

Комитет по образованию Администрации г. Подольска  
Муниципальное образовательное учреждение «Лицей № 26»

**«Методика решения и составления задач  
на Паскале с текстовыми типами данных»**

*Кривко-Красько Сергей Васильевич,  
учитель информатики*

## Оглавление

Введение .....	3
Текстовые файлы.....	4
Описание файловой переменной текстового типа .....	4
Сопоставление файла файловой переменной .....	5
Чтение из текстового файла .....	5
Запись информации в текстовый файл .....	6
Функции для работы с файлом.....	6
Базовые задачи на использование текстовых файлов .....	7
Структурированные типы данных в заданиях ЕГЭ (С4).....	10
Задачи для практических работ учащихся.....	13
Заключение .....	14
Список литературы .....	15

## **Введение**

Преподавание основ алгоритмизации и программирования в школьном курсе информатики и ИКТ не регламентировано достаточно строго нормативными документами (1), определяющими содержание предмета, не только на базовом, но и на профильном уровне, поэтому степень знакомства учащихся со структурированными типами данных зависит прежде всего от учителя. Учебники различных авторов предлагают различный подход к глубине изучения этой темы. Так, в учебнике Н.Д.Угриновича (2) рассматриваются только массивы и файловый тип, в учебнике И.Г.Семакина и др. (3) кроме массивов строковый и комбинированный типы, в учебнике К.Ю.Полякова и Е.А. Еремина (4) - массивы, строковый, файловый, комбинированный типы.

Задания ЕГЭ не требуют наличия знаний по использованию структурированных типов (кроме массивов и строк), но ряд таких заданий проще выполняется с использованием записей. Задания С4, хотя не предполагают ввода данных из файла, значительно проще отрабатываются с использованием текстовых файлов.

Поэтому далее в работе рассматривается методика обучения программированию с использованием текстовых файлов. Теоретический материал является основой лекции в классах физико-математического и информационно-технологического профилей, базовые задачи разбираются на уроках, также приводятся задания практических работ.

Теоретический материал подготовлен с использованием учебного пособия С.М. Окулова (5).

## Текстовые файлы

### *Описание файловой переменной текстового типа*

Текстовый файл – совокупность строк переменной длины, доступ к которым осуществляется последовательно. Информация хранится в виде символов в коде ASCII (как текст, так и числа). Чтение из файла происходит так, как происходило бы при вводе с клавиатуры, а вывод в файл – как вывод на экран. Последовательный доступ означает, что чтение следующим оператором (процедурой) Read производится с того места, где закончилось предыдущее чтение, а выведенная информация вслед за предыдущей. Можно представить, что с файлом связан невидимый указатель, перемещающийся по файлу при чтении или записи. Вернуть назад указатель нельзя (только закрыв доступ к файлу и открыв заново).

Перед использованием текстового файла в программе в разделе описания переменных должна быть обязательно описана связанная с ним переменная. Общий вид описания переменной текстового типа следующий:

**Var** <Имя\_переменной>: **text**;

где: <Имя\_переменной> - имя переменной, которая впоследствии будет связана с файлом; text – тип переменной.

Текстовые файлы состоят из строк символов. Каждая строка заканчивается символами возврата каретки **CR** (этот символ имеет в таблице ASCII код 13) и перевода строки **LF** (код 10). Вышеупомянутые символы относятся к управляющим символам, которые при просмотре и редактировании файла не выводятся на экран компьютера. Условно совокупность этих символов обозначается как EOLN (End of line).

В конце файла находится признак конца файла, который сокращенно обозначается **EOF** (End Of File). Файл такого вида обычно имеет расширение **txt**.

## ***Сопоставление файла файловой переменной***

Чтобы связать с описанной файловой переменной реальный файл, используется процедура **Assign**. Общий вид процедуры:

**Assign** (<имя\_файловой\_переменной>, <имя\_файла>);

где **Assign** – служебное слово, означающее «назначить»;

<имя\_файловой\_переменной> - имя соответствующей файлу переменной;

<имя\_файла> - полное имя текстового файла, т.е. или переменная строкового типа, содержащее имя файла в операционной системе, или же текстовая константа, содержащая это имя, заключённое в одинарные кавычки.

Примеры:

- **Assign (F, 'C:\input.txt');**

- **Readln(Fname); Assign (F, Fname);**

где **F** – переменная файлового типа, **Fname** – строковая переменная.

## ***Чтение из текстового файла***

В зависимости от того, что делать с файлом в дальнейшем, необходимо открыть его либо для чтения, либо для записи.

Если содержимое файла надо прочитать, то его открытие осуществляется процедурой:

**Reset** (<имя\_файловой\_переменной>);

Для чтения информации из файла:

**Read**(<имя\_файловой\_переменной>, список переменных);

(Аналогично, **Readln**, чтение до конца строки включая EOLN)/

Пример:

**Readln(F, str);**

После окончания операций чтения или записи файл следует закрыть с помощью процедуры **Close**(<имя\_файловой\_переменной>).

### ***Запись информации в текстовый файл***

Назначение файлу соответствующей ему файловой переменной производится так же, как и в случае чтения файла, но открытие файла для записи производится либо процедурой **Rewrite** либо процедурой **Append**.

**Rewrite** (<имя\_файловой\_переменной>);

Эта процедура при открытии файла очищает его, т.е. предыдущее содержимое файла (если таковое было) удаляется и создается новый файл.

**Append** (<имя\_файловой\_переменной>);

Используется, если нужно дописать информацию в файл (дополнить текущее содержимое текстового файла новой информацией, добавляя ее в конец файла).

Процесс записи информации в файл производится либо с помощью оператора Write, либо Writeln:

**Write**(<имя\_файловой\_переменной>, список выражений).

### ***Функции для работы с файлом.***

**EOF** (<имя файловой переменной>): Boolean - возвращает значение «истина», если достигнут конец файла (указатель файла показывает на символ конца файла);

**EOLN** (<имя файловой переменной>): Boolean - возвращает значение «истина», если достигнут конец строки (указатель файла показывает на символ конца строки).

## Базовые задачи на использование текстовых файлов

При использовании текстовых файлов в программировании можно выделить ряд задач (назову их базовыми), имеющих свои особенности:

1. Чтение строк из файла и вывод на экран.
2. Чтение чисел из файла.
3. Чтение строк из файла, содержащих числа и текст.
4. Чтение строк из файла, содержащих текст и числа.
5. Вывод в файл путем создания нового файла или же добавления в файл.

Рассмотрим типовые задачи.

**Задача 1.** Составить программу чтения файла (имя вводится с клавиатуры) и вывода на экран всех строк этого файла.

Составим программу (все программы отлажены в СП PascalABC):

```
Program File1;
Var F: text; Fname, S:string;
Begin
  Write('Введите имя файла'); Readln(Fname);
  Assign(F,Fname); Reset(F);
  while not EOF(F) do
    Begin
      Readln(F,S); WriteLn(S);
    End;
  Close(F);
End.
```

**Задача 2.** В каждой строке файла хранятся целые числа, разделенные одним или несколькими пробелами. Количество чисел в строках и количество строк неизвестно. Вычислить сумму чисел в каждой строке и вывести на экран.

```
program File2;
var F:text; Fname:string; k,s:integer;
begin
  write(('Введите имя файла')); readln(Fname);
  assign(F,Fname); reset(F);
  while not eof(F) do
    begin
      s:=0;
      while not eoln(F) do
        begin read(F,k);s:=s+k; end;
      readln(F); WriteLn('s=',s);
    end;
  close(F);
end.
```

**Задача 3.** В каждой строке текстового файла содержится следующая информация: номер школы (целое число), количество учащихся в школе, фамилия, имя, отчество директора. Числа и слова разделены пробелами. Вывести на экран информацию только о тех школах, количество учащихся в которых превышает 1000.

Файл для задачи:

```
1 1234 Петрова Анна Ивановна
2 567 Семенова Валентина Павловна
...
26 1567 Устинова Лариса Николаевна
...
```

*Комментарий.* В данной задаче нет проблемы организации ввода. Данные из строки читаются одним оператором `readln`. Условие «больше 100» позволяет избежать возможности чтения всей строки как единой символьной с дальнейшим выводом на экран.

```
program F3;
var F:text; Fname, Fdir:string; sch,k:integer;
begin
write(Введите имя файла'); readln(Fname);
assign(F,Fname); reset(F);
while not eof(F) do
begin
readln(F,sch,k,Fdir);
if k>1000 then writeln(sch:4,k:5,Fdir);
end;
close(F);
end.
```

**Задача 4.** В каждой строке текстового файла содержится следующая информация: номер школы (целое число), фамилия, имя, отчество директора и количество учащихся в школе. Между числами и словами по одному пробелу. Вывести на экран информацию только о тех школах, количество учащихся в которых превышает 1000.

Файл для задачи:

```
1 Петрова Анна Ивановна 1234
2 Семенова Валентина Павловна 567
...
26 Устинова Лариса Николаевна 1567
...
```

*Комментарий.* Перестановка последовательности данных в строке (сначала текст, затем число) приводит к значительному изменению программы, т.к. не удастся читать данные из строки одним оператором `readln`. Приходится организовывать посимвольный ввод слов до пробела.

```

program F4;
var F:text; Fname, Fdir:string; sch,k:integer; c:char;
begin
write(Введите имя файла'); readln(Fname);
assign(F,Fname); reset(F);

while not eof(F) do
begin
read(F,sch);           {*чтение номера школы*}
read(f,c);           {*чтение пробела после номера школы*}
Fname:='';
repeat
read(F,c); Fname:=Fname+c;
until c=' ';         {*чтение фамилии*}
repeat
read(F,c); Fname:=Fname+c;
until c=' ';         {*чтение имени*}
repeat
read(F,c); Fname:=Fname+c;
until c=' ';         {*чтение отчества*}
readln(F,k);         {*чтение количества учащихся*}
if k>1000 then writeln(sch:4,k:5,Fdir);
end;
close(F);
end.

```

**Задача 5** (ремейк задачи №3). В каждой строке текстового файла содержится следующая информация: номер школы (целое число), количество учащихся в школе, фамилия, имя, отчество директора. Числа и слова разделены пробелами. Вывести в другой файл информацию только о тех школах, количество учащихся в которых превышает 1000, а в исходный файл добавить строку с количеством учащихся во всех школах списка.

*Комментарий.* В задаче необходимо использовать два вида доступа к файлу для записи. Часть информации выводится в новый файл. Часть – в имеющийся. Но для этого надо закрыть доступ к нему для чтения, а затем открыть доступ для вывода добавлением.

```

program F5;
var F1,F2:text; Fname1,Fname2, Fdir:string; s, sch,k:integer;
begin
write(Введите имя исходного файла'); readln(Fname1);
assign(F1,Fname1); reset(F1);
write(Введите имя файла для вывода'); readln(Fname2);
assign(F2,Fname2); rewrite(F2);
s:=0;
while not eof(F1) do
begin
readln(F1,sch,k,Fdir); s:=s+k;
if k>1000 then writeln(F2, sch:4,k:5,Fdir);
end;
close(F1,F2);
assign(F1,Fname1); append(F1);
writeln(F1,s); close(F1);
end.

```

## Структурированные типы данных в заданиях ЕГЭ (С4)

Формально задания ЕГЭ не предполагают использования файлового типа. Но формат ввода в задач С4 таков, что отладка программ при подготовке к ЕГЭ становится очень громоздкой и трудоемкой. Поэтому, удобнее ввод данных осуществлять из текстового файла. В ряде задач С4 ЕГЭ также удобно для хранения связанной по смыслу информации использовать комбинированный тип данных (запись).

Разберем пример такой задачи:

Имеется список людей с указанием их фамилии, имени и даты рождения. Напишите эффективную по времени работы и по используемой памяти программу (укажите используемую версию языка программирования, например, Borland Pascal 7.0), которая будет определять самого старшего человека из этого списка и выводить его фамилию, имя и дату рождения, а если имеется несколько самых старших людей с одинаковой датой рождения, то определять их количество. На вход программе в первой строке подается количество людей в списке N. В каждой из последующих N строк находится информация в следующем формате:

**<Фамилия> <Имя> <Дата рождения>**

где <Фамилия> – строка, состоящая не более, чем из 20 символов без пробелов,

<Имя> – строка, состоящая не более, чем из 20 символов без пробелов,

<Дата рождения> – строка, имеющая вид ДД.ММ.ГГГГ, где ДД – двузначное число от 01 до 31, ММ – двузначное число от 01 до 12, ГГГГ – четырехзначное число от 1800 до 2100.

Пример входной строки:

Иванов Сергей 27.03.1993

Программа должна вывести фамилию, имя самого старшего человека в списке.

Пример выходных данных:

Иванов Сергей

Если таких людей, несколько, то программа должна вывести их количество. Пример вывода в этом случае:

3

*Комментарий. Задача содержит несколько самостоятельных проблем, решение каждой из которых возможно различными способами.*

- 1) Ввод данных. Можно организовать посимвольный ввод (см. задачу 4), но некоторую сложность представляет ввод трех чисел, образующих дату из-за разделителя – точки; Можно ввести целиком строку и «разобрать» ее на части. Ниже будет рассмотрен первый способ;*
- 2) Хранение данных. Можно использовать набор отдельных переменных для хранения фамилии с именем и даты (три целых числа), но эффективнее использовать комбинированный тип.*
- 3) Проблема поиска минимальной даты дополняется необходимостью подсчета количества равных минимальных дат.*
- 4) При поиске минимальной даты необходимо выполнять сравнение с довольно сложным условием. Рационально использовать функцию для выполнения этого сравнения. Но так как результат сравнения трехзначен (больше, меньше или равно), то функция должна быть не логического, а целого типа.*

Приступим к постепенной разработке программы.

Описание типов и переменных:

```
Type Date=record
    Year: 1800..2100;
    Month: 1..12;
    Day:1..31;
    End;
    Person=record
        Fam:string[15];
        Bday:date;
        End;
Var p,p_old:person; k_old:integer;c:char;
    F:text; Fname,s:string; N, I, comparison:integer;
```

Функция для сравнения двух дат:

```
Function Compare(D1,D2:date): integer;  
Begin  
If (D1.year=D2.year) and (D1.month= D2.month) and  
(D1.day=D2.day) then Compare:=0 Else  
  if ((D1.year<D2.year) or  
      (D1.year=D2.year) and (D1.month< D2.month) or  
      (D1.year=D2.year) and (D1.month= D2.month) and  
      (D1.day<D2.day) then Compare:=-1  
  Else Compare:=-1;  
End;
```

*{Функция возвращает 0, если даты совпадают, -1, если 1-ая дата предшествует 2-ой и 1, если 2-ая дата более ранняя}*

Начинаем программу. Начальные присваивания, подготовка ввода.

```
Begin  
write(Введите имя файла'); readln(Fname);  
assign(F,Fname); reset(F);  
P_old.bday.year:=2100;  
p_old.bday.month:=12;  
P_old.bday.day:=31; k_old:=0;
```

Организуем построчное чтение файла и одновременно с чтением строки поиск минимальной даты с запоминанием информации:

```
Read(F,N);  
For i:=1 to N do  
  Begin  
    with p do  
      begin  
        fam:='';  
        Repeat Read(F,c); fam:=fam+c; until c=' '  
        Repeat Read(F,c); fam:=fam+c; until c=' '  
        Readln(F,s);  
        Day :=strtoint(copy(s,1,2);  
        Month:=strtoint(copy(s,4,2);  
        Year :=strtoint(copy(s,7,4);  
      end;  
      comparison:= Compare(P.bday, p_old.bday);  
      if comparison =-1 then begin p_old:=p; k_old:=1 end;  
      else if comparison=0 then inc(k_old);  
    End;
```

*Комментарий. Фамилия и имя вводятся посимвольно, дата вводится в виде символьной строки, затем эта строка «разбирается» по частям в 3 целых числа.*

Остается организовать вывод результата и закончить программу:

```
If k_old=1 then writeln(p_old.fam) else writeln(k_old);  
Close(F);  
End.
```

## Задачи для практических работ учащихся

**Задача ПР1.** Составить программу чтения файла Computer.txt и вывода на экран тех строк этого файла, в которых есть слово «компьютер».

**Задача ПР2.** В файле EX.TXT содержится информация о результатах сдачи экзаменов студентами группы института в следующем виде:

Антонова 5 3 4 4  
Петров 2 2 4 2 и т.д.

Составить программу чтения файла EX.TXT и вывода на экран списка студентов, имеющих оценку 2 по какому-либо предмету.

**Задача ПР3.** В файле EX.TXT содержится информация о результатах сдачи ЕГЭ абитуриентами по трем предметам в следующем виде:

в первой строке N – количество абитуриентов,  
в следующих N строках:

Иванов 67 85 53  
Бобров 76 70 82  
Николаев 45 63 48 и т.д.

Составить программу чтения файла EX.TXT и вывода на экран списка студентов, набравших максимальное суммарное количество баллов.

**Задача ПР4.** В файле WEATHER.TXT содержится информация по прогнозу погоды на 28 ноября следующего виде:

Архангельск 0 без осадков  
Владивосток +8 без осадков  
Вологда 0 осадки  
Курск 0 снег  
и т.д.

Составить программу чтения файла weather.txt и вывода на экран списка тех городов, в которых ожидается температура ниже нуля, с указанием осадков.

**Задача ПР5.** Составить программу, которая читает файл Computer.txt, и в файл Comp1.txt записывает нечетные строки файла Computer.txt, а в файл Comp2.txt - четные строки этого файла, затем к файлу Comp1.txt дописывает файл Comp2.txt.

## **Заключение**

Освоение использования текстовых файлов при изучении программирования сопряжено с рядом проблем. Наибольшая сложность возникает при вводе из строки текстового файла смешанной информации (тексты, числа). Поэтому необходимо выстраивание методики преподавания таким образом, чтобы, во-первых, происходило последовательное усложнение задач и, во-вторых, были охвачены основные принципиально различные случаи.

При подготовке к ЕГЭ (задачи С4) использование ввода из текстового файла облегчает отладку программ, а использование комбинированного типа оптимизирует программу.

## **Список литературы**

1. *Программы для общеобразовательных учреждений: Информатика. 2-11 классы / Составитель М.Н.Бородин.* М. : БИНОМ. Лаборатория знаний, 2007. стр. 448.
2. **Угринович Н.Д.** *Информатика и ИКТ: Учебник для 10 класса.* М. : БИНОМ. Лаборатория знаний., 2007. стр. 371.
3. **Семакин И.Г., Хеннер Е.К., Шестакова Л.В.** *Информатика и ИКТ. Профильный уровень: учебник для 11 класса.* М. : БИНОМ. Лаборатория знаний, 2012. стр. 350.
4. **Поляков К.Ю., Еремин Е.А.** *Информатика. Углублённый уровень: учебник для 10 класса в 2 ч. Ч. 2.* М. : БИНОМ. Лаборатория знаний, 2013. стр. 304.
5. **Окулов С.М.** *Основы программирования.* М. : БИНОМ. Лаборатория знаний, 2004. стр. 424.