#include<stdio.h>

#include<stdlib.h>

#include <string.h>

struct student{

int id;

char name[50];

int age;

float gpa;

};

struct node{

struct student data;

struct node \*next;

};

struct node \*head =NULL;

//the seven function

void addstudent(const struct student \*const ptr);//pass the struct by adress in constant pointer

void displayStudents(void);

void searchStudentByID(int id);

void UpdateStudent(int id);

float calculateAverageGPA(void);

void searchHighestGPA(void);

void deleteStudent(int id);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

int main(void){

setvbuf(stdout, NULL, \_IONBF,0);

setvbuf(stderr, NULL, \_IONBF,0);

struct student s1;

int id;

float average;

int choise;

while(1){

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("1:add new student \n2:display \n3:search student by id \n4:update Student \n5:calculateAverageGPA \n6:search highest GPA \n7:Delete Student \n8:exit program \n");

printf("choise: ");

scanf("%d",&choise);

printf("-----------------------------------------------------------------------------------\n");

switch(choise){

case 1:

printf("Enter student ID: ");

scanf("%d", &s1.id);

printf("Enter student name: ");

scanf("%s", s1.name);

printf("Enter student age: ");

scanf("%d",&s1.age);

printf("Enter student gpa: ");

scanf("%f",&s1.gpa);

addstudent(&s1);

break;

case 2:

displayStudents();

break;

case 3:

printf("Enter student ID: ");

scanf("%d",&id);

searchStudentByID(id);

break;

case 4:

printf("Enter student ID that need to updated : ");

scanf("%d",&id);

UpdateStudent(id);

break;

case 5:

average=calculateAverageGPA();

if(average){

printf("the average GPA of the Students is = %f",average);

}

break;

case 6:

searchHighestGPA();

break;

case 7:

printf("Enter student ID that need to deleted : ");

scanf("%d",&id);

deleteStudent(id);

break;

case 8:

break;

}

if(choise==8){

printf("exit program...");

break;

}

}

return 0;

}

//function to add student

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void addstudent(const struct student \*const ptr)//pass the struct by adress in constant pointer

{

struct node\*link=(struct node\*)malloc(sizeof(struct node));

link->next=NULL;

link->data.id =ptr->id;

strcpy(link->data.name, ptr->name);

link->data.age = ptr->age;

link->data.gpa = ptr->gpa;

struct node\*delete\_node=NULL;

struct node\*current=NULL;

if(head==NULL){

head=link;

return;

}

current=head;

while(current!=NULL){

if(current->data.id==ptr->id){

delete\_node=link;

printf("----------------------\n");

printf("this id is used before\n");

free(delete\_node);

return;

}

current=current->next;

}

current=head;

while(current->next!=NULL){

current=current->next;

}

current->next=link;

}

//function displayyy

//----------------------------------------------------------------

void displayStudents(void){

struct node\*ptr=NULL;

ptr=head;

if (head==NULL){

printf("the list is empty please fill the student information \n");

return;

}

while(ptr!=NULL){

printf("the id of student : %d \t",ptr->data.id );

printf("the name of student : %s \t",ptr->data.name );

printf("the age of student : %d \t" ,ptr->data.age );

printf("the gpa of student : %0.2f\n",ptr->data.gpa );

ptr=ptr->next;

}

}

//function search by ID

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void searchStudentByID(int id){

struct node\*ptr=NULL;

ptr=head;

if (head==NULL){

printf("the list is empty please fill the student information \n");

return;

}

while(ptr!=NULL){

if(ptr->data.id==id){

printf("the id of student : %d \t",ptr->data.id );

printf("the name of student : %s \t",ptr->data.name );

printf("the age of student : %d \t" ,ptr->data.age );

printf("the gpa of student : %0.2f\n",ptr->data.gpa );

return;

}

ptr=ptr->next;

}

if(ptr==NULL){

printf("this id not found \n");

}

}

//function to uptade item

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void UpdateStudent(int id){

struct node\*ptr=NULL;

ptr=head;

if (head==NULL){

printf("the list is empty please fill the student information \n");

return;

}

while(ptr!=NULL){

if(ptr->data.id==id){

printf("enter the new name :\n");

scanf("%s",ptr->data.name);

printf("enter the new age :\n");

scanf("%d",&ptr->data.age);

printf("enter the new gpa :\n");

scanf("%f",&ptr->data.gpa);

return;

}

ptr=ptr->next;

}

if(ptr==NULL){

printf("this id not found \n");

return;

}

}

//function to get the highest gpa

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void searchHighestGPA(void){

struct node\*ptr=NULL;

float max=0;

struct node\*current=NULL;

current=head;

ptr=head;

if (head==NULL){

printf("the list is empty please fill the student information \n");

return;

}

while(ptr!=NULL){

if(ptr->data.gpa>=max){

max=ptr->data.gpa;

}

ptr=ptr->next;

}

//after this while loop max contain the max gpa

while(current!=NULL){

if(current->data.gpa==max){

printf("the id of student : %d \t",current->data.id );

printf("the name of student : %s \t",current->data.name );

printf("the age of student : %d \t" ,current->data.age );

printf("the gpa of student : %0.2f\n",current->data.gpa );

return;

}

current=current->next;

}

}

//function to calculate average gpa

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

float calculateAverageGPA(void){

struct node\*current=NULL;

float sum=0;

int count=0;

float average;

if (head==NULL){

printf("the list is empty please fill the student information \n");

return 0;

}

current=head;

while(current!=NULL){

sum+=current->data.gpa;//make sum equal the gpa of all list

count++;

current=current->next;

}

average=sum/count;

return average;

}

//function to delete student by id

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void deleteStudent(int id){

struct node\* current,\*prev,\*delete\_node=NULL;

if (head==NULL){

printf("the list is empty please fill the student information \n");

return;

}

if (head->data.id==id){

delete\_node=head;

head=head->next;

free(delete\_node);

printf("\n Done");

return;

}

current=head->next;

prev=head;

while(current!=NULL){

if(current->data.id==id){

prev->next=current->next;

delete\_node=current;

free(delete\_node);

printf("\n Done");

return;

}

prev=current;

current=current->next;

}

if(current==NULL){

printf("this id doesn't exist");

}

}