Лабораторная работа № 6

Файловый ввод-вывод.

Задание 1.1

Постановка задачи:

Напишите программу, которая записывает массив одинаковых структур с данными о студентах в текстовый файл формате CSV в режиме "добавления" ("a"), а затем

читает построчно этот файл в другой массив структур в режиме "чтения" ("r").

struct Student {

unsigned int id;

char[50] name;

char[50] faculty;

float rating;

};

В структуре перечислены:

1. id - уникальный идентификатор,

2. name - фамилия и имя,

3. faculty - название факультета,

4. rating - текущий рейтинг студента.

Данные о студентах вводятся вами с клавиатуры. Используйте функции файлового ввода-вывода.

Математическая модель:

-

Список идентификаторов:

Имя	**Тип**	**Назначение**
n	int	Количество студентов на ввод и далее на вывод.
students	struct Student*	Массив структур на ввод.
file	FILE *	students.cvs для записи/чтения вводимых структур.
students2	struct Student*	Массив структур на чтение.

Код программы:

```c
#include <stdlib.h>
#include "students.h"


int
main()
{
    int n;
    printf("Enter the amount of students:");
    scanf("%d", &n);
    getchar();

    struct Student *students = allocate_array(n);

    fill_array(n, students);

    FILE *file = NULL;
    file = fopen("students.cvs", "a");

    write_array(file, n, students);

    fclose(file);
    free(students);


    puts("Students to read:");
    scanf("%d",&n);
    getchar();

    if(n <= 0)
        exit(0);


    file = fopen("students.cvs","r");

    if(!file){
        puts("Could not open file!");
        exit(0);
    }

    struct Student *students2 = read_file_to_array(file, n);

    fclose(file);

    puts("From file:");
    print_array(n, students2);

    free(students2);
    return 0;
}
```

Результат выполненной работы:

```
>make
gcc -Wall -g -shared -fpic -o build/libstudents.so src/students.c
gcc -Wall -g -lstudents -L ./build/ -o main1.elf src/main1.c
echo export LD_LIBRARY_PATH="D_LIBRARY_PATH:WD"
export LD_LIBRARY_PATH=D_LIBRARY_PATH:WD
gcc -Wall -g -lstudents -L ./build/ -o main2.elf src/main2.c
echo export LD_LIBRARY_PATH="D_LIBRARY_PATH:WD"
export LD_LIBRARY_PATH=D_LIBRARY_PATH:WD
>./main1.elf
Enter the amount of students:2

Enter student's № 0 info:
        id:1
        name:qwerty
        faculty:ytrewq
        rating:10

Enter student's № 1 info:
        id:2
        name:dvorak
        faculty:karovd
        rating:10
Students to read:
2
From file:
Student №0
        id:1
        name:nig
        faculty:ger
        rating:0.000000
Student №1
        id:2
        name:maxwel gaylord
        faculty:travel
        rating:10.000000
```

students.cvs:

```
1 1,qwerty,ytrewq,10.000000
2 2,dvorak,karovd,10.000000
```

Задание 1.2

Постановка задачи:

Напишите программу, которая записывает массив одинаковых структур с данными о студентах в двоичный (бинарный) файл в режиме "записи" (перезаписи) ("wb"), а затем читает этот файл в другой массив структур в режиме "чтения" ("rb").

struct Student {

unsigned int id;

char[50] name;

char[50] faculty;

float rating;

};

В структуре перечислены:

1. id - уникальный идентификатор,

2. name - фамилия и имя,

3. faculty - название факультета,

4. rating - текущий рейтинг студента.

2

Используйте текст лекции и документацию по ссылкам выше для чтения/записи в двоичном режиме.

Математическая модель:

-

Список идентификаторов:

Имя	**Тип**	**Назначение**
n	int	Количество студентов на ввод и далее на вывод.
students	struct Student*	Массив структур на ввод.
file	FILE *	students.bin для записи/чтения вводимых структур.
students2	struct Student*	Массив структур на чтение.

Код программы:

```c
#include <stdlib.h>
#include "students.h"


int
main()
{
    int n;
    printf("Enter the amount of students:");
    scanf("%d", &n);
    getchar();

    struct Student *students = allocate_array(n);

    fill_array(n, students);

    FILE *file = NULL;
    file = fopen("students.bin", "wb");

    bin_write_array(file, n, students);

    fclose(file);
    free(students);


    puts("Students to read:");
    scanf("%d",&n);
    getchar();

    if(n <= 0)
        exit(0);


    file = fopen("students.bin","rb");

    if(!file){
        puts("Could not open file!");
        exit(0);
    }

    struct Student *students2 = bin_read_file_to_array(file, n);

    fclose(file);

    puts("From file:");
    print_array(n, students2);

    free(students2);
    return 0;
}
```

Результат выполненной работы:

```
>./main2.elf
Enter the amount of students:2

Enter student's № 0 info:
        id:1
        name:qwerty
        faculty:ytrewq
        rating:10

Enter student's № 1 info:
        id:2
        name:dvorak
        faculty:karvod
        rating:10
Students to read:
2
From file:
Student №0
        id:1
        name:qwerty
        faculty:ytrewq
        rating:10.000000
Student №1
        id:2
        name:dvorak
        faculty:karvod
        rating:10.000000
```

students.h

```
#ifndef STUDENTS_H
#define STUDENTS_H

#include <stdio.h>

struct Student *allocate_array(int n);
void fill_array(int n, struct Student *students);
void print_array(int n,struct Student *students);

void write_array(FILE *file, int n, struct Student *students);
struct Student* read_file_to_array(FILE *file, int n);

void bin_write_array(FILE *file,int n, struct Student *students);
struct Student* bin_read_file_to_array(FILE *file, int n);

#endif
```

students.c

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

struct Student{
    unsigned id;
    char name[50];
    char faculty[50];
    float rating;
};

struct Student *
allocate_array(int n){
    struct Student *students = NULL;
    students = malloc(sizeof(struct Student)*n);

    if(!students)
        exit(1);

    return students;
}

void
fill_array(int n, struct Student *students)
{
    for(int i = 0; i < n; i++){
        printf("\nEnter student's № %d info:\n", i);

        printf("\tid:");
        scanf("%u", &students[i].id);
        getchar();

        printf("\tname:");
        fgets(students[i].name, sizeof(students[i].name), stdin);
        students[i].name[strcspn(students[i].name, "\n")] = 0;

        printf("\tfaculty:");
        fgets(students[i].faculty, sizeof(students[i].faculty), stdin);
        students[i].faculty[strcspn(students[i].faculty, "\n")] = 0;

        printf("\trating:");
        scanf("%f", &students[i].rating);
        getchar();
    }
}
```

```c
void
write_array(FILE *file, int n, struct Student *students)
{
    for(int i = 0; i < n; i++){
        fprintf(file, "%u,%s,%s,%f\n", students[i].id, students[i].name, students[i].faculty, students[i].rating);
    }
}

void
bin_write_array(FILE *file, int n, struct Student *students)
{
    fwrite(&students[0], sizeof(struct Student), n, file);
}

struct Student*
bin_read_file_to_array(FILE *file, int n)
{
    struct Student *students = allocate_array(n);
    fread(students, sizeof(struct Student), n, file);
    return students;
}
```

```c
struct Student *
read_file_to_array( FILE *file, int n)
{
    char *buf = NULL;
    buf = malloc(255);
    struct Student *students = allocate_array(n);

    if(!buf)
        exit(1);

    char *tmp;

    int i = 0;
    while(fgets(buf, sizeof(struct Student), file) != NULL && i != n){
        if ((strlen(buf)>0) && (buf[strlen (buf) - 1] == '\n'))
            buf[strlen (buf) - 1] = '\0';

        tmp = strtok(buf, ",");
        students[i].id = atoi(tmp);

        tmp = strtok(NULL, ",");
        strcpy(students[i].name, tmp);

        tmp = strtok(NULL, ",");
        strcpy(students[i].faculty, tmp);

        tmp = strtok(NULL, ",");
        students[i].rating = atof(tmp);
        i++;
    }
    free(buf);
    return students;
}

void
print_array(int n, struct Student *students)
{
    for(int i = 0; i < n; i++){
        printf("Student N%d\n\tid:%u\n\tname:%s\n\tfaculty:%s\n\trating:%f\n", i, students[i].id, students[i].name, students[i].faculty, students[i].rating);
    }
}
```

makefile:

```
BUILD_DIR=build
SRC_DIR=src
CC=gcc
CFALGS=-Wall -g

all: main1.elf main2.elf

main1.elf: libstudents.so
    $(CC) $(CFALGS) -lstudents -L ./$(BUILD_DIR)/ -o main1.elf $(SRC_DIR)/main1.c
    echo export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$PWD"

main2.elf: libstudents.so
    $(CC) $(CFALGS) -lstudents -L ./$(BUILD_DIR)/ -o main2.elf $(SRC_DIR)/main2.c
    echo export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$PWD"


libstudents.so:
    $(CC) $(CFALGS) -shared -fpic -o $(BUILD_DIR)/libstudents.so $(SRC_DIR)/students.c

clean:
    rm $(BUILD_DIR)/*o
    rm main*.elf
```