МИНИСТЕРСТВО ТРАНСПОРТА И КОММУНИКАЦИЙ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Белорусский государственный университет транспорта»

Кафедра «Информационно-управляющие системы и технологии»

Отчет  
по практическим работам

по дисциплине «Основы алгоритмизации и программирования»

Выполнила Проверил

студентка группы ГИ-11 ст. преп. каф. «ИУСиТ»

Зайцев И. С. Голдобина Т. А.

Гомель, 2020

Оглавление

[Практическая работа № 21 Тема: «Разработка, отладка и выполнение программы обработки файлов» 3](#_Toc37518198)

[1.1 Задания 3](#_Toc37518199)

[1.2 Контрольные вопросы 12](#_Toc37518200)

# Практическая работа № 21 Тема: «Разработка, отладка и выполнение программы обработки файлов»

### **Цель**

Разработать, отладить и выполнить программы для обработки файлов.

## Задания

### Задание 1

### Условие

Составить функцию, с помощью которой открыть текстовый файл, записать в него символ и закрыть файл. Открыть тот же файл для чтения, считать символ и вывести в консоль.  
Выполнить тесты для цифр, латинских букв верхнего и нижнего регистров и кириллических букв верхнего и нижнего регистров.

### Листинг

void task\_1()

{ int arr[N];

FILE \*fp;

fp=fopen("file.txt", "w");

fputc('a', fp);

fclose(fp);

fp=fopen("file.txt", "r");

while (fgets (arr, N, fp) != NULL)

printf("%s", arr);

printf("\n");

fclose(fp);

}

### Результат

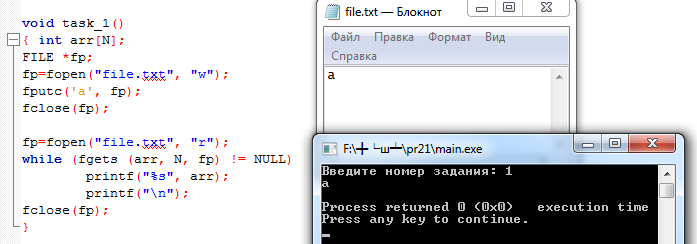


Рисунок – Решение задания 1

### Задание 2

### Условие

Составить функцию, с помощью которой открыть текстовый файл, записать в него строку, массив чисел и закрыть файл. Далее открыть тот же файл, считать информацию и вывести в консоль.

### Листинг

void task\_2()

{

int i,n, arr[N];

char a[20];

int b[] = {1, 2, 5};

FILE \*file1;

scanf("%s", a);

file1 = fopen("file.txt", "w");

fprintf(file1, "%s", a);

for(i=0; i<3; i++)

fprintf(file1, " %d", b[i]);

fclose(file1);

file1=fopen("file.txt", "r");

while (fgets (arr, N, file1) != NULL)

printf("%s", arr);

printf("\n");

fclose(file1);

}

### Результат

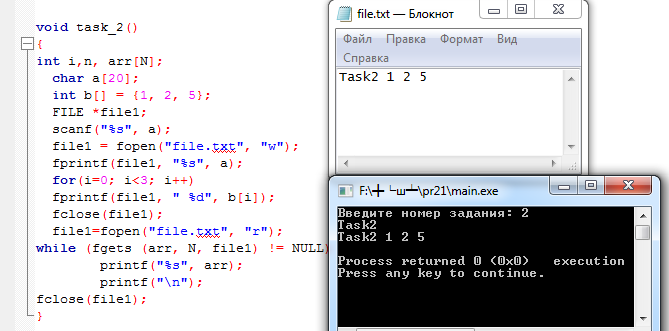


Рисунок 2 – Решение задания 2

### Задание 3

### Условие

Создать файл (вручную). Записать в этот файл число, символы (латинские и кириллица), строки (латинскими и кириллицей). Сохранить, закрыть. Составить функцию, с помощью которой созданный файл открыть, все считать, вывести в консоль и закрыть.

### Листинг

void task\_3()

{ char arr[N];

FILE \*file1;

file1=fopen("task\_3.txt", "r");

while (fgets (arr, N, file1) != NULL)

printf("%s", arr);

printf("\n");

fclose(file1);

}

### Результат

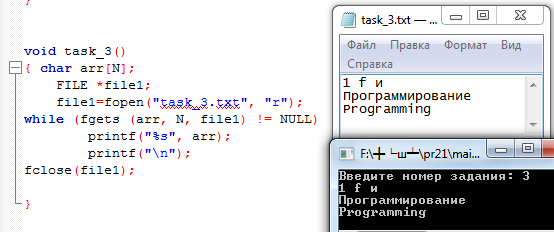


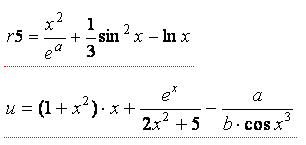
Рисунок 3 – Решение задания 3

### Задание 4

### Условие

Преобразовать программу, разработанную ранее (ПР № 5, Задание 1) таким образом, чтобы входные данные считывались из внешних файлов, а выходные - записывались во внешний файл. Условие задания привести полностью, включая расчетные формулы.

ПР №5, Задание 1:



### Листинг

void task\_4()

{ int x, a, b;

float r5, u;

FILE \*inFile;

FILE \*outFile;

inFile=fopen("input4.txt", "r");

fscanf(inFile, "%d", &x);

fscanf(inFile, "%d", &a);

fscanf(inFile, "%d", &b);

fclose(inFile);

printf("Данные из файла: %d %d %d\n", x,a,b);

r5=((x\*x)/exp(a))+ 1/3.\*sin(x)\*sin(x)-log(x);

u=(1+x\*x)\*x+ (exp(x)/(2\*x\*x+5))- (a/(b\*cos(x)\*cos(x)\*cos(x)));

printf("Ответы: r5=%f, u=%f", r5, u);

outFile=fopen("output4.txt", "w");

fprintf(outFile, "%f\n", r5);

fprintf(outFile, "%f", u);

fclose(outFile);

printf("\n");

}

### Результат

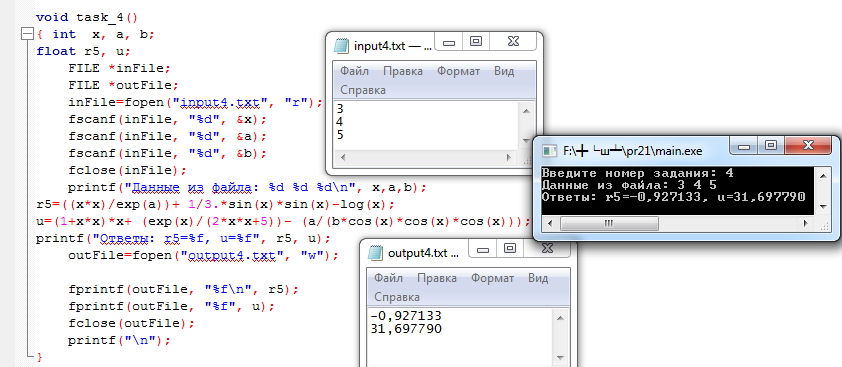


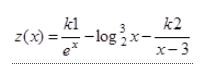
Рисунок 4 – Решение задания 4

### Задание 5

### Условие

Преобразовать программу, разработанную ранее (ПР № 8, Задание 1) таким образом, чтобы входные данные считывались из внешних файлов, а выходные - записывались во внешний файл. Условие задания привести полностью, включая расчетные формулы.

ПР №8, Задание 1:



при каждом значении переменной x Є [1.5 ; 6] , изменяющейся с шагом  Δx=0.3.

Использовать цикл **for**. Значения констант k1 и k2 - произвольные. Учесть возможность возникновения исключений во время выполнения программы.

### Листинг

void task\_5()

{ FILE \*inFile;

FILE \*outFile;

inFile=fopen("input5.txt", "r");

int k1, k2, i=0;

float z,x ;

fscanf(inFile, "%d", &k1);

fscanf(inFile, "%d", &k2);

printf("Входные данные: %d %d\n", k1, k2);

outFile=fopen("output5.txt", "w");

printf("x\t\tz(x)\n");

fclose(inFile);

fprintf(outFile, "x\t\tz(x)\n");

for(x=1.5; x<=6.3; x+=0.3){

i++;

if(i==6)

{

printf("%.3f\t\t--\n", x);

fprintf(outFile, "%.3f\t\t--\n", x);

continue;

}

z=k1\*1./exp(x) - pow(log2(x\*1.),3) - k2\*1./(x-3);

printf("%.3f\t\t%.3f\n", x, z);

fprintf(outFile, "%.3f\t\t%.3f\n", x ,z);}

fclose(outFile);

}

### Результат

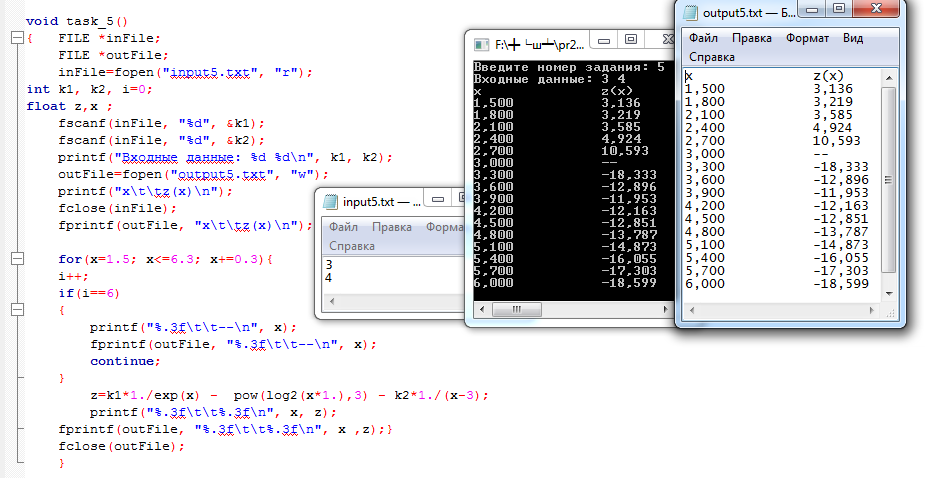


Рисунок 5 - Решение задания 5

### Задание 6

### Условие

Преобразовать программу, разработанную ранее (ПР № 11, Задание 1) таким образом, чтобы входные данные считывались из внешних файлов, а выходные - записывались во внешний файл. Условие задания привести полностью, включая расчетные формулы.

ПР № 11, Задание 1:

Ввести одномерный массив *U* из десяти элементов и число *n*. Для каждого элемента массива вычислить . Вывести полученный массив. Определить произведение отрицательных элементов массива *D*. Поменять местами значения последнего и минимального элементов *D*.

Вывод результатов в процессе отладки программы осуществлять на экран поэлементно в цикле.

### Листинг

void task\_6()

{ FILE \*inFile;

FILE \*outFile;

inFile=fopen("input6.txt", "r");

int U[10], min=0, zam=0;

int i, D[10], n,x=1,z=0;

fscanf (inFile, "%d", &n);

for (i=0; i<10; i++)

{

fscanf(inFile, "%d", &U[i]);

}

fclose(inFile);

outFile=fopen("output6.txt", "w");

for (i=0; i<10; i++)

{D[i]=(1/tan(U[i]))/(n+1)-log(abs(n\*U[i]));}

for (i=0; i<10; i++)

{

fprintf(outFile,"D[%d]=%d\n",i, D[i]);

} for (i=0; i<10; i++) {

if (D[i]<0)

{

x\*=D[i];

}

} fprintf (outFile, "Произведение отрицательных элементов массива: %d\n\n",x);

for (i=0; i<10; i++){

if (min>=D[i]) {min=D[i]; z=i;}

}

zam=D[z];

D[z]=D[9];

D[9]=zam;

fprintf(outFile, "Замена минимального и последнего элемента массива:\n");

for (i=0; i<10; i++)

{

fprintf(outFile,"D[%d]=%d\n",i, D[i]);

}

fclose(outFile);

}

### Результат

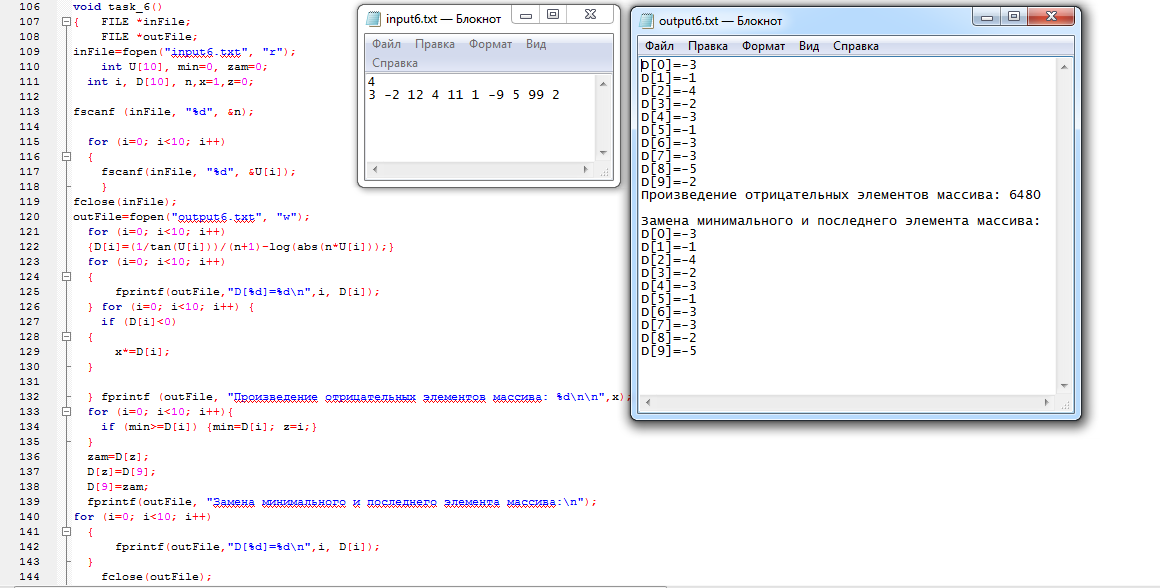


Рисунок 6 - Решение задания 6

### Задание 7

### Условие

Преобразовать программы, разработанные ранее (ПР № 15, Задания 1, 2) таким образом, чтобы входные данные считывались из внешних файлов, а выходные - записывались во внешний файл. Условия заданий привести полностью.

ПР № 15, Задание 1:

Составить программу, в которой вводится строка (не менее 20 символов, осмысленная, содержащая строчные и прописные буквы и цифры, которые могут быть изменены по заданию).

Выводится копия строки, в которой: сделать первые буквы каждого слова строчными, а остальные - прописными, заменить букву "к" на "т" (с учетом регистра).

ПР № 15, Задание 2:

Составить программу, в которой вводится строка (не менее 20 символов, осмысленная, содержащая строчные и прописные буквы и цифры).

Выводится строка, в которой: каждая третья буква, кроме 9-й, заменяется на прописную.

### Листинг

void task\_7\_1()

{FILE \*fp;

int i;

char string[1024], \*mass\_char;

mass\_char=string;

fp=fopen("input7\_1.txt", "r");

fgets(string, 80, fp);

fclose(fp);

printf("%s", string);

for(i=0; i<strlen(string); i++ ){

if(\*mass\_char == 107){\*mass\_char = 116;}

else if(\*mass\_char == 75){\*mass\_char = 84;}

if(\*mass\_char>=65 && \*mass\_char<=90){

\*mass\_char+=32;

fprintf(stdout, "%c", \*mass\_char);}

else if(\*mass\_char==32){

fprintf(stdout, "%c", \*mass\_char);

mass\_char++;

\*mass\_char=tolower(\*mass\_char);

if(\*mass\_char == 107){\*mass\_char = 116;}

fprintf(stdout, "%c", \*mass\_char);

i++;}

else if(\*mass\_char>=97 && \*mass\_char<=122){

\*mass\_char-=32;

fprintf(stdout, "%c", \*mass\_char);}

else {fprintf(stdout, "%c", \*mass\_char);}

mass\_char++;}

fp=fopen("output7\_1.txt", "w");

fprintf(fp, "%s", string);

fclose(fp);

}

void task\_7\_2()

{FILE \*fp;

char string[1024];

char \*mass\_char, ch;

mass\_char=string;

fp=fopen("input7\_2.txt","r");

fgets(string, "%s", fp);

fclose(fp);

printf("\n\n%s\n", string);

for(int i=2; i<7; i+=3 )

{

if (96<string[i] && string[i]<123 || 159<string[i] && string[i]<176)

{

string[i]-=32;

}

else if (223<string[i] && string[i]<240)

{

string[i]-=80;

}

}

for(int i=11; i<strlen(string); i+=3 )

{

if (96<string[i] && string[i]<123 || 159<string[i] && string[i]<176)

{

string[i]-=32;

}

else if (223<string[i] && string[i]<240)

{

string[i]-=80;

}

}

for(int i=0; i<strlen(string); i++ )

{

printf("%c", string[i]);

}

fp=fopen("output7\_2.txt","w");

fprintf(fp, "%s", string);

fclose(fp);

}

### Результат

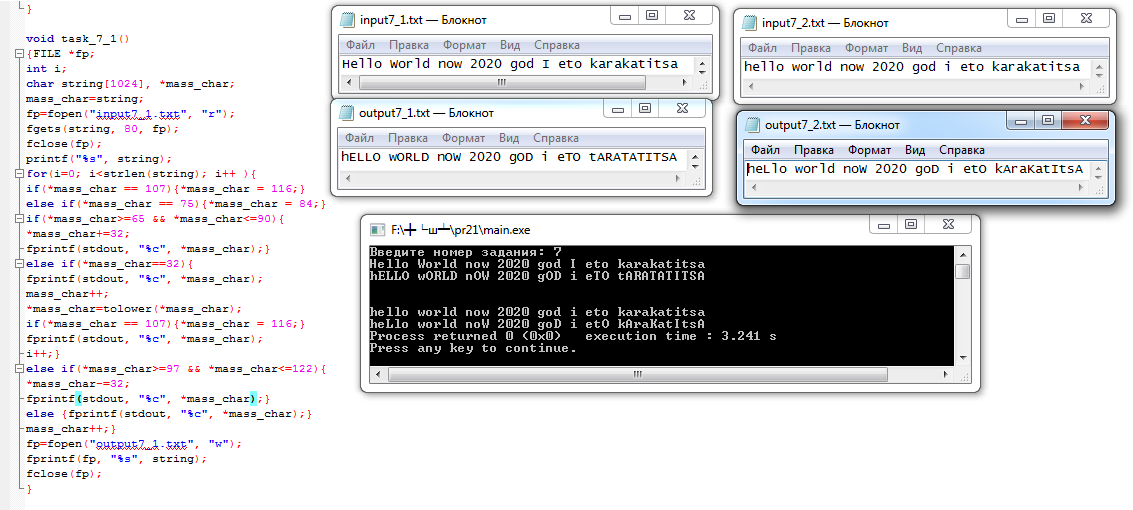


Рисунок 7- Решение задания 7

## Контрольные вопросы

1. Где применяются файлы данных (назначение)?

Работа с файлами, содержащими результаты измерений, в т. ч. полученные автоматически, или отсканированные данные, требующие дальнейшего чтения и анализа. Взаимодействие с другими программами, данные, сохраненные в файлы могут быть импортированы в другие приложения и наоборот. Сохранение информации для повторного использования, сравнения и анализа. Обмен данными и результатами.

1. Классификация файлов данных.

Файлы подразделяют на входные (с исходными данными) и выходными (с результатами работы программы).

1. Этапы работы с файлами данных.

Создать файловый объект (переменную, поток), выполнить чтение / запись файла, закрыть файл (очистить поток).

1. Что такое поток на C/C++?

Файловая система языка С предназначена для работы с самыми разными устройствами, в том числе терминалами, дисководами и накопителями на магнитной ленте. Даже если какое-то устройство сильно отличается от других, буферизованная файловая система все равно представит его в виде логического устройства, которое называется потоком.

1. Виды потоков файловой системе C/C++.

Текстовый и бинарный.

1. Особенности текстового потока.

Текстовый поток — это последовательность символов. В стандарте С считается, что текстовый поток организован в виде строк, каждая из которых заканчивается символом новой строки. Однако в конце последней строки этот символ не является обязательным. В текстовом потоке по требованию базовой среды могут происходить определенные преобразования символов. Например, символ новой строки может быть заменен парой символов — возврата каретки и перевода строки. Поэтому может и не быть однозначного соответствия между символами, которые пишутся (читаются), и теми, которые хранятся во внешнем устройстве. Кроме того, количество тех символов, которые пишутся (читаются), и тех, которые хранятся во внешнем устройстве, может также не совпадать из-за возможных преобразований.

1. Указатель файла. Объявление указателя.

Указатель файла — это то, что соединяет в единое целое всю систему ввода/вывода языка С. Указатель файла — это указатель на структуру типа FILE. Он указывает на структуру, содержащую различные сведения о файле, например, его имя, статус и указатель текущей позиции в начало файла. В сущности, указатель файла определяет конкретный файл и используется соответствующим потоком при выполнении функций ввода/вывода. Чтобы выполнять в файлах операции чтения и записи, программы должны использовать указатели соответствующих файлов. Чтобы объявить переменную-указатель файла, используйте такого рода оператор: FILE \*fp;

1. Функция fopen(). Назначение.

Функция fopen() открывает поток и связывает с этим потоком определенный файл. Затем она возвращает указатель этого файла. Чаще всего (а также в оставшейся части этой главы) под файлом подразумевается дисковый файл. Прототип функции fopen() такой:

FILE \*fopen(const char \**имя\_файла*, const char \**режим*);

где имя\_файла — это указатель на строку символов, представляющую собой допустимое имя файла, в которое также может входить спецификация пути к этому файлу. Строка, на которую указывает *режим*, определяет, каким образом файл будет открыт.

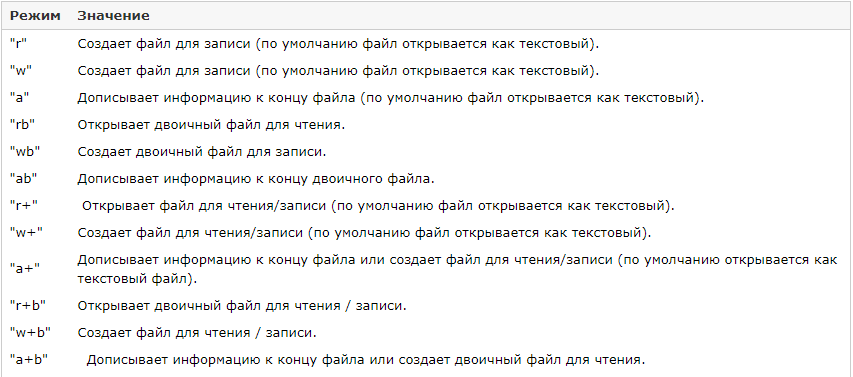
1. Функция fclose(). Назначение.

Функция fclose() закрывает поток, который был открыт с помощью вызова fopen().Функция fclose() записывает в файл все данные, которые еще оставались в дисковом буфере, и проводит, так сказать, официальное закрытие файла на уровне операционной системы. Отказ при закрытии потока влечет всевозможные неприятности, включая потерю данных, испорченные файлы и возможные периодические ошибки в программе. Функция fclose() также освобождает блок управления файлом, связанный с этим потоком, давая возможность использовать этот блок снова. Так как количество одновременно открытых файлов ограничено, то, возможно, придется закрывать один файл, прежде чем открывать другой. Прототип функции fclose() такой:

int fclose(FILE \*уф);

где уф — указатель файла, возвращенный в результате вызова fopen(). Возвращение нуля означает успешную операцию закрытия. В случае же ошибки возвращается EOF. Чтобы точно узнать, в чем причина этой ошибки, можно использовать стандартную функцию ferror() (о которой вскоре пойдет речь). Обычно отказ при выполнении fclose() происходит только тогда, когда диск был преждевременно удален (стерт) с дисковода или на диске не осталось свободного места.

1. Режимы открытия файла.



1. Функции для чтения символов из файлов. Описание.

Для ввода символа также имеются две эквивалентные функции: getc() и fgetc(). Обе определяются для сохранения совместимости со старыми версиями С. В этой книге используется getc() (которая обычно реализуется в виде макроса), но если хотите, применяйте fgetc().

Функция getc() записывает символы в файл, который с помощью fopen() уже открыт в режиме для чтения. Прототип этой функции следующий:

int getc(FILE \*fp);

где fp— это указатель файла, имеющий тип FILE и возвращенный функцией fopen(). Функция getc() возвращает целое значение, но символ находится в младшем байте. Если не произошла ошибка, то старший байт (байты) будет обнулен.

1. Функция ferror(). Назначение и применение.

Функция ferror() определяет, произошла ли ошибка во время выполнения операции с файлом. Прототип этой функции следующий:

int ferror(FILE \*fp);

где fp — допустимый указатель файла. Она возвращает значение true (истина), если при последней операции с файлом произошла ошибка; в противном же случае она возвращает false (ложь). Так как при любой операции с файлом устанавливается свое условие ошибки, то после каждой такой операции следует сразу вызывать ferror(), а иначе данные об ошибке могут быть потеряны.

### Вывод по практической работе

Разработал, отладил и выполнил программы для обработки файлов.