Министерство образования и науки РФ
Государственное образовательное учреждение
высшего профессионального образования

Тульский государственный университет

КАФЕДРА АВТОМАТИКИ И ТЕЛЕМЕХАНИКИ

ДИНАМИЧЕСКИЕ ПЕРЕМЕННЫЕ

Лабораторная работа № 6

по курсу «Программирование на ЯВУ»

Вариант № 4

Выполнил: студент группы 220601 _____ Белым А.А.
(подпись)

Проверил: _____ Сулимова В.В.
(подпись)

Тула 2011

Цель работы

Цель работы заключается в том, что научиться создавать, использовать и уничтожать динамические переменные (ссылки). Требуется реализовать лабораторную работу номер 11 с помощью динамических переменных, т. е. сначала из файла считываются все данные в память, обрабатываются и оттуда записываются в файл.

Задание

Определить, какая буква чаще всего встречается в заданном тексте.

Теоретическая справка

Схема алгоритма

На рисунке 1 представлена схема алгоритма ввода данных, поиска самых частых букв и их вывода.

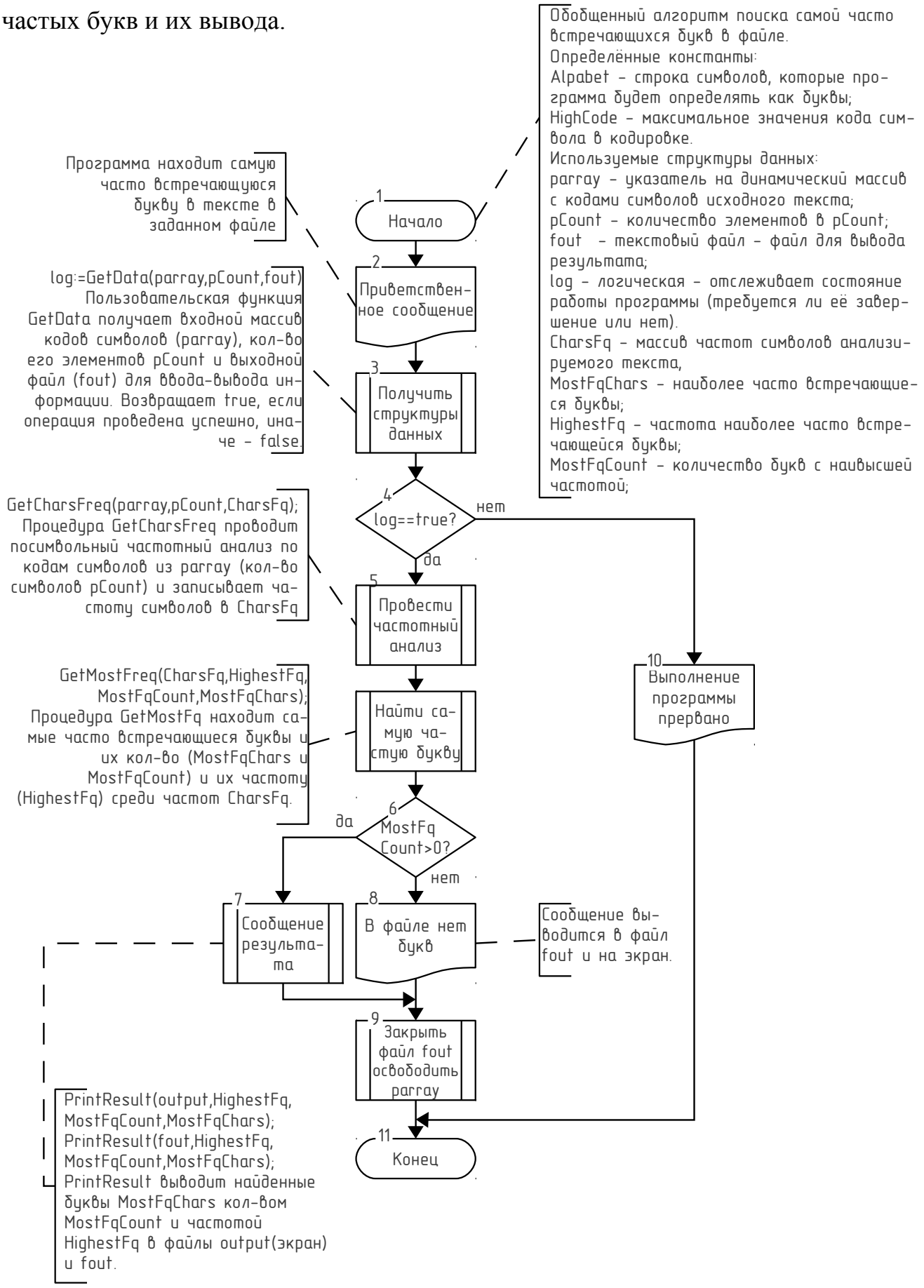

Рисунок 1 — Блок-схема обобщенного алгоритма поиска самой часто встречающейся буквы

На рисунке 2 представлена схема алгоритма получения необходимых для работы структур данных.

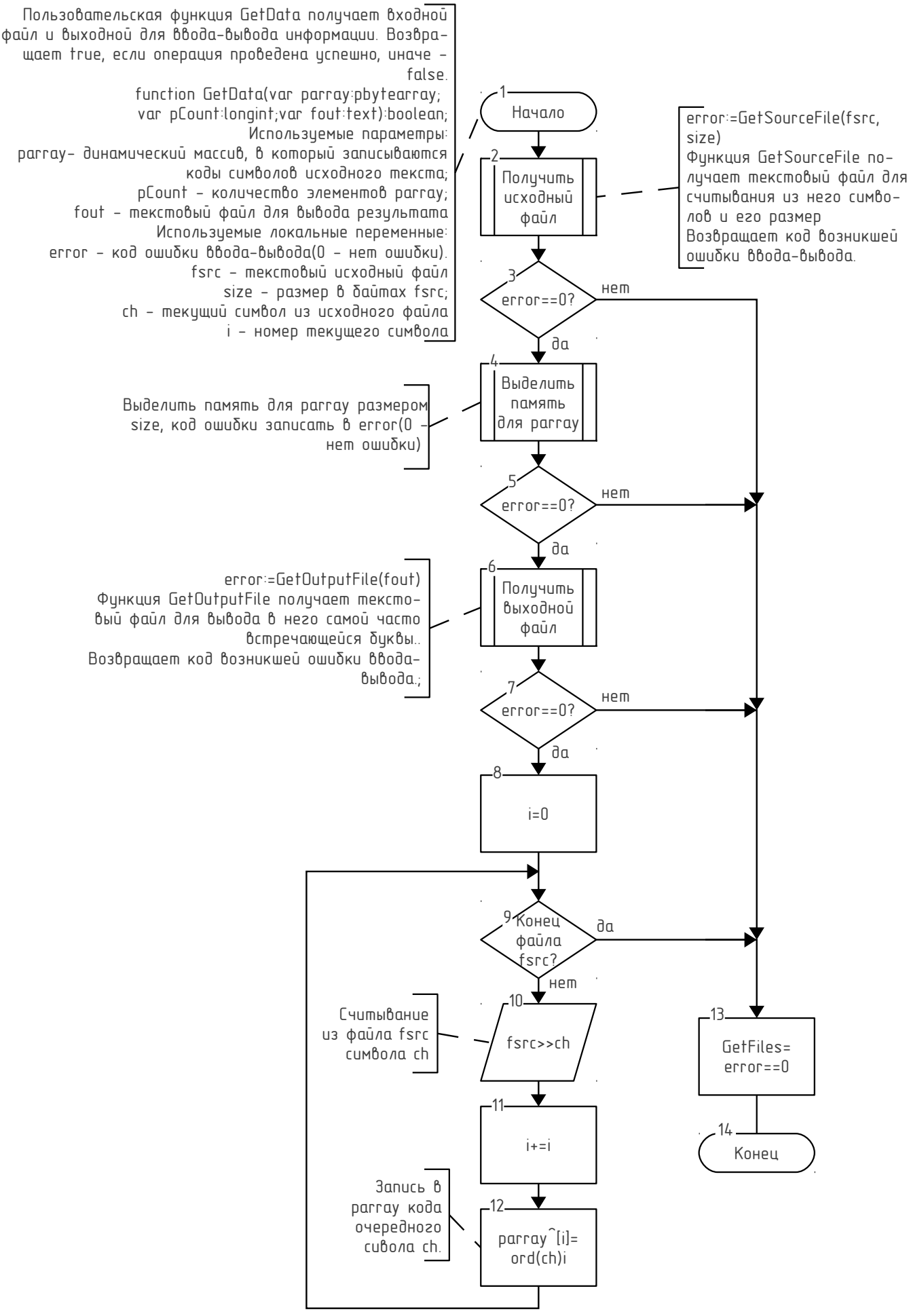

Рисунок 2 — Блок-схема алгоритма получения необходимых данных

На рисунке 3 представлена схема алгоритма получения файла-источника.

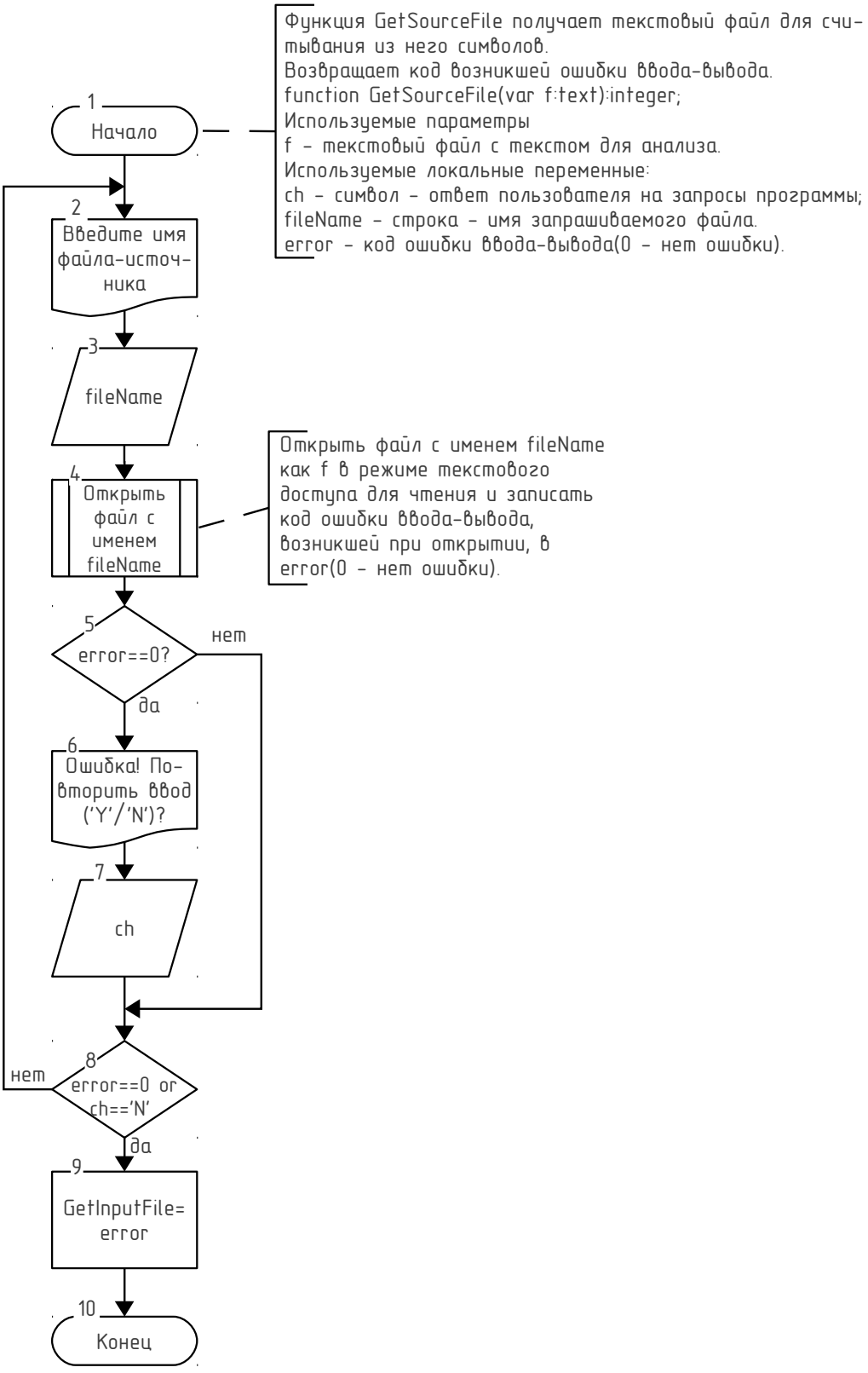

Рисунок 3 — Блок-схема алгоритма получения файла-источника

На рисунке 4 представлена схема алгоритма получения выходного файла.

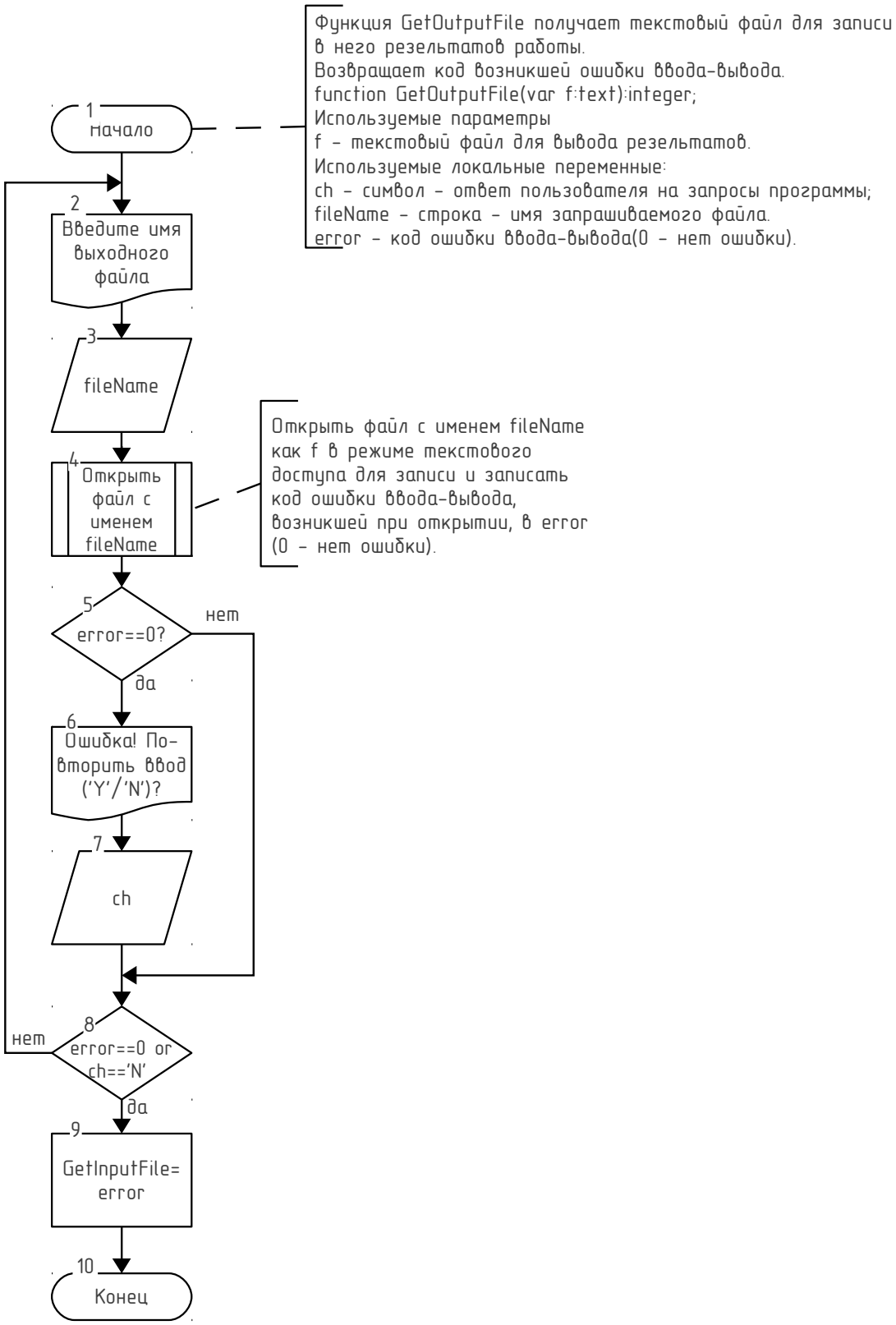

Рисунок 4 — Блок-схема алгоритма получения выходного файла

На рисунке 5 представлена схема посимвольного частотного анализа исходного текста с использованием кодов символов из динамического массива.

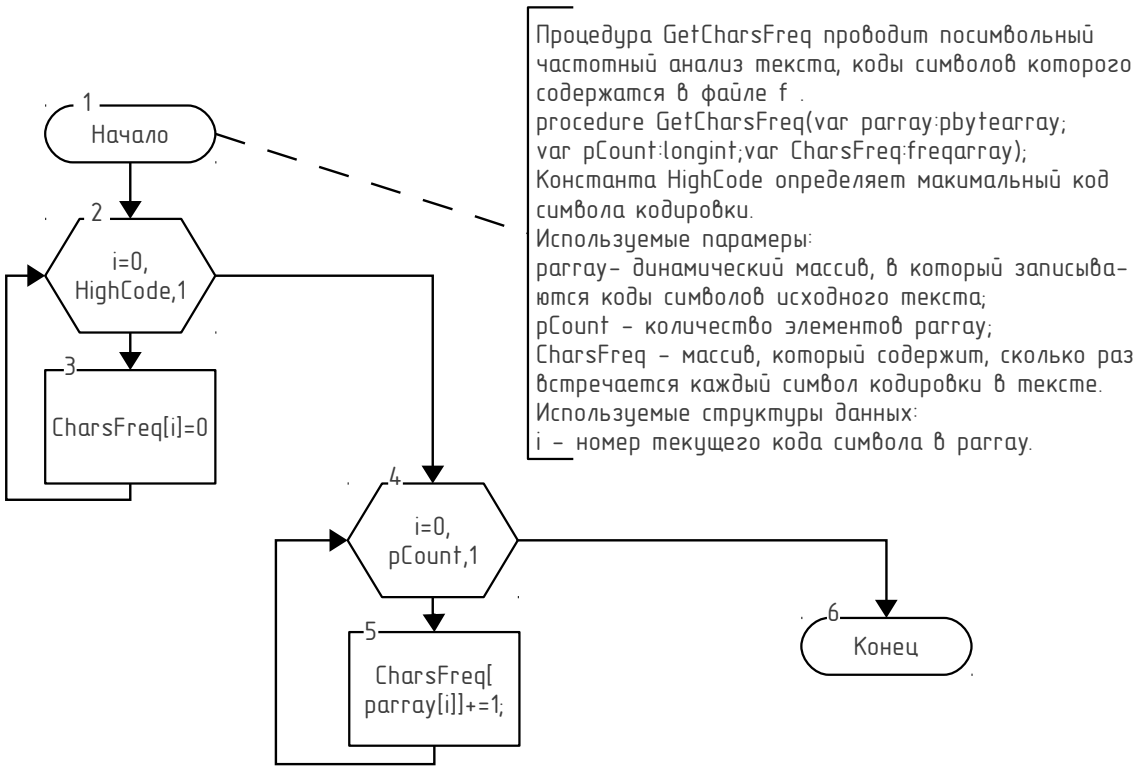

Рисунок 5 — Блок-схема алгоритма посимвольного частотного анализа

На рисунке 6 представлена схема алгоритма поиска самых частых букв при помощи данных частотного анализа.

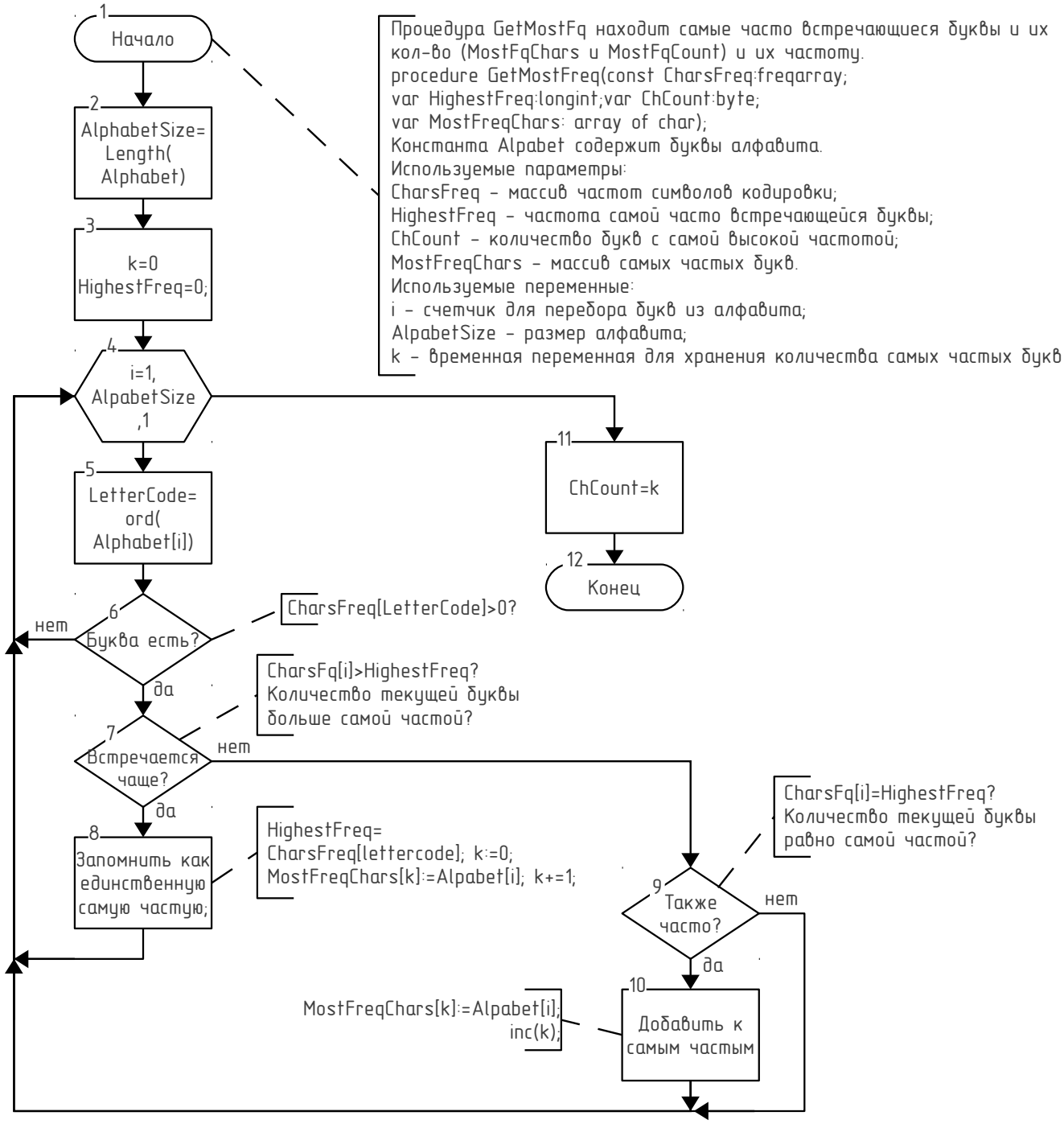

Рисунок 6 — Блок-схема алгоритма поиска самых часто встречающихся букв

На рисунке 7 представлена схема алгоритма вывода в файл самых часто встречающихся букв исходного текста.

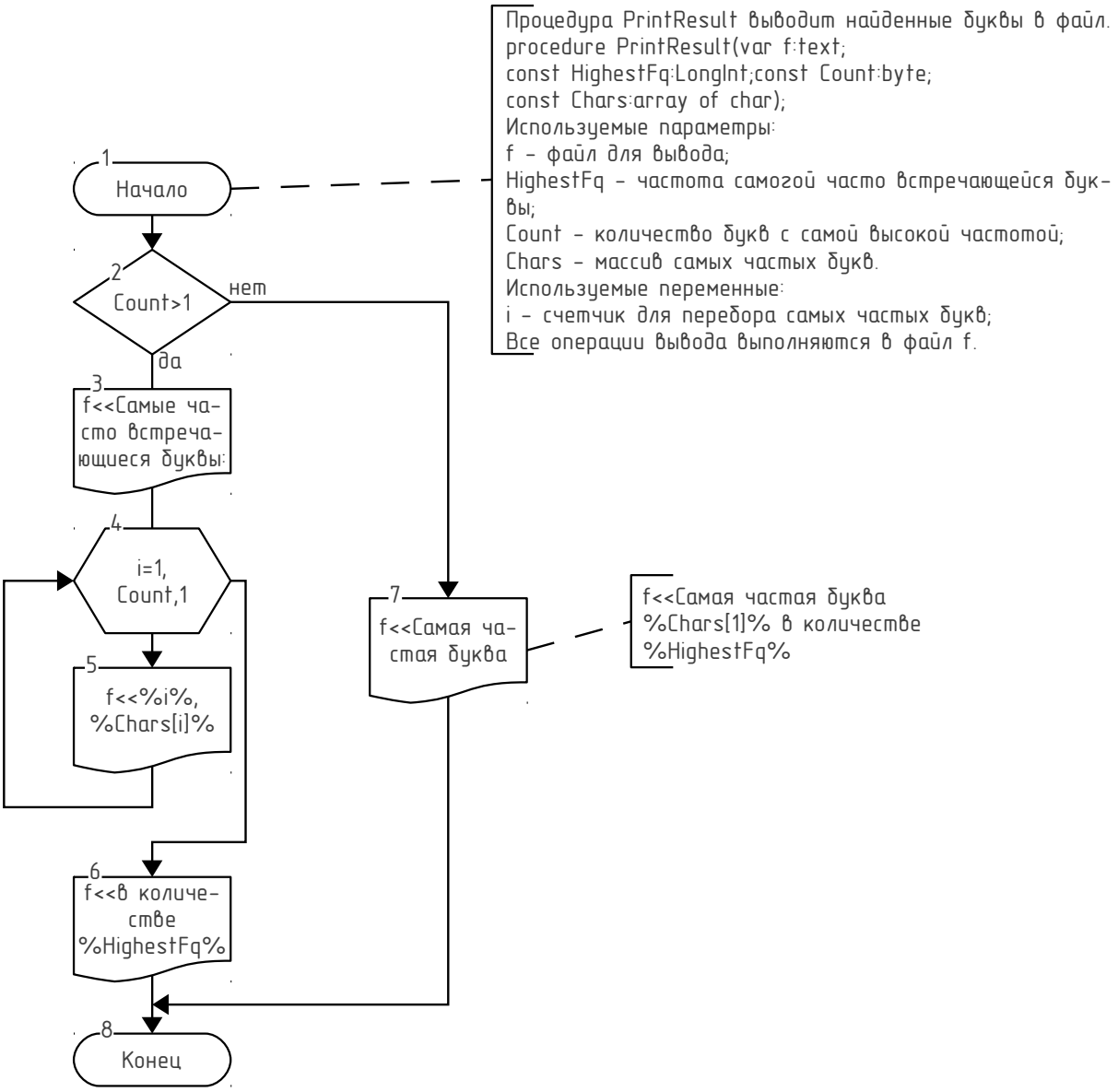

Рисунок 7 — Схема алгоритма вывода в файл самых частых букв

Инструкция пользователю

Данная программа позволяет найти самую часто используемую букву в текстовом файле.

Для работы программы необходимо ввести имена некоторых файлов. Первый файл - это файл, в котором находится текст для поиска. Его имя длинной не более 255 символов нужно передать программе. Далее передайте программе имя файла для вывода результатов поиска. Внимание! Если файл с указанным именем существует, то вся информация в нем будет стерта! Пожалуйста, проверьте имя файла, так как восстановить потерянные данные буде невозможно. В случае неправильного ввода имени файла-источника (если такой не существует), или если он имеет слишком большой размер, или при невозможности получить доступ к файлу для вывода, имеется возможность ввести заново имя этого файла, ответив на запрос программы о продолжении ввода 'Y', или отказаться от повторного ввода и завершить программу, ответив 'N'.

Если найдена всего одна самая часто встречающаяся буква, то будет выведена она и её частота в указанный файл и на экран; если существует несколько букв с самой высокой частотой появления, то они будут выведены нумерованным списком, и в конце будет выведена их частота, в указанный файл и на экран.

Инструкция программисту

При создании программы поиска одногруппников были предприняты следующие действия.

В основной части программы определены константы:

1. Alphabet - строка, символы которой должны определяться как буквы;

2. HighCode - максимальное значения кода символа в кодировке. Программа работает корректно с русскими символами только в 8-битных кодировках, поэтому значение HighCode, скорее всего должно быть нестрого меньше 255.

3. maxmemalloc - максимальное количество памяти, выделяемое для динамического массива.

Определены типы:

1. bytearray как array [1..maxmemalloc] of byte - тип массива, в который считываются коды символов из файла источника.

2. pbytearray как ^bytearray - указатель на динамический массив кодов символов;

3. freqarray как array [0..HighCode] of longint - тип массива частот символов кодировки в тексте.

Были введены структуры данных, описание которых представлено в таблице 1.

Таблица 1 - Структуры данных, используемые в в основной части программы поиска самой частой буквы в тексте

имя	тип	предназначение
parray	pbytearray	Динамический массив с кодами символов.
pCount	longint	Количество кодов в динамическом массиве.
fout	text	Выходной файл для записи результатов работы
HighestFq	longint	Частота наиболее часто встречающейся буквы.
MostFqCount	byte	Количество наиболее часто встречающихся букв.
CharsFq	freqarray	Массив частот символов кодировки в тексте.
MostFqChars	array [byte] of char	Наиболее часто встречающиеся буквы текста.

log	boolean	Указывает на отсутствие ошибок (значение true) ввода данных или желания пользователя прервать программу.

Кроме того, в процессе создания вышеуказанной программы были определены следующие подпрограммы:

1. Функция GetData запрашивает у пользователя все необходимые для работы файлы и выделяет память для динамического массива, и в динамический массив коды символов из файла-источника. Возвращает true, если все необходимые файлы получены, иначе возвращается false.

function GetData(var parray:pbytearray; var pCount:longint;var fout:text):boolean;

В теле функции через локальные функции GetSourceFile, GetOutputFile запрашиваются файл-источник с текстом, также получается его размер и текстовый выходной файл соответственно. Если на каком-то этапе возникла ошибка, то последующие этапы не выполняются и возвращается false. После получения файла источника делается попытка выделения памяти для динамического массива. Если память не была выделена, то последующие этапы не выполняются и возвращается false. Далее из файла-источника считывается каждый символ, и коды этих символов записываются в динамический массив. После завершения этой операции файл-источник закрывается. Далее функция завершает работу и возвращает true.

Используемые функцией параметры-переменные приведены в таблице 2, локальные переменные - в таблице 3.

Таблица 2 - Параметры-переменные функции получения файлов

имя	**тип**	**предназначение**
parray	pbytearray	Динамический массив с кодами символов.
pCount	longint	Количество кодов в динамическом массиве.
fout	text	Выходной файл для записи результатов.

Таблица 3 - Локальные переменные функции получения файлов

имя	**тип**	**предназначение**
error	integer	Код ошибки, возникшей при открытии файла,и ли при выделении памяти(код 0 - нет ошибки).

fsrc	text	Файл-источник с исходным текстом.
i	byte	Код текущего символа, считанного из текста.
ch	char	Текущий символ, считанный из текста.

Для получения файла каждого типа были введены следующие функции:

1.1 Функция GetSourceFile запрашивает у пользователя имя файла-источника с анализируемым текстом, и, если возможно, открывает его для чтения. Возвращает код возникшей при открытии ошибки ввода-вывода (код 0 - нет ошибки). Использует константу maxmemalloc.

function GetSourceFile(var f:text):integer;

В теле функции в цикле с постусловием происходит запрос у пользователя имени файла-источника, и проводится попытка его окрытия в режиме для чтения. Если файл открылся успешно, то этот файл переоткрывается как типизированный символьный файл, определяется его размер, и проверяется не больше ли он maxmemalloc. Если в обоих случаях условия выполнены, функция завершает работу и передает открытый файл, и значение функции, равное 0. Если нет, то производится запрос пользователя о продолжении работы. Если пользователь ответил 'N', функция завершает работу и возвращает значение ошибки ввода-вывода, Иначе операция повторяется до достижения двух вышеуказанных условий.

Используемые функцией параметры-переменные приведены в таблице 4, локальные переменные - в таблице 5.

Таблица 4 - Параметры-переменные функции получения файла-источника

имя	**тип**	**предназначение**
f	text	Файл-источник с исходным текстом.
size	longint	Размер файла-источника в байтах.

Таблица 5 - Локальные переменные функции получения файла-источника

имя	**тип**	**предназначение**
ch	char	Содержит ответ пользователя на запросы программы о повторении ввода данных.
fileName	string	Имя файла-источника.

error	integer	Код ошибки ввода-вывода, возникшей при открытии файла(код 0 - нет ошибки) .
tempf	file of char	Типизированный символьный файл для определения размеров файла-источника.

1.2 Функция GetOutputFile запрашивает у пользователя имя выходного файла, и пытается открыть его для записи. Возвращает код возникшей при открытии ошибки ввода-вывода (код 0 - нет ошибки).

function GetOutputFile(var f:text):integer;

В теле функции в цикле с постусловием происходит запрос у пользователя имени выходного файла, и проводится попытка его окрытия в режиме для записи. Если файл открылся успешно, функция завершает работу и передает открытый файл , и значение функции, равное 0. Если нет, то производится запрос пользователя о продолжении работы. Если пользователь ответил 'N', функция завершает работу и возвращает значение ошибки ввода-вывода, Иначе операция повторяется до достижения двух вышеуказанных условий.

Используемые функцией параметры-переменные приведены в таблице 6, локальные переменные - в таблице 9.

Таблица 6 - Параметры-переменные функции получения текстового выходного файла

имя	**тип**	**предназначение**
f	text	Выходной текстовый файл.

Таблица 7 - Локальные переменные функции получения текстового выходного файла

имя	**тип**	**предназначение**
ch	char	Содержит ответ пользователя на запросы программы о повторении ввода данных.
fileName	string	Имя выходного файла.
error	integer	Код ошибки ввода-вывода, возникшей при открытии файла(код 0 - нет ошибки) .

2. Функция GetCharsFreq проводит частотный анализ текста, коды которого записаны в динамическом массиве. Использует константу HighCode.

procedure GetCharsFreq(var f:bytefile;var CharsFreq:freqarray);

В начале массив частот символов кодировки (от 0 до HighCode) обнуляется. Затем из динамического массива считываются коды символов, и элемент массива частот, имеющий индекс, равный коду символа, увеличивается на единицу.

Используемые функцией параметры-переменные приведены в таблице 8; локальные переменные - в таблице 9.

Таблица 8 - Параметры-переменные процедуры посимвольного частотного анализа

имя	**тип**	**предназначение**
parray	pbytearray	Динамический массив с кодами символов.
pCount	longint	Количество кодов в динамическом массиве.
CharsFreq	freqarray	Массив частот символов кодировки в тексте.

Таблица 9 - Локальные переменные процедуры посимвольного частотного анализа

имя	**тип**	**предназначение**
i	byte	Код текущего символа, считанный из типизированного файла.

3. Функция GetMostFreq с помощью данных частотного анализа находит самые частые буквы в тексте. Использует константу Alphabet.

procedure GetMostFreq(const CharsFreq:freqarray; var HighestFreq:longint;

var ChCount:byte; var MostFreqChars:array of char);

При инициализации количество самых частых букв и наивысшая частота обнуляются.

В цикле с параметром перебираются буквы из строки Alphabet, и если частота её появления не равна 0 (элемент массива частот имеющий индекс, равный коду символа, не равен 0), то тогда если её частота больше максимальной, то эта частота запоминается как максимальная, количество букв устанавливается в 1, и буква добавляется в массив; если равна, то количество букв увеличивается на 1, и буква добавляется в массив.

Используемые функцией параметры-константы приведены в таблице 10; параметры-переменные - в таблице 11; локальные переменные - в таблице 12.

Таблица 10 - Параметры-константы процедуры поиска самых часто встречающихся букв

имя	тип	предназначение
CharsFreq	freqarray	Массив частот символов кодировки в тексте.

Таблица 11 - Параметры-переменные процедуры поиска самых часто встречающихся букв

имя	тип	предназначение
HighestFreq	longint	Частота наиболее часто встречающейся буквы.
ChCount	byte	Количество наиболее часто встречающихся букв.
MostFreqChars	array of char	Наиболее часто встречающиеся буквы текста.

Таблица 12 - Локальные переменные процедуры поиска самых часто встречающихся букв

имя	тип	предназначение
i	byte	Переменная-счетчик для перебора ,букв - элементов строки Alphabet.
k	byte	Временная переменная для хранения количества наиболее часто встречающихся букв.
AlphabetSize	byte	Количество букв в Alphabet.
lettercode	byte	Код текущей буквы из Alphabet.

4. Процедура PrintResult выводит найденные самые частые буквы в тексте.

```
procedure PrintResult(var f:text;const HighestFq:LongInt;const Count:byte;
                                                const Chars:array of char);
```

Если в переданном массиве всего одна буква, то она и её частота просто выводятся в соответствующем сообщении; иначе выводятся нумерованный список букв, а в конце - частота их появлению.

Используемые процедурой параметры-константы приведены в таблице 13, параметры-константы - в таблице 14, локальные переменные - в таблице 15 .

Таблица 13 - Параметры-константы процедуры вывода самых часто встречающихся букв

имя	тип	предназначение
HighestFq	LongInt	Частота наиболее часто встречающейся буквы.
Count	byte	Количество наиболее часто встречающихся букв.
Chars	array of char	Наиболее часто встречающиеся буквы текста.

Таблица 14 - Параметры-переменные процедуры вывода самых часто встречающихся букв

имя	тип	предназначение
f	text	Выходной файл для записи результатов.

Таблица 15 - Локальные переменные процедуры вывода самых часто встречающихся букв

имя	тип	предназначение
i	byte	Переменная-счетчик для обработки массива.

Текст программы

Ниже представлен текст программы, написанной на языке Turbo Pascal 7, которая находит самую часто встречающуюся букву в тексте.

```
const {алфавит букв}
      Alphabet:string='qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFGHJKLZXCVBNM'+
            'ёйцукенгшщзхъфывапролджэячсмитьбюЁЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЭЯЧСМИТЬБЮ';
const maxmemalloc=64*1024-1; {максимум выделения мамяти}
const highcode=255; {максимальный код символа}
type bytearray=array [1..maxmemalloc] of byte; {массив кодов символов}
type pbytearray=^bytearray;
type freqarray=array [0..highcode] of longint; {массив частот символов}
{-----ФУНКЦИЯ ПОЛУЧЕНИЯ ФАЙЛОВ-----}
{Параметры:
parray - массив с кодвами символов,pCount - количество элементов parray,
fout - файл для вывода.}
function GetData(var parray:pbytearray; var pCount:longint;var fout:text):boolean;

{-----ФУНКЦИЯ ПОЛУЧЕНИЯ ФАЙЛА ИСТОЧНИКА------}
function GetSourceFile(var f:text;var size:longint):integer;
var ch:char;error:integer;
    fileName:string;tempf:file of char;
begin
    repeat
        WriteLn('Введите имя файла для анализа');
        Write('Файл:');ReadLn(fileName);
        Assign(f,fileName);Assign(tempf,fileName);
        {$I-}
            Reset(f);
        {$I+}
        error:=ioresult;
        if (error<>0) or (fileName='') then begin
            error:=-1;
            WriteLn('Неправильное имя файла! Повторить ввод? <Y>/<N>');
            ReadLn(ch);
        end else begin
            Close(f);
                ReSet(tempf); size:=FileSize(tempf);
            close(tempf); Reset(f);
                if (size>maxmemalloc) or (size>maxavail) then begin
                    error:=1;
                    WriteLn('Файл слишком большой! Повторить ввод? <Y>/<N>');
                    ReadLn(ch);
            end;
```

```
            end;
    until (UpCase(ch)='N') or (error=0);
      GetSourceFile:=error;
end;

{-----ФУНКЦИЯ ПОЛУЧЕНИЯ ВЫХОДНОГО ФАЙЛА-----}
function GetOutputFile(var f:text):integer;
var ch:char; error:integer;
    fileName:string;
begin
    repeat
        WriteLn('Введите имя файла-результата.');
        Write('Файл:');ReadLn(fileName);
        Assign(f,fileName);
        {$I-}
            ReWrite(f);
        {$I+}
        error:=ioresult;
        if (error<>0) or (fileName='') then begin
            error:=-1;
            WriteLn('Ошибка при создании файла! Повторить ввод? <Y>/<N>');
            ReadLn(ch);
        end;
    until (UpCase(ch)='N') or (error=0);
    GetOutputFile:=error;
end;
var error:integer; fsrc:text; ch:char;Size:longint;var i:longint;
begin
    error:=GetSourceFile(fsrc,Size);
    if error=0 then begin
        GetMem(parray,Size);
        if parray=NIL then begin
            error:=-1;
            WriteLn('Ошибка выделения памяти!');
        end;
    end;
    if error=0 then
        error:=GetOutputFile(fout);
    if error=0 then begin
        Writeln('Считываются данные в оперативную память...');
        Write(0:3,'%');i:=0;
        while not EOF(fsrc) do begin
            Read(fsrc,ch);inc(i);
            parray^[i]:=ord(ch);Write(#13,i/size*100:3:0,'%');
```

```
        end;
        Close(fsrc);
        pCount:=i;
    end;
    writeln;
    GetData:=error=0;
end;

{-----ФУНКЦИЯ ПОСИМВОЛЬНОГО ЧАСТОТНОГО АНАЛИЗА-----}
{Параметры:
parray - массив с кодами символов, pCount - количество элементов parray,
CharsFreq  массив частот символов.}
function GetCharsFreq(var parray:pbytearray; var pCount:longint;var
CharsFreq:freqarray):boolean;
var  k:longint;
begin
    for k:=0 to highcode do
        CharsFreq[k]:=0;

    WriteLn('Выполнено:');
    Write(0:3,'%');
    for k:=1 to pCount do begin
        Write(#13,k/pCount*100:3:0,'%');
        inc(CharsFreq[parray^[k]]);
    end;

    WriteLn;
end;

{-----ФУНКЦИЯ ПОЛУЧЕНИЯ САМЫХ ЧАСТЫХ БУКВ-----}
{Параметры:
CharsFreq - массив частот символов, HighestFreq - наивысшая частота букв,
ChCount - количество букв, MostFreqChars - самые частые буквы}
procedure GetMostFreq(const CharsFreq:freqarray;
                            var HighestFreq:longint;var ChCount:byte; var
MostFreqChars:array of char);
var k,i:byte; AlphabetSize,lettercode:byte;
begin
    AlphabetSize:=length(alphabet);
      k:=0;
    HighestFreq:=0;
      for i:=1 to AlphabetSize do begin
        lettercode:=ord(Alphabet[i]);
        if (CharsFreq[lettercode]>0)  then begin
```

```
            if (CharsFreq[lettercode]>HighestFreq) then begin
                HighestFreq:=CharsFreq[lettercode];
                k:=0;
                MostFreqChars[k]:=Alphabet[i]; inc(k);
                  end else
                if CharsFreq[lettercode]=HighestFreq then begin

                   MostFreqChars[k]:=Alphabet[i]; inc(k);
                end;
          end;
        end;
        ChCount:=k;
end;

{-----ФУНКЦИЯ ВЫВОДА САМЫХ ЧАСТЫХ БУКВ НА ЭКРАН-----}
{Параметры:
f - файл для вывода, HighestFq - наивысшая частота букв,
Count - количество букв, Chars - буквы для вывода}
procedure PrintResult(var f:text;const HighestFq:LongInt;const Count:byte;
                            const Chars:array of char);
var i:byte;
begin
    if Count>1 then begin
            WriteLn(f,'Наиболее часто всречаются буквы');
            for i:=0 to Count-1 do begin
                  Write(f,i+1,'.''',Chars[i],''' ');
            end;
          WriteLn(f,#13#10,'в количестве ',HighestFq,' шт.');

      end else begin
            WriteLn(f,'Наиболее часто всречается буква',#10#13,
        '''',Chars[0],''' в количестве ',HighestFq,' шт.');
    end;
end;

{-----ОСНОВНАЯ ПРОГРАММА-----}
var log:boolean;MostFqCount:byte; HighestFq:LongInt; parray:pbytearray; fout:text;
    CharsFq:freqarray;MostFqChars:array[byte] of char;pCount:longint;
      i:byte;
BEGIN
    WriteLn('Программа находит самые часто встречающиеся буквы в тексте');
    WriteLn('и выводит результат на экран и в текстовый файл.');
    log:=GetData(parray,pCount,fout);
```

```
    if log then begin
        WriteLn('Выполняется поиск самой часто встречающейся буквы.');
        GetCharsFreq(parray,pCount,CharsFq);
        FreeMem(parray,pCount);
        GetMostFreq(CharsFq,HighestFq,MostFqCount,MostFqChars);
        if MostFqCount>0 then begin
            PrintResult(output,HighestFq,MostFqCount,MostFqChars);
            PrintResult(fout,HighestFq,MostFqCount,MostFqChars);
        end else begin
            WriteLn('В тексте нет букв.');
            WriteLn(fout,'В тексте нет букв.');
        end;
        Close(fout);
    end else
        WriteLn('Работа программы прервана.');
    WriteLn('Нажмите <Enter>...');
    ReadLn;
END.
```

Тестовые примеры

На рисунке 8 представлен пример работы программы для текста на русском языке размером примерно 90 Кбайт.

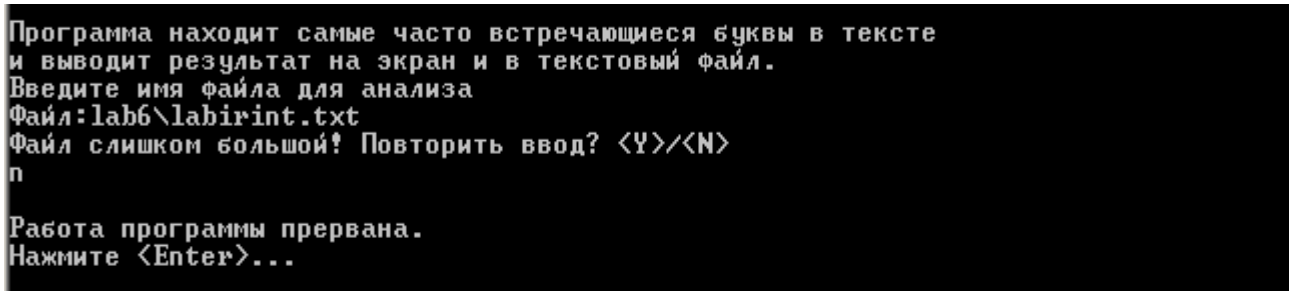

Рисунок 8 - Пример работы программы для произвольного файла

На рисунке 9 представлен пример работы программы для файла, содержащего строку “sssdddfffggghj”.

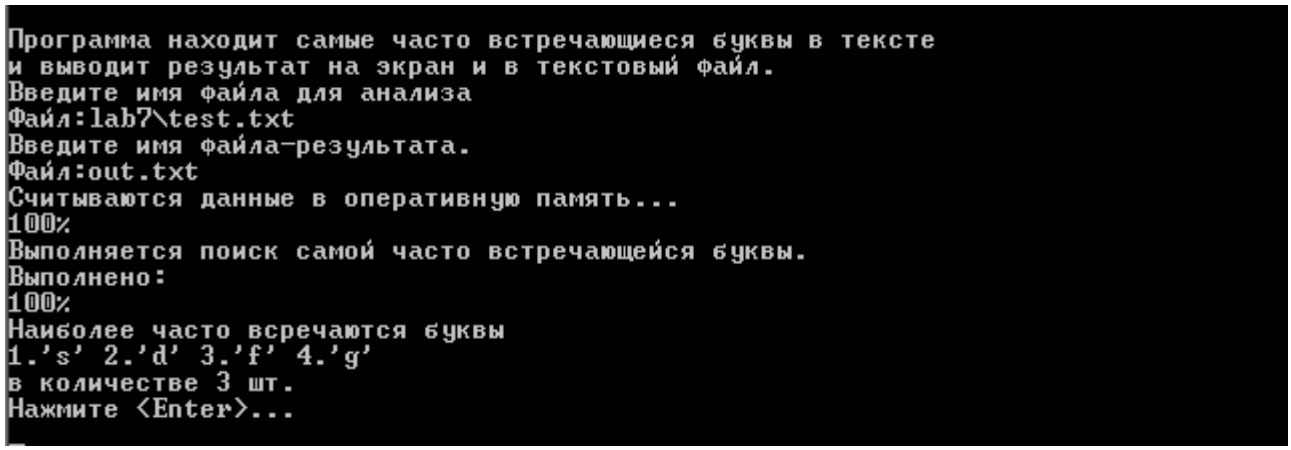

Рисунок 9 - Пример работы программы для файла с несколькими самыми частыми буквами

На рисунке 10 представлен результат обработки программой собственного текста.

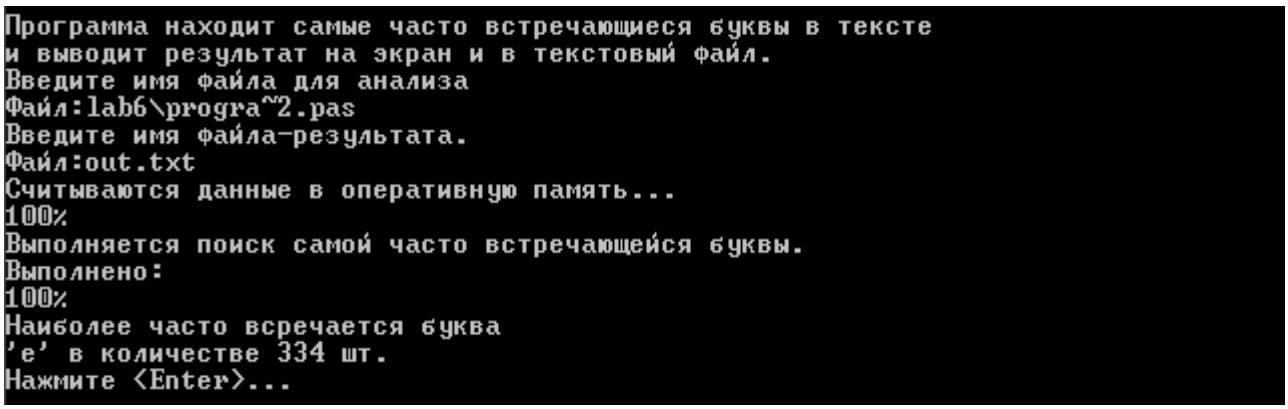

Рисунок 10 - Пример обрабоки программой собственного текста

Вывод

В ходе выполнения данной лабораторной работы я научился использовать указатели и динамические переменные, в частности, динамические массивы при разработке программ. Динамическая память позволяет быстро обрабатывать большие объёмы данных, однако требует от программиста повышенного внимания и аккуратности. К сожалению, в 16-разрядной реализации Borland Pascal 7.0 размер выделяемого сегмента динамической памяти ограничен 64 килобайтами, что создает довольно неприятные ограничения на размер входных данных.