分配失败，输出错误信息

exit(EXIT\_FAILURE); // 终止程序

}

strcpy(newNode->name, name); // 复制学生姓名到新节点

newNode->score = score; // 设置新节点的分数

newNode->next = \*head; // 将新节点插入到链表头部

\*head = newNode; // 更新头指针

}

// 遍历学生信息

void TraverseList(Student \*head) {

Student \*current = head; // 从头节点开始遍历

while (current) {

printf("Name: %s, Score: %d\n", current->name, current->score); // 打印学生信息

current = current->next; // 移动到下一个节点

}

}

// 删除学生信息（按姓名删除）

void DeleteStudent(Student \*\*head, char name[]) {

Student \*current = \*head;

Student \*prev = NULL;

while (current) {

if (strcmp(current->name, name) == 0) { // 如果找到匹配的学生

if (prev) {

prev->next = current->next; // 如果不是头节点，更新前一个节点的next指针

} else {

\*head = current->next; // 如果是头节点，更新头指针

}

free(current); // 释放当前节点内存

return; // 删除成功，返回

}

prev = current; // 更新前一个节点

current = current->next; // 移动到下一个节点

}

printf("Student not found.\n"); // 如果没有找到匹配的学生，输出提示信息

}

// 修改学生信息（按姓名修改）

void ModifyStudent(Student \*head, char name[], int newScore) {

Student \*current = head;

while (current) {

if (strcmp(current->name, name) == 0) { // 如果找到匹配的学生

current->score = newScore; // 更新学生分数

return; // 修改成功，返回

}

current = current->next; // 移动到下一个节点

}

printf("Student not found.\n"); // 如果没有找到匹配的学生，输出提示信息

}

// 查询学生信息（按姓名查询）

void QueryStudent(Student \*head, char name[]) {

Student \*current = head;

while (current) {

if (strcmp(current->name, name) == 0) { // 如果找到匹配的学生

printf("Name: %s, Score: %d\n", current->name, current->score); // 打印学生信息

return; // 查询成功，返回

}

current = current->next; // 移动到下一个节点

}

printf("Student not found.\n"); // 如果没有找到匹配的学生，输出提示信息

}

// 菜单函数（此处只展示了部分代码，未完整实现）

void Menu() {

// ... 省略了部分代码

}

// 主函数（此处未给出，但通常Menu函数会在主函数中调用）

// int main() {

// // ... 省略了部分代码

// }