20211155, korennova iuliia, 미디어&엔터텐먼트전공

보고서 프로젝트 (C)

프로젝트에서는 가능하면 구조물을 만들고 조합을 사용할 필요가 있었다. 또한 데이터가 들어 있는 파일을 만들고, 이 데이터를 구조물에 입력하며, 출력 창과 새 텍스트 파일을 동시에 출력해야 했다. 또한 사용자가 사용할 수 있는 메뉴를 추가해야 했습니다. 나는 내가 할 수 있는 한 요건을 충족시키려고 노력했다.

“inputfile" 파일을 위해 다음 정보를 업로드했다:

Stepan 2009000 20 1 yes no adrIstra h1 543 880012345 3.0

Lisa 2009001 30 2 yes no adrMoskva h2 33 890012345 4.0

Rita 2009002 40 3 no yes adrPeter h3 345 849512345 5.0

신분증, 나이, 현재 대학에서 공부하고 있는 학생이라면 장학금을 몇 번 받았는지, 지금 교환학생을 하고 있는 학생이라면 주소, cgpa 등이 적혀 있다.

코드가 다음과 같다:

거기서 두 개의 '구조물'을 만들어 하나로 묶었는데, 사실 많이 필요하지 않다고 생각한다. 아마도 구조를 수정할 가치가 있을 것입니다. 제가 알기로는, 사용자와 학생은 비슷한 (공통) 키를 가지고 있습니다. (이름과 전화 번호 - 공통 엔티티로 분리하는 것이 좋습니다. 활자 구조{ char type, student 또는 professor 범주; 문자 이름, name - type 독립. long int phone, phone type 독립; long int는 2 147 483 647로 제한되기 때문에 long int -- 그리고 전화번호는 -- 8800003 3222 - {9자 이상}일 수 있습니다.}

1. 몇 가지 기능을 사용하기 위해 이 라이브러리를 추가했다.

#define CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include<ctype.h>

1. 두 가지 "구조"가 있습니다. 하나는 교수님을 위한 것이고 다른 하나는 학생들을 위한 것이다. 각 구조(학생 및 교수)에는 포함된 것과 동일한 변수가 포함된다. 문자열에는 약간의 공간이 추가된다.

typedef struct

{

char name[20];

char mail[30];

long int mobile;

} PROFESSOR;

typedef struct

{

char name[20];

char address[50];

char schoolstat[10];

int scholarship;

char abroad[10];

long int phone;

int age;

int ID;

float cgpa;

} STUDENT;

1. 그러면 이 두 구조 모두 "union" PERSON에 의해 결합된다.

typedef struct

{

char type;

union

{

PROFESSOR prof;

STUDENT stu;

} u;

} PERSON;

int main()

{

1. 최대 학생 수이다. 필요하다면 더 만들 수도 있지만, 지금은 많이 필요하지 않다. int MAX\_NUMBER\_OF\_STUDENTS = 6;
2. 파일 및 유니언에 대한 포인터를 정의합니다.

FILE \*infile, \*ofp;

PERSON data[MAX\_NUMBER\_OF\_STUDENTS];

1. 파일 읽기. 만약 그것이 읽기에 오류가 있다면, 그것은 그것을 알려줄 것이다.

infile = fopen("inputfile.txt", "r");

if (infile == NULL)

{

fprintf(stderr, "\nError opening file\n");

exit(1);

}

1. 조정 및 쓰기를 위해 파일을 엽니다.

ofp = fopen("outputfile.txt", "w");

if (ofp == NULL)

{

printf("\nOutput file open error!\n");

return 1;

}

1. 교수 초기화한다. 나는 데이터를 입력하는 것과 파일에서 데이터를 꺼내는 것을 합치기로 했다. 그것이 어떻게 작동할지 아는 것은 흥미로워 보였다.

data[0] = (PERSON){

.type = 'p',

.u.prof.name = "Sam",

.u.prof.mail = "ssss@mail.com",

.u.prof.mobile = 88003334545};

1. 학생 재설정이다. 앞으로 "빈" 학생이 표시되지 않도록 'n'으로 설정했다.

for (int i = 1; i < MAX\_NUMBER\_OF\_STUDENTS; i++)

{

data[i] = (PERSON){.type = 'n', .u.stu = (STUDENT){0}, .u.prof = (PROFESSOR){0}};

}

1. 나이 및 cgpa 카운팅 변수를 초기화한다.

int i = 1;

int total = 0;

int total\_age = 0;

1. Input 파일에서 데이터를 읽는다. 파일이 끝날 때까지 또는 구조의 한계를 넘을 때까지 실행한다.

while (fscanf(

infile,

"%s %d %d %d %s %s %15[^\n] %ld %f\n",

data[i].u.stu.name,

&data[i].u.stu.ID,

&data[i].u.stu.age,

&data[i].u.stu.scholarship,

data[i].u.stu.schoolstat,

data[i].u.stu.abroad,

data[i].u.stu.address,

&data[i].u.stu.phone,

&data[i].u.stu.cgpa) != EOF &&

i != MAX\_NUMBER\_OF\_STUDENTS)

{

1. 문자열을 읽고 사용자 범주 Student를 추가한다. 모든 PERSON을 표시할 때 추가 줄을 표시하지 않기 위해 필요하다.

data[i].type = 's';

1. [DEBUG] 코멘트를 해제하여 콘솔로 출력되는 내용을 확인할 수 있다.

// printf("\n[DEBUG] name = %s", data[i].u.stu.name);

total += data[i].u.stu.cgpa;

total\_age += data[i].u.stu.age;

// printf("\n[DEBUG] total = %d", total);

// printf("\n[DEBUG] total age = %d\n", total\_age);

1. 데이터를 파일에 저장한다.

fprintf(ofp, "%d %s %d %d %s %s %s %ld %f\n",

data[i].u.stu.ID,

data[i].u.stu.name,

data[i].u.stu.age,

data[i].u.stu.scholarship,

data[i].u.stu.schoolstat,

data[i].u.stu.abroad,

data[i].u.stu.address,

data[i].u.stu.phone,

data[i].u.stu.cgpa);

i++;

}

1. 콘솔에 모든 PERSON 데이터의 값 출력한다.

printf("Student: \tName\tAge\tID\t Scholarship InSchool Abroad Address\t\tPhone\tCGPA");

for (int i = 0; i < MAX\_NUMBER\_OF\_STUDENTS; i++)

{

if (data[i].type == 'p')

{

printf("\n\nProfessor: %s,\n%s, %ld\n\n",

data[i].u.prof.name,

data[i].u.prof.mail,

data[i].u.prof.mobile);

}

else if (data[i].type == 's')

{

printf("Student: \t%s \t%d \t%d \t%d \t%s \t%s \t%s \t%ld \t%f\n\n",

data[i].u.stu.name,

data[i].u.stu.age,

data[i].u.stu.ID,

data[i].u.stu.scholarship,

data[i].u.stu.schoolstat,

data[i].u.stu.abroad,

data[i].u.stu.address,

data[i].u.stu.phone,

data[i].u.stu.cgpa);

}

else

{

// do nothing

}

}

1. 새로운 변수를 선언된다. 이것은 간단한 메뉴이며 이에 대한 지침이 콘솔에 출력된다. 이 메뉴에는 평균 연령 찾기, 파일에 있는 줄 수 계산, 총 cgpa 점수 계산 및 목록에서 이름별로 사람의 전화 번호 찾기 옵션이 있다. 사용자가 메뉴를 통해 얻을 수 있는 모든 결과도 파일에 출력한다.

int menu;

printf("menu:\n1. Count the lines.\n2. Total cgpa\n3. Average age\n4.Find a phone number by name from the list\n5. Exit\n");

printf("enter the menu number: ");

scanf("%d", &menu);

if(menu==1){

printf("\nNumber of lines: %d\n", i);

fprintf(ofp, "\nNumber of lines: %d\n", i);

}

else if(menu==2){

printf("Total value of cgpa: %d", total);

fprintf(ofp, "Total value of cgpa: %d", total);

}

else if(menu==3){

printf("Average age: %.2f", (float)total\_age/(i-1));

fprintf(ofp, "Average age: %.2f", (float)total\_age/(i-1));

}

else if(menu==4){

1. 여기서 저는 학생들의 구조체로부터 연령층을 위한 공간을 할당하려고 했지만, 효과가 없었습니다.

//int numStudents=MAX\_NUMBER\_OF\_STUDENTS;

//STUDENT\* students = malloc(numStudents \* sizeof \*students);

//for (int i= 0; i < numStudents; i++)

// {

// students[i].age=(char\*)malloc(sizeof(char\*));

//}

1. 입력한 이름으로 전화 번호를 검색한다.

char name[6];

printf("name? ");

scanf("%s", &name);

int index=-1;

for(i=0;i<6;i++){

if(strcmp(data[i].u.stu.name, name)==0){

index=i;

break;

}

}

if(index>=0){

printf("%s's phone number: %d", data[index].u.stu.name, data[index].u.stu.phone);

fprintf(ofp, "%s's phone number: %d", data[index].u.stu.name, data[index].u.stu.phone);

}

else{

printf("did not find.\n");

}

}

else if(menu==5){

exit(1);

}

1. 파일 닫는다.

fclose(infile);

fclose(ofp);

1. 결과를 바다서 프로그램을 나간다.

return 0;

}