Московский Авиационный Институт (национальный исследовательский университет) Факультет «Прикладная математика и информатика» Кафедра 806

Курсовой проект по теме:

Обработка последовательной файловой структуры на языке Си

Выполнила студент группы М80-101Б-22

Соколова Виктория

Москва, 2023

Цель

Разработать последовательную структуру данных для представления простейшей базы данных на файлах в СП Си.

Идея

Реализовать в программе ввод с файла, содержащего необходимые данные, создать функцию для обхода таблицы и вывода значений, соответствующих заданному параметру p.

Содержимое и структура файла

Информация об успеваемости студентов данной группы по всем предметам: фамилия, инициалы, пол, номер группы, отметки по экзаменам и зачетам.

Задание

Выяснить, в какой группе учится максимальное число студенток с максимальным на курсе средним баллом. 21.

Входной файл test.txt

//фамилия инициалы пол группа баллы по предметам

Ivanov I.I. M 10 90 90 90 90

Ivanova E.I. F 10 100 100 100 99

Sidorov S.E. M 10 78 66 99 99

Petrova A.S F 10 97 97 97 100

Mashina D.D. F 56 78 75 49 49

Durov E.F. M 789 100 89 93 93

Sashina A.R. F 123 56 56 56 56

Petrova S.A F 96 67 97 97 97

Perova A.A F 96 67 97 97 97

Perkova I.A F 11 66 98 98 96

Тесты

PS C:\code\test\kp\kp6> ./a

Enter the file name: test.txt

=====================================================================

Surname |N.P |Gender |# gr. |Math |Russian|Phisics|Inf |Sum|

=====================================================================

Ivanov |I.I. |M |10 |90 |90 |90 |90 |360|

Ivanova |E.I. |F |10 |100 |100 |100 |99 |399|

Sidorov |S.E. |M |10 |78 |66 |99 |99 |342|

Petrova |A.S |F |10 |97 |97 |97 |100 |391|

Mashina |D.D. |F |56 |78 |75 |49 |49 |251|

Durov |E.F. |M |789 |100 |89 |93 |93 |375|

Sashina |A.R. |F |123 |56 |56 |56 |56 |224|

Petrova |S.A |F |96 |67 |97 |97 |97 |358|

Perova |A.A |F |96 |67 |97 |97 |97 |358|

Perkova |I.A |F |11 |66 |98 |98 |96 |358|

=====================================================================

Nomen with max middle mark:

Ivanova E.I.

Group: 10

People who have passed the selection:

Ivanova E.I. F 10 100 100 100 99

PS C:\code\test\kp\kp6> ./a

Enter the file name: fffff

Error, can't open file

Код

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#define NAME\_SIZE 30

#define INITIALS\_SIZE 4

int length = 0;

int cond = 0;

int help = 0;

typedef struct \_student {

    char Name[NAME\_SIZE];

    char Initials[NAME\_SIZE];

    char gender;

    int group;

    int math;

    int rus;

    int phis;

    int info;

    int sum;

}Student;

typedef struct \_list {

    Student person;

    struct \_list \*next;

    struct \_list \*prev;

}List;

typedef struct st

{

    int val;

    struct st\* next;

}st;

st \*push(st\* HEAD, int a)

{

    st\* TMP;

    if ((TMP = malloc(sizeof(st))) == NULL)

    {

        printf("No memory");

        return NULL;

    }

    TMP -> val = a;

    TMP -> next = HEAD;

    return TMP;

}

int reader(FILE \*in, Student \*person)

{

    fscanf(in, "%s", person->Name);

    fscanf(in, "%s", person->Initials);

    fscanf(in, "%s", &person->gender);

    fscanf(in, "%d", &person->group);

    fscanf(in, "%d", &person->math);

    fscanf(in, "%d", &person->rus);

    fscanf(in, "%d", &person->phis);

    fscanf(in, "%d", &person->info);

}

List \*create(List \*stips, Student \*person)

    {

    List \*temp = malloc(sizeof(List));

    temp->next = NULL;

    temp->prev = NULL;

    temp->person = \*person;

    length++;

    return temp;

}

List \*add(List \*stips, Student \*person)

{

    List \*temp = malloc(sizeof(List));

    temp->next = NULL;

    temp->prev = stips;

    temp->person = \*person;

    stips->next = temp;

    stips = temp;

    length++;

    return stips;

}

void printer(List \*element)

{

    Student man;

    if (length != 0) {

    while (element -> prev != NULL) {

        element = element -> prev;

    }

    man = element->person;

    printf("%s ", man.Name);

    printf("%s\n", man.Initials);

    while (element -> next != NULL) {

        element = element -> next;

        man = element->person;

        printf("%s ", man.Name);

        printf("%s\n", man.Initials);

        }

    }

}

int pop(struct st\*\* HEAD)

{

    struct st\* TMP;

    int help\_var;

    if (\*HEAD == NULL) {

        return '\0';

    }

    TMP = \*HEAD;

    help\_var = TMP -> val;

    \*HEAD = TMP -> next;

    free(TMP);

    return help\_var;

}

void print\_st(struct st\* st) {

    help = st -> val;

    printf("%d ", help);

    if (st -> next != NULL) {

        print\_st(st -> next);

    }

}

int main(int argc, char \*argv[]) {

    int group = 0, q = 0;

    char l[40];

    printf("Enter the file name: ");

    scanf("%s", &l);

    FILE \*in = fopen(l, "r");

    FILE \*out = fopen("Outdata", "wb");

    if((!in)){

        printf("Error, can't open file\n");

        return 0;

    }

    Student person;

    List \*stips = NULL, \*result = NULL;

    st \*numbers = NULL; // memorize F with the maximum score

    int c; // for trash

    printf("=====================================================================\n");

    printf("Surname\t|N.P\t|Gender\t|# gr.\t|Math\t|Russian|Phisics|Inf\t|Sum|\n");

    printf("=====================================================================\n");

    int sum = 0, sum\_max = 0;

    int step = 0;

    while (feof(in) == 0) {

        if (person.Name == NULL){

            break;

        }

        reader(in, &person);

        sum = person.math + person.rus + person.phis + person.info;

        person.sum = sum;

        if (person.gender == 'F') {

            if (sum == sum\_max) {

                numbers = push(numbers, step + 1);

            }

            if (sum > sum\_max) {

                while (numbers != NULL) {

                    c = pop(&numbers);

                }

                sum\_max = sum;

                numbers = push(numbers, step+1);

            }

        }

        printf( "%s\t|", person.Name);

        printf( "%s\t|", person.Initials);

        printf( "%s\t|", &person.gender);

        printf( "%d\t|", person.group);

        printf( "%d\t|", person.math);

        printf( "%d\t|", person.rus);

        printf( "%d\t|", person.phis);

        printf( "%d\t|", person.info);

        printf( "%d|\n", sum);

        ++step;

        if (stips == NULL) {

            stips = create(stips, &person);

        } else {

            stips = add(stips, &person);

        }

    }

    printf("=====================================================================\n");

    if (numbers == NULL) {

        printf("No female student(");

        return 0;

    }

    int step\_result = 0;

    c = pop(&numbers);

    if (c == step) {

        stips = stips->prev;

        person = stips->person;

        --step;

    } else {

        numbers = push(numbers, c);

    }

    int Array[step];

    int counter = step;

    while (counter > 0) {

        if (numbers == NULL) {

            break;

        }

        c = pop(&numbers);

        while (counter != c) {

            stips = stips->prev;

            person = stips->person;

            --counter;

        }

        if (result == NULL)

            result = create(result, &person);

        else

            result = add(result, &person);

        Array[step\_result] = person.group;

        ++step\_result;

    }

    printf("\nNomen with max middle mark:\n");

    printer(result);

    printf("\n");

    int BiggestCounter = 0, CurrentCounter = 0, FrequentEl;

    for (int i = 0; i < step\_result; i++)

    {

        for(int j = i; j < step\_result; j++)

        {

            if(Array[i] == Array[j])

            {

                CurrentCounter++;

            }

        }

        if(CurrentCounter > BiggestCounter)

        {

            BiggestCounter = CurrentCounter;

            FrequentEl = Array[i];

        }

        CurrentCounter = 0;

    }

    printf("Group: %d\n", FrequentEl);

    person = result->person;

    counter = step\_result;

    printf("\nPeople who have passed the selection:\n");

    while (counter > 0) {

        if (person.group == FrequentEl) {

            printf("%s %s %s %d %d %d %d %d\n", person.Name, person.Initials, &person.gender, person.group, person.math, person.rus, person.phis, person.info);

            fprintf(out, "%s %s %s %d %d %d %d %d\n", person.Name, person.Initials, &person.gender, person.group, person.math, person.rus, person.phis, person.info);

        }

        if (counter == 1)

        {

        break;

        }

        result = result->prev;

        person = result->person;

        --counter;

    }

    fclose(in);

    fclose(out);

    free(stips);

    free(result);

    return 0;

}