**Лабораторная работа № 18. Работа с файлами на языке С**

*Файл* – это набор данных, размещенный на внешнем носителе и рассматриваемый в процессе обработки как единое целое. Каждый файл завершается маркером конца файла (end-of-file marker или EOF) или указанным числом байтов, записанным в служебную структуру данных.

В программу должна быть включена директива **#include <stdio.h>**.

|  |  |
| --- | --- |
| **Задание** | **Краткие теоретические сведения** |
| 1. Изучить *форматированный* ввод и вывод при работе с *текстовыми* файлами на языке С, выполнив программу, записанную в данном пункте.  Дополнить программу операторами чтения файла и вывода на экран его содержимого. | Файл открывается с помощью функци **fopen**, общий вид которой:  **fopen(const char \*filename, const char \*mode)**  Здесь **filename** − имя файла, **mode** − режим открытия файла.  Режимы открытия файла: **r** − открывает файл для чтения (если файл не существует, то вызов  завершается ошибкой); **w** − открывает пустой файл для записи (если файл существует, его содержимое удаляется); **a** − открывает файл для записи в конец файла (создает файл, если он не существует); **r+** − открывает для чтения и записи в существующий файл; **w+** − открывает пустой файл для чтения и записи (если файл существует, его содержимое удаляется); **a** − открывает файл для чтения и добавления в конец. Могут быть также добавлены символы **t** (открыть файл в текстовом режиме) либо **b** (открыть файл в бинарном режиме).  Используется также функция, общий вид которой:  **fopen\_s( FILE\*\* pFile, const char \*filename, const char \*mode)**, где **pFile** − указатель на файловый указатель.  Файл закрывается с помощью функции **fclose**.  **#include <stdio.h>**  **#include <iostream>**  **int main()**  **{ setlocale(LC\_ALL,"");**  **int a; errno\_t err;**  **FILE \*f;**  **err = fopen\_s(&f,"a.txt", "w");**  **if (err != 0)**  **{ perror("Невозможно создать файл\n");**  **return EXIT\_FAILURE;**  **}**  **for(a = 0; a < 70; a += 3)**  **{ fprintf(f, "%d, ", a);**  **}**  **printf("Данные записаны в файл test.txt\n");**  **fclose(f);**  **return 0;**  **}**  Функции для форматированного ввода и вывода: **fprintf()**, **fscanf()**.  Эти функции схожи с аналогичными функциями, работающими с клавиатурой и консолью, но предназначены для работы с файлами.  Пример. Записать в текстовый файл с именем **test.txt**, который находится в текущей папке проекта, числа от 0 до 100, кратные 3.  В программе **"w"** означает, что файл открывается для записи и, если он существует, то содержимое файла стирается.  Если файл не открывается, то в программе выдается сообщение.  Тип **errno\_t** возвращает в соответствующей переменной код возникающей при работе с файлом ошибки.  Функция **perror()** интерпретирует значение глобальной переменной ERRNO в строку и выводит эту строку на стандартный поток вывода с сообщением, указанным в параметре функции.  Макрос EXIT\_FAILURE используется для возвращения кода неудачного завершения программы. |
| 2. Изучить *строковый* ввод и вывод при работе с *текстовыми* файлами на языке С, выполнив программу, записанную в данном пункте.  Дополнить программу операторами чтения файла и вывода на экран его содержимого. | Функции для строкового ввода и вывода: **fgets()** − чтение из файла одной строки текста полностью, если ее длина меньше 50 символов, **fputs()** − запись строки в файл (функция, в отличие от **puts()** сама не помещает в конце строки **'\n'**).  **#include <iostream>**  **#include <stdio.h>**  **int main ()**  **{ setlocale(LC\_ALL,"");**  **FILE \*mf; char str[50]; errno\_t err;**  **char \*estr;**  **printf("Введите текст "); gets(str);**  **fopen\_s (&mf, "tst.txt", "w"); //Открытие для записи**  **fputs(str, mf); fputs("\n", mf);**  **fclose (mf);**  **err = fopen\_s(&mf, "tst.txt", "r"); //Открытие для чтения**  **if (err != NULL) //Проверка открытия файла**  **{ printf ("Ошибка открытия файла\n"); return -1;**  **}**  **else**  **printf ("\nСчитаны строки:\n");**  **estr = fgets (str, sizeof(str), mf); //Чтение строки файла**  **if (estr == NULL) //Конец файла или ошибка чтения?**  **if (feof(mf) != 0)**  **printf ("\nЧтение файла закончено\n");**  **else**  **{ printf ("\nОшибка чтения из файла\n");**  **return -1;**  **}**  **puts (str);**  **if ( fclose (mf) == EOF)**  **printf ("Ошибка закрытия\n");**  **return 0;**  **}**  Пример. Записать в текстовый файл с именем **tst.txt** строку символов и прочитать ее. |
| 3. Выполнить программу, записанную в данном пункте, которая осуществляет запись *блока* информации в файл.  Дополнить программу операторами чтения из файла и вывода на экран его содержимого. | С помощью функций **fwrite()** и **fread()** осуществляется блоковый ввод/вывод и работа с бинарными файлами.  Для записи информации служит функция:  **fwrite(void** **\*ptr**, **int** **size**, **int** n, **FILE \*pfile);**  где **\*ptr** – указатель на область памяти, в которой размещаются считываемые данные; **size** – размер одного считываемого элемента; **n** – количество считываемых элементов.  Для считывания информации используется функция, общий вид которой:  **#include <iostream>**  **#include <stdio.h>**  **int main(void)**  **{ setlocale(LC\_CTYPE, "Russian");**  **FILE \*fp;**  **errno\_t err;**  **char const \*st = "привет";**  **err = fopen\_s(&fp,"a.bin", "w");**  **if (err != 0)**  **{**  **perror("ошибка открытия a.txt");**  **return EXIT\_FAILURE;**  **}**  **fwrite(st, strlen(st), 1, fp);**  **printf("Записан элемент\n");**  **fclose(fp);**  **return 0;**  **}**  **fread( void \*ptr, int size, int n, FILE \*pfile);**  В случае успешного считывания информации функция возвращает число прочитанных элементов.  Пример программы на языке **С**, которая открывает файл **a.bin** в текущей папке проекта (если файла нет, то создает его), записывает блок информации, состоящий из строки символов (слово «привет») и закрывает файл.  Функция **fwrite** записывает содержимое переменной **st** длиной **strlen(st)**, **fp** − указатель на файл, в который производится запись. |
| 4. Выполнить программу, записанную в правой части, которая реализует операции *позиционирования* в файле и *блоковый* вывод.  Изменить программу так, чтобы выводились на экран подряд все предложения с 1 по **n**. | Функция **long ftell(FILE \*fp)** возвращает **текущую позицию** в файле. Если текущая позиция не определена, функция возвращает  -1L.  Функция **int  fseek(FILE \*fp, long pos, int mode)** устанавливает текущую позицию в файле на байт с номером **pos**. Возвращает 0 при успешном позиционировании и **-1 (EOF)** - в случае неудачи. Параметр **mode** определяет, относительно чего отсчитывается текущая позиция в файле (0 − относительно начала файла, 1 − относительно текущей позиции, 2 − относительно конца файла).  Пример. Пусть в текстовом файле “**b.bin**” (в текущей папке проекта) записано несколько предложений. Конец предложения обозначен точкой. Требуется ввести номер предложения **number** и вывести из текстового файла на экран предложение с этим номером.  **#include <iostream>**  **using namespace std;**  **void main()**  **{ setlocale(LC\_CTYPE, "Russian");**  **int m = 0, number = 0, pr = 0;**  **long fsize;**  **char pd;**  **FILE \*fd;**  **errno\_t err;**  **err = fopen\_s(&fd,"b.txt", "r");**  **if (err != 0)**  **{ perror("ошибка открытия a.txt");**  **return;**  **}**  **fseek(fd, 0L, SEEK\_END);**  **fsize = ftell(fd);**  **fseek(fd, 0L, SEEK\_SET);**  **printf("Введите номер предложения ");**  **scanf("%d", &number);**  **for (int k = 1; k <= fsize; k++)**  **{ fread((void\*)&pd, sizeof(char), 1, fd);**  **if (pd == '.') pr++;**  **if ((number - 1) == pr) m++;**  **if (number == pr)**  **{ long pos1 = k - m - 1;**  **if(number != 1) pos1++;**  **fseek(fd, pos1, SEEK\_SET);**  **printf("%d-е предложение: ", number);**  **for (int i = 0; i <= m; i++)**  **{ fread((void\*)&pd, sizeof(char), 1, fd);**  **printf("%c", pd);**  **}**  **break;**  **}**  **}**  **if (number > pr)**  **printf("Такого номера нет");**  **fclose(fd);**  **}**  Здесь в программе открывается файл, с помощью функции **fseek** устанавливается текущая позиция в файле на конец, и функцией **ftell** определяется размер содержимого файла.  Далее с помощью функции **fseek** устанавливается текущая позиция в файле на начало и происходит чтение информации функцией **fread** и поиск точки. |

5. В соответствии со своим вариантом разработать программы для условий, приведенных в таблице ниже, и изучить способы работы с файлами на ***языке С***.

| **№ варианта** | **Условие задачи** |
| --- | --- |
| **1** | 1. Даны два файла вещественных чисел с именами **fA** и **fB**, содержащие элементы прямоугольных матриц **M1** и **M2** по строкам, причем начальный элемент каждого файла содержит количество столбцов соответствующей матрицы. Создать файл той же структуры с именем **fC**, содержащий произведение матриц **M1·M2**. Если матрицы **M1** и **M2** нельзя перемножать, то оставить файл **fC** пустым.  2. Компоненты файла **f** – целые двухзначные числа (положительные и отрицательные). Получить файл **g**, образованный из **f** включением только чисел кратных **K**. |
| **2** | 1. Даны два файла вещественных чисел с именами **fA** и **fB**, содержащие элементы прямоугольных матриц **M1** и **M2** по строкам, причем начальный элемент каждого файла содержит количество столбцов соответствующей матрицы. Создать файл той же структуры с именем **fC**, содержащий сумму **M1+M2**.  2. Компоненты файла**f** – целые двухзначные (отличные от нуля) числа, причем 10 положительных чисел, 10 отрицательных, и т. д. Получить файл **g***,* в котором записаны сначала 5 положительных чисел, затем 5 отрицательных и т. д. |
| **3** | 1. Дан файл вещественных чисел, содержащий элементы квадратной матрицы (по строкам), причем начальный элемент файла содержит значение количества столбцов матрицы. Создать новый файл той же структуры, содержащий транспонированную матрицу.  2. Даны три файла целых чисел одинакового размера с именами **NameA**, **NameB** и **NameC**. Создать новый файл с именем **NameD**, в котором чередовались бы элементы исходных файлов с одним и тем же номером: **A0, B0, C0, A1, B1, C1, A2, B2, C2, ...** |
| **4** | 1. Компоненты файла **fileA** – целые отличные от нуля положительные и отрицательные числа. Получить файл **fileB**, состоящий из положительных чисел.  2. Компоненты файла **f** – целые двухзначные числа. Получить файл **g**, образованный из **f** включением только чисел больших некоторого числа, вводимого с клавиатуры. |
| **5** | 1. Даны два файла целых чисел, содержащие элементы квадратных матриц с именами **A** и **B** по строкам, причем начальный элемент каждого файла содержит количество столбцов соответствующей матрицы. Создать файл той же структуры с именем **C**, содержащий произведение матриц **А** и **В**.  2. Создать текстовый файл **F1** не менее, чем из 6 строк, и записать в него информацию. Скопировать в файл **F2** только те строки из **F1**, которые начинаются с буквы «**А**». |
| **6** | 1. Компоненты файла **fA** – вещественные числа (положительные и отрицательные). Определить и вывести на экран порядковый номер того из них, которое наиболее близко к введенному пользователем целому числу.  2. Создать текстовый файл **F1** не менее, чем из 6 строк, и записать в него информацию. Скопировать в файл **F2** только четные строки из **F1**. |
| **7** | 1. Даны три файла целых чисел одинакового размера с именами **NameA**, **NameB** и **NameC**. Создать новый файл с именем **NameD**, в который записать максимальные элементы исходных файлов с одним и тем же номером: **max(*A0*, *B0*, *C0)***, **max( *A1*, *B1*, *C1)***, **max(*A2*, *B2*, *C2)*, ...**  2. Даны три файла целых чисел одинакового размера с именами **A**, **B** и **C**. Создать новый файл с именем **D**, в котором чередовались бы элементы исходных файлов с одним и тем же номером: **a0, b0, c0, a1, b1, c1, a2, b2, c2,** ... . |
| **8** | 1. Дан файл целых чисел с элементами **A(i), i = 0, ..., N – 1** (**N** − размер файла). Заменить исходное расположение его элементов на следующее: **A(0), A(N – 1), A(1), A(N – 2), A(2), ... .**  2. Даны два файла целых чисел, содержащие элементы матрицы **A** размерности **n** x **n** и **B** размерности **n** x **1** по строкам. Начальный элемент каждого файла содержит количество столбцов соответствующей матрицы. Создать файл **C**, содержащий произведение матриц **А** и **В**. |
| **9** | 1. Компоненты файла **fileA** –  целые числа, значения которых повторяются.  Получить файл **fileB**, образованный из **fileA** исключением повторных вхождений одного и того же числа.  2. Создать текстовый файл **F1** не менее, чем из 4 строк, и записать в него информацию. Скопировать из файла **F1** в файл **F2** строки, количество символов в которых больше, чем заданное число, которое вводится с клавиатуры. |
| **10** | 1. Компоненты файла **fileA** –  целые отличные от нуля числа: **y1,... yn**. Вывести на экран два члена этой последовательности, среднее арифметическое которых ближе всего к числу **х**. Целое число **х** вводится с клавиатуры.  2.Создать текстовый файл **F1** не менее, чем из 5 строк, и записать в него информацию. Скопировать в файл **F2** только строки из **F1**, которые не содержат цифр. |
| **11** | 1. Компоненты файла **fileA** –  целые числа, значения которых повторяются.  Получить файл **fileB**, образованный из **fileA** числами, которые не повторяются.  2. Создать текстовый файл **F1** не менее, чем из 6 строк, и записать в него информацию. Скопировать в файл **F2** только те строки из **F1**, которые заканчиваются символом «**а**». |
| **12** | 1. Дан файл вещественных чисел, содержащий элементы квадратной матрицы по строкам, причем начальный элемент файла содержит значение количества столбцов матрицы. Создать новый файл, содержащий **k**-ую строку исходной матрицы.  2. Даны три файла целых чисел одинакового размера с именами **NameA**, **NameB** и **NameC**. Создать новый файл с именем **NameD**, в который записать минимальные элементы исходных файлов с одним и тем же номером: **min(A0, B0, C0)**, **min( A1, B1, C1)**, **min(A2, B2, C2), ...** |
| **13** | 1. Компоненты файла **fileA** –  целые числа, значения которых повторяются.  Получить файл **fileB**, образованный из **fileA** числами, которые встречаются в **fileA** более двух раз.  2. Создать текстовый файл **F1** не менее, чем из 5 строк, и записать в него информацию. Скопировать в файл **F2** только те строки из **F1**, которые начинаются с цифры. |
| **14** | 1. Дан файл вещественных чисел, содержащий элементы квадратной матрицы по строкам, причем начальный элемент файла содержит значение количества столбцов матрицы. Создать новый файл той же структуры, содержащий **k**-ый столбец исходной матрицы.  2. Компоненты файла **f** – целые числа. Получить файл **g**, образованный из файла **f** исключением повторных вхождений одного и того же числа. |
| **15** | 1. Дан файл вещественных чисел, содержащий элементы квадратной матрицы по строкам, причем начальный элемент файла содержит значение количества столбцов матрицы. Создать новый файл той же структуры, содержащий **k**-ый столбец исходной матрицы.  2. Создать текстовый файл **F1** не менее, чем из 8 строк, и записать в него информацию. Скопировать из файла **F1** в файл **F2** строки, начиная с номера **k** до **k + 3**. Целое число **k** вводится с клавиатуры. |
| **16** | 1. Даны два файла целых чисел с именами **fileA** и **fileB**. Получить новый файл с именем **fileC**, который содержит сумму элементов файлов **fileA** и **fileB**.  2. Компоненты файла**file1** – целые двухзначные (отличные от нуля) числа, причем сначала записаны 5 положительных чисел, затем 5 отрицательных, и т. д. Получить файл **file2***,* в котором записаны числа из файла **file1**,сначала 10 положительных чисел, затем 10 отрицательных и т. д. |



[В начало практикума](#_Содержание)