# Лабораторная работа № 6

«Программирование операций над структурами и бинарными файлами»

# Цель работы

Изучение способов описания структур данных на языке С. Исследование особенностей обработки бинарных файлов, хранящих структурные типы данных.

# Ход выполнения работы

# Задание

Описать структуру с именем PRICE, содержащую следующие поля:

– название товара;

– название магазина, в котором продается товар;

– стоимость товара в грн.

Написать программу, выполняющую следующие действия с помощью функций:

– ввод с клавиатуры данных в файл, состоящий из элементов типа PRICE; записи должны быть размещены в алфавитном порядке по названиям товаров;

– чтение данных из этого файла;

– вывод на экран информации о товаре, название которого введено с клавиатуры;

– если таких товаров нет, выдать на дисплей соответствующее сообщение.

# Описание алгоритма решения задания

# Определение входных и выходных данных

Входные и выходные данные зависят от выбранного в меню действия.

При вводе нового товара от пользователя требуется ввести Характеристику товара/ов, а именно: Наименование товара и Наименование организации (строки длиной до 1024 символов), Стоимость товара (целое беззнаковое).

При поиске товара от пользователя требуется ввести Наименование товара.

При печати содержимого файла программа выводит на экран характеристику каждого товара, хранящегося в файле.

# Структурная схема алгоритма

Структурная схема алгоритма продемонстрирована в Приложении Б на Рисунках Б.1, Б.2 и Б.3.

# Исходный код программы (ЯП С, ОС GNU/Linux Ubuntu, компилятор gcc)

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

#include <ctype.h>

///////////////////////////////////

// Блок препроцессорных директив //

///////////////////////////////////

#define DEFAULT\_FILE\_NAME ".database"

#define NAME\_SIZE 1024

#define MARKET\_SIZE 1024

#define CHAR\_SIZE 6

#define MENU\_CLEAR\_FILE 1

#define MENU\_ADD\_STRUCTS 2

#define MENU\_PRINT\_STRUCTS 3

#define MENU\_SEARCH\_STRUCT 4

#define MENU\_EXIT 5

//////////////////////////////////////

// Блок объявления типов и структур //

//////////////////////////////////////

struct PRICE {

char name [NAME\_SIZE \* CHAR\_SIZE];

char market[MARKET\_SIZE \* CHAR\_SIZE];

unsigned long int price;

};

////////////////////////////////

// Блок объявления прототипов //

////////////////////////////////

unsigned long int fsize (char \*name);

void file\_clear (char \*name);

void file\_PrintStructs (char \*name);

void file\_AddStructs (char \*name);

void file\_SortStructs (char \*name);

int struct\_GetFromFile(FILE \*f, struct PRICE \*s);

void struct\_AddToFile (FILE \*f, struct PRICE s);

void file\_PrintStruct (char \*name);

/////////////////////////////////////////

// Основная функция //

/////////////////////////////////////////

int main ()

{

// Подготовительный этап //

unsigned short int answer = 0;

FILE \*f;

if (!fsize(DEFAULT\_FILE\_NAME)) { f = fopen(DEFAULT\_FILE\_NAME, "w"); fclose(f); } // Создать файл, если он не существует

// Выполнение основной части программы //

while (answer != MENU\_EXIT)

{

printf("Меню программы:\n");

printf(" (%d) Очистить файл.\n", MENU\_CLEAR\_FILE );

printf(" (%d) Добавить структуры в файл.\n", MENU\_ADD\_STRUCTS );

printf(" (%d) Вывести содержимое файла.\n", MENU\_PRINT\_STRUCTS);

printf(" (%d) Найти элемент.\n", MENU\_SEARCH\_STRUCT);

printf(" (%d) Выйти из программы.\n", MENU\_EXIT );

printf("\n");

printf("$ "); scanf("%hd", &answer);

switch (answer)

{

case MENU\_EXIT:

printf("Осуществляется выход из программы...\n");

break;

case MENU\_CLEAR\_FILE:

printf("Очистка файла от старых данных...");

file\_clear(DEFAULT\_FILE\_NAME);

printf("УДАЧА\n");

break;

case MENU\_PRINT\_STRUCTS:

printf("Запуск функции печати структур из файла.\n");

file\_PrintStructs(DEFAULT\_FILE\_NAME);

break;

case MENU\_ADD\_STRUCTS:

printf("Запущена функция добавления структур в файл.\n");

file\_AddStructs(DEFAULT\_FILE\_NAME);

break;

case MENU\_SEARCH\_STRUCT:

printf("Запущена функция поиска структуры в файле.\n");

file\_PrintStruct(DEFAULT\_FILE\_NAME);

break;

}

printf("Нажмите на \"%s\" для продолжения...\n", "h");

while ( 'h' != getchar());

}

// Завершающий этап //

return 0;

}

/////////////////////////////////////////

// Блок инициализации функций //

/////////////////////////////////////////

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

\* @brief Определяет размер файла в байтах. Возращает 0, если размер файла равен нулю или если файл невозможно открыть для чтения

\*

\* @param name Путь к файлу

\* @return unsigned long int Размер файла в байтах

\*/

unsigned long int fsize (char \*name)

{

FILE \*f = fopen(name, "r"); if (!f) return 0;

fseek(f, 0, SEEK\_END);

unsigned long int l = ftell(f);

fclose(f);

return l;

}

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

\* @brief Функция очистки файла.

\*

\* @param name Путь к файлу.

\*/

void file\_clear (char \*name)

{

fclose(fopen(name, "w"));

return;

}

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

\* @brief Сохраняет структуру в данный файл.

\*

\* @param f Данный файл.

\* @param s Структура для записи.

\*/

void struct\_AddToFile (FILE \*f, struct PRICE s)

{

fwrite(s.name, sizeof(char), CHAR\_SIZE \* NAME\_SIZE, f);

fwrite(s.market, sizeof(char), CHAR\_SIZE \* MARKET\_SIZE, f);

fwrite(&s.price, sizeof(s.price), 1, f);

return;

}

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

\* @brief Получение структуры из файла.

\*

\* @param f Данный файл.

\* @param s Структура, куда записать данные.

\*/

int struct\_GetFromFile (FILE \*f, struct PRICE \*s)

{

fread(s->name, sizeof(char), CHAR\_SIZE \* NAME\_SIZE, f);

fread(s->market, sizeof(char), CHAR\_SIZE \* MARKET\_SIZE, f);

return fread(&(s->price), sizeof(s->price), 1, f);

}

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

\* @brief Функция печати всех структур из файла.

\*

\* @param name Имя данного файла.

\*/

void file\_PrintStructs (char \*name)

{

printf("Проверка файла на существование данных...");

if (fsize(name) < 1)

{

printf("ПОТРАЧЕНО: файл пуст или недоступен для чтения.\n");

return;

}

printf("УДАЧА\n");

printf("Осуществляется проверка валидности сохранённых данных...");

if ( fsize(name) % sizeof(struct PRICE) )

{

printf("ПОТРАЧЕНО: данные в файле не соответствуют стандартам.\n");

return;

}

printf("УДАЧА\n");

FILE \*f = fopen(name, "r");

struct PRICE s; unsigned long int counter = 1;

while (struct\_GetFromFile(f, &s))

{

printf("Товар №%lu\n", counter++);

printf(" \* Наименование: \"%s\".\n", s.name );

printf(" \* Поставщик: \"%s\".\n", s.market);

printf(" \* Стоимость товара: %lu.\n", s.price );

printf("\n");

}

fclose(f);

}

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

\* @brief Функция добавления структур в файл.

\*

\* @param name Имя данного файла.

\*/

void file\_AddStructs (char \*name)

{

FILE \*f;

printf("Попытка открыть файл...");

if (!(f = fopen(DEFAULT\_FILE\_NAME, "a")))

{

printf("ПОТРАЧЕНО: файл недоступен для записи.\n");

return;

}

printf("УДАЧА\n");

struct PRICE s;

char answer = 'y';

size\_t nl = NAME\_SIZE, ml = MARKET\_SIZE;

printf("Перед началом ввода нажмите на \"%s\" для продолжения...\n", "h"); while ( 'h' != getchar());

while (tolower(answer) != 'n')

{

printf("А теперь введите цену товара:\n");

printf("$ "); scanf("%lu", &s.price);

getchar();

printf("Введите Наименование товара:\n");

printf("$ "); fseek(stdin, 0, SEEK\_END); scanf("%s", s.name);

printf("Введите Название организации:\n");

printf("$ "); fseek(stdin, 0, SEEK\_END); scanf("%s", s.market);

struct\_AddToFile(f, s);

printf("Отлично! Новый продукт добавлен в файл.\n");

answer = '\n';

while (answer == '\n')

{

printf("Продолжить ввод? [Y/n] ");

scanf("%c", &answer);

printf("\n");

}

}

fclose(f);

file\_SortStructs(name);

return;

}

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

\* @brief Функция, которая сортирует структуры в данном файле.

\*

\* @param name Имя данного файла.

\*/

void file\_SortStructs (char \*name)

{

FILE \*f;

if (!(f = fopen(name, "r+b"))) return;

struct PRICE s1, s2;

int c = fsize(DEFAULT\_FILE\_NAME) / sizeof(struct PRICE);

for (int j = c - 1; j > 0; j--)

{

for (int i = 0; i < j; i++)

{

fseek(f, i \* sizeof(struct PRICE), SEEK\_SET);

struct\_GetFromFile(f, &s1);

struct\_GetFromFile(f, &s2);

if (strcmp(s1.name, s2.name) > 0)

{

fseek(f, i \* sizeof(struct PRICE), SEEK\_SET);

struct\_AddToFile(f, s2);

struct\_AddToFile(f, s1);

}

}

}

fclose(f);

return;

}

/\*\*

\* @brief Функция поиска структуры в файле. Найденная структура выводится на экран.

\*

\* @param name Имя данного файла.

\*/

void file\_PrintStruct(char \*name)

{

FILE \*f;

if (!(f = fopen(name, "r+b"))) return;

struct PRICE s\_user, s\_file;

int isFound = 0;

printf("Введите наименование товара: "); fseek(stdin, 0, SEEK\_END); scanf("%s", s\_user.name);

while (!feof(f))

{

struct\_GetFromFile(f, &s\_file);

if (!strcmp(s\_user.name, s\_file.name))

{

isFound = 1;

break;

}

}

if (isFound)

{

printf("Элемент найден:\n");

printf(" \* Наименование: \"%s\".\n", s\_file.name);

printf(" \* Поставщик: \"%s\".\n", s\_file.market);

printf(" \* Стоимость: %lu.\n\n", s\_file.price);

} else {

printf("Элемент не найден.\n\n");

}

fclose(f);

return;

}

# Описание тестовых примеров

Тестовые примеры представлены в Приложении В на Рисунках В.1 – В.6.

На Рисунке В.1 показано главное меню приложения.

На Рисунке В.2 показана функция очистки/создания нового файла сохранений.

На Рисунке В.3 показана работоспособность функции добавления элемента в файл. Пользователем были введены данные: “89”, “КонсервыЗФлот”, “ЗФлот”.

На Рисунке В.4 продемонстрировано содержимое файла. Файл также содержал другие элементы. Все элементы в файле отсортированы по полю “Наименование” в алфавитном порядке.

На Рисунках В.5 и В.6 показана работа функции поиска элемента в файле. В первом случае ввёл название существующего в файле товара, поэтому программа вывела информации о товаре. Во втором случае товар в файле не существует, поэтому программа вывела ошибку.

# Вывод

В ходе данной лабораторной работы были получены навыки разработки приложений, умеющих работать с бинарными типизированными файлами: чтение данных из файла с помощью fread, запись в файл с помощью fwrite и сортировка этих данным по определённому критерию, поиск в бинарном файле с помощью функции fseek. Также были получены навыки работы со структурами в языках C/C++.