С.Сейфуллиннің 125 жылдығына арналған **«Сейфуллин оқулары – 15: Жастар, ғылым, технологиялар: жаңа идеялар мен перспективалар»** атты халықаралық ғылыми-теориялық конференциясының материалдары = Материалы Международной научно-теоретической конференции **«Сейфуллинские чтения – 15: Молодежь, наука, технологии – новые идеи и перспективы»**, приуроченной к 125-летию С.Сейфуллина. -2019. - Т.II, Ч 1 - С.188-190

РАБОТА С ФАЙЛАМИ НА ЯЗЫКЕ ПРОГРАММИРОВАНИЯ С++ 14 В СРЕДЕ VISUAL STUDIO 17

Турсынбаева Е.Т.

Аннотация. В статье рассматриваются особенности работы с файлами на языке программирования С++ 14 в среде Visual Studio 17.

Ключевые слова: файлы последовательного доступа, файлы произвольного доступа, библиотечные функции.

При последовательном доступе обмен информации производится через специальный буфер, резервируемый системой ввода-вывода. Компиляция языка С++ 14 рассматривает ввод-вывод как поток файлов, которые поступают последовательно, байт за байтом. Каждый поток связывается с файлом на магнитном диске или с файлом, который поставлен в соответствии физическому устройству, например клавиатуре. Связь потока с файлом устанавливается при его открытии. Открытие файла осуществляется функцией fopen. Данная функция возвращает указатель на файл. Указатель на файл необходимо объявлять, например:

FILE * lst;

Здесь FILE-имя типа, описанное в стандартном определении stdio. h;

lst-указатель на файл (логическое имя).

Обращение к функции fopen в программе производится так:

lst= fopen (физическое имя файла, вид использования файла);

Физическое имя файла может, например, быть “prn”-для устройства печати;

“b:zni.f”-для файла zni.f на дискете b:

Если файл открывается для записи или дополнения, но ещё не существует, то он создаётся. Открытие существующего файла для записи приводит к уничтожению его старого содержимого. Попытка прочитать несуществующий файл-это ошибка (fopen выдаёт указатель со значением NULL). Для работы с файлами используются библиотечные функции fprintf, fscanf, fgets, fputs. Их применение рассмотрим ниже на примерах. После окончания работы с файлом он должен быть закрыт. Это делается с помощью библиотечной функции fclose, например:

fclose (lst)

lst-указатель на файл;

Рассмотрим организацию ввода информации на печатающее устройство:

```
/*Работа с файлами (вывод на печатающее устройство )*/
#include<stdio.h>
main ( )
{   int i = 150;
FILE *lst;
/*lst-указатель на файл ( объект типа FILE)*/
lst= fopen (“prn”,”w”);
/*lst получает адрес открытого файла с именем prn, предназначенного для записи в него информации (символ w);
prn-стандартное имя устройства печати, информация из файла с адресом lst будет выводится на печатающее устройство*/
fprintf(lst,”\n число i = %d\n”,i) ;
/*первый параметр функции fprintf-это указатель на соответствующий файл; на печать будет выведена строка: число i = 150*/
fclose (lst);
/*функция fclose закрывает файл с указателем lst теперь ссылку lst можно использовать для другого файла */
```

Функция fprintf подобна функции printf и отличается от неё тем, что в качестве первого параметра использует указатель на соответствующий файл. Следующая программа показывает организацию вывода информации на дискету:

```
/* Работа с файлами (запись информации в файл на дискету)*/
#include < stdio.h >
main ( )
{ int i = 150
FILE *rsd;
Rsd=fopen(“B:ZNI.F”,”r ”);
/*второй параметр функции fopen-теперь “r ”,говорящий о чтении информации*/
fscanf(rsd,”%d”,&i);
/*из файла ZNI.F(дискета B:) будет прочитано значение i*/
printf(“число i=%d(начальное значение i)\n”,i);
while (fscanf(rsg,”%d”,&i)!=EOF)
printf(“число i = %d\n,i”);
/*последовательный вывод целых чисел из файла; вывод прекращается, когда будет достигнут конец файла(EOF)*/
fclose(rsd);
}
```

Первая функция fscanf обеспечивает чтение из файла B:ZNI.F значение целого числа i. Затем полученное значение выводится функцией printf на экран дисплея. Последующий фрагмент программы (начиная с оператора while) позволяет вывести другие целые числа из этого файла, если их там не более одного. После того как чисел в файле не остаётся, функция fscanf

выдаёт особое значение EOF-признак конца файла. Его можно использовать для прекращения чтения информации. Добавим в файл ZNI.F новые данные:

```
/*Работа  с файлами (дополнение файла на дискете)*/
#include < stdio.h >
main ( )
{ int c;
FILE * lds;
lds = fopen("B:ZNI.F","a");
/*Второй параметр функции fopen – теперь "a" , говорящий о
возможности дополнения файла*/
puts ("введите целое число");
scanf ("%d",&c);
fprintf (lds,"%d\n",c);
/*файл ZNI.F будет дополнен первым числом*/
fclose (lds);
}
```

В ответ на сообщение "введите целое число" необходимо ввести его, и файл добавится одним числом. Если необходимо добавить в файл несколько данных, можно организовать цикл. С началом работы любой программы автоматически открываются три файла, и для них определяются соответствующие указатели. Первый из них-это файл для стандартного ввода информации с клавиатуры с указателем stdin, второй – для стандартного вывода информации на экран дисплея с указателем stdout, третий – для стандартного вывода ошибок на экран дисплея с указателем stderr. Объекты stdin, stdout, stderr – константы, а не переменные, и им нельзя что-либо присваивать. Файл с указателем stderr обычно используется для хранения различных диагностических сообщений. Для выполнения произвольного доступа применяется функция fseek, позволяющая обрабатывать файл подобно массиву и непосредственно достигать любого определенного байта. Эта функция имеет вид:

fseek (stream, offset, origin);

stream – указатель на файл;

origin – указатель направления отсчета.

Функция перемещает (внутренний) указатель файла, связанного с потоком stream на новое место в файле, которое вычисляется по смещению offset и указанию направления отсчета origin.

Следующая операция с указанным потоком stream будет выполнена, начиная с той позиции, на которое произошло перемещение. Аргумент origin должен быть одной из следующих констант:

0 – начало файла;

1 – текущая позиция указателя файла;

2 – конец файла.

Для того, чтобы перемещать указатель, необходимо знать количество символов в записи. Для дальнейших рассуждений обозначим номер записи через n, а количество символов в записи – через m.

Рассмотрим ситуацию, когда origin = 0. Тогда для того, чтобы установить указатель на запись с номером n offset должен быть вычислен по следующей формуле:

offset = (n-1)*m

Рассмотрим вторую ситуацию, когда origin = 1. Установим указатель на начало файла (offset=0). Для того, чтобы перейти на следующую (вторую) запись offset должен быть равен нулю. Для перехода с первой на третью запись offset = m и т.д.

Можно двигаться не только от начала к концу файла, но и в обратном направлении. Для этого offset должен быть отрицательным. Например, если указатель стоял на пятой записи, то для того, чтоб перейти на четвертую offset = -2*m, а для перехода с пятой на третью offset = -3*m и т. д.

Рассмотрим третью ситуацию, когда origin = 2. В этом случае указатель перемещается от конца файла. Параметр offset может быть как с плюсом, так и с минусом. Например, для того, чтобы переместить указатель на предпоследнюю запись, offset должен быть вычислен по формуле: offset = 2*m

Список литературы

1. Бьярне Страуструп. Программирование. Принципы и практика с использованием C++. Второе издание. Москва, Санкт-Петербург, Киев 2016. – 1397 с.
2. Дейтел Х.М. Как программировать на C++/пер. с англ. под ред. Тимофеева В.В.- 5-е малое издание.- М.: Изд-во БИНОМ, 2015.- 800 с.
3. Brice Carnahan, James O. Wilkes. The IBM Personal Computers and the Michigan Terminal System. – UM Libraries, 1987-01-01. – 366 с.

Научный руководитель: ст.преподаватель кафедры ИКТ Казешев А.