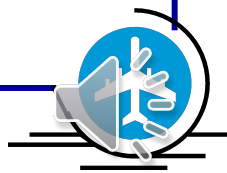
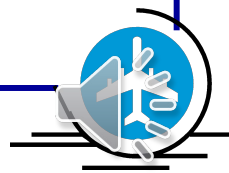
Лекция 3

Работа с файлами

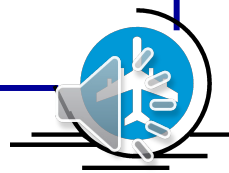


Работа с файлами

Файлы, обрабатываемые программами на языках C/C++, могут быть ***текстовыми*** и ***двоичными*** (***бинарными***). Текстовые файлы, представляют собой последовательность символьных строк. Каждый символ занимает один байт. Строка заканчивается двумя символами: «возврат каретки» (с кодом 13) и «перевод строки» (с кодом 10).

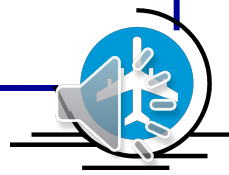


- ***Двоичные файлы*** содержат информацию во внутреннем представлении. Примером двоичного файла является exe-файл, содержащий программу в машинных командах. Прикладная программа тоже может создать двоичный файл, записав в него данные в том виде, в каком они хранятся в памяти (к примеру, типа ***int, float***).



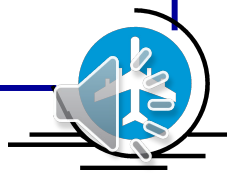
- При работе с файлами программа на языке C/C++ должна вначале открыть каждый файл с помощью функции ***fopen_s()*** (или ***fopen()***).

- При этом для каждого файла создается структура типа ***FILE,*** содержащая всю необходимую информацию о файле. Тип такой структуры с именем *FILE* определен в файле ***stdio.h***. В программе необходимо для каждого файла описать указатель на структуру типа *FILE* и передать его адрес функции *fopen_s().*



- Для **чтения информации из файла** служат функции:

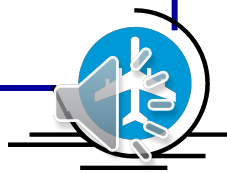
- ***fscanf_s()*** – форматированный ввод,
- ***fgets()*** – чтение одной строки,
- ***fgetc()*** – чтение одного символа,
- ***fread()*** – ввод заданного числа байтов (символов).



- Для **записи информации в файл** используются функции:

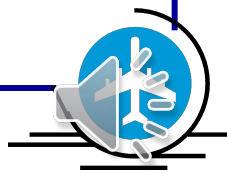
- ***fprintf()*** – форматированный вывод,
- ***fputs()*** – вывод строки,
- ***fputc()*** – вывод одного символа,
- ***fwrite()*** – вывод заданного числа байтов (символов).

- После завершения работы с файлом его следует закрыть с помощью функции ***fclose()***.



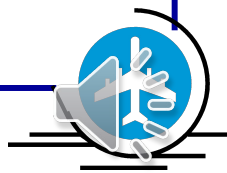
Примеры программ обработки файлов

- Задача 1. *Дан текстовый файл. Заменить все прописные латинские буквы на строчные и записать результат в другой файл.*

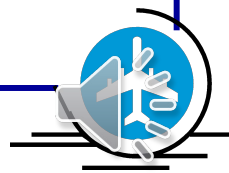


Программа:

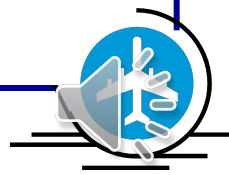
```
#include "stdafx.h"
#include <stdio.h>
#include <locale.h>
 /* Функция замены прописных латинских букв
    на строчные  */
char  * ZamenaBukv(char *str)
{
  int  i;   // индекс текущего символа строки
  for (i = 0; str[i]; i++)
   if (str[i] >= 'A' && str[i] <= 'Z')
  str[i] += 32;
  return  str;
}
```



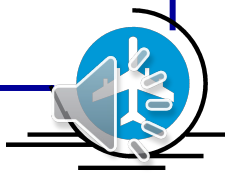
```
/* Главная функция */
int  main()
{   FILE  *f1, *f2;      // указатели на вх. и вых. файлы
    char  fname1[20], fname2[20]; // имена вх. и вых. файлов
    char  str[81];      // очередная строка файла
    setlocale(LC_ALL, "RUS");
    puts(" Введите имя входного файла");
    gets_s(fname1);
    fopen_s(&f1, fname1, "r");
    if (f1 == NULL)
    {puts("Файл не найден");
     return 1;
    }
```



```
    puts("Введите имя выходного файла");
    gets_s(fname2);
    fopen_s(&f2, fname2, "w");
    while (fgets(str, 81, f1))
    {
     fputs(ZamenaBukv(str), f2);
    }
    fclose(f1);
    fclose(f2);
    return 0;
}
```



- Задача 2. Дан текстовый файл, содержащий строки длиной до 50 символов. Число строк не более 30. Напечатать строки в алфавитном порядке.



- **Алгоритм решения задачи:**
- 1. Чтение строк файла и запоминание их в массиве.
- 2. Сортировка массива в алфавитном порядке (по возрастанию кодов символов).
- 3. Печать массива.

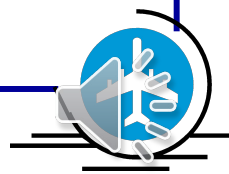
- Программа состоит из двух функций:
- **main()** – главная функция;
- **Sort()** – функция сортировки массива строк.

Программа:

```
#include "stdafx.h"        //  или      #include "pch.h"
#include <stdio.h>
#include <string.h>
#include <conio.h>
#include <locale.h>

#define  DLMAX  51      // макс. длина строки (с '\0')
#define  NMAX   30      /* макс. число строк (эл-тов
                           массива) */
 // прототип функции сортировки
void  Sort  (char  ms[NMAX][DLMAX],  int n);
```

```
/*--------------------------*/
/*    главная функция   */
/*--------------------------*/
int  main()
{ char  ms[NMAX][DLMAX]; // массив строк
   int  n;      // число элементов массива ms
   int  i;      // индекс тек. эл-та массива
   FILE *f;     // указатель на входной файл
   char  fname[13]; // имя входного файла
```

```
setlocale(LC_ALL, "Rus");
puts ("Введите имя файла");
gets_s (fname);
fopen_s (&f,  fname, "r");
if (f == NULL)
{    puts ("Файл  не найден");
     _getch();
     return 1;
}
```

```
// чтение строк файла и запись их в массив
n=0;
while (n<NMAX  &&  fgets (ms[n],DLMAX, f))
     n++;
fclose(f);
Sort (ms, n);  // сортировка массива
```

```
    puts ("Строки в алфавитном порядке:");
    for (i=0; i<n; i++)   // вывод массива
    {   printf("%s", ms[i]);
    }
    _getch();
    return 0;
}
```

```
/*-------------------------------------------------------*/
/* Функция сортировки массива строк                      */
/*          в алфавитном порядке                         */
/*-------------------------------------------------------*/
void  Sort (char ms[NMAX][DLMAX], int n)
        // Вх. данные: ms – исходный массив,
        //                n – число эл-тов массива
        // Вых. данные:   ms – сорт. массив
{   int  k,i;        //  параметры циклов
    char  s[DLMAX];     /* вспомогательная
                        переменная для перестановки
                        элементов массива  */
```

```
    for (k=n-1; k>0; k--)
        for (i=0; i<k; i++)
          if (strcmp (ms[i], ms[i+1]) > 0)
          {    // перестановка эл-тов ms[i] и ms[i+1]
              strcpy_s (s, ms[i]);
              strcpy_s (ms[i], ms[i+1]);
              strcpy_s (ms[i+1], s);
          }
}
```

Некоторые функции доступа к файлам

1. *fopen* и *fopen_s* – открытие файла.

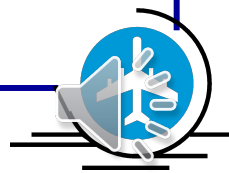
- Прототипы функций:

```
FILE * fopen (char * fname, char * mode); /* Для старых
                                  версий VS и др. систем прогр-ния */
int  fopen_s (FILE **f, char *fname, char *mode);// VS 2017
```

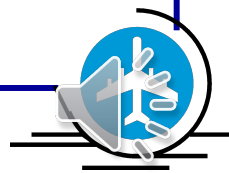
- Параметр ***fname*** задает имя открываемого файла, параметр ***mode*** – режим открытия файла и вид его обработки. Он может задаваться в виде:
- "**r**" – чтение файла,
- "**w**" – запись в файл (если файл существует, он стирается),
- "**a**" – добавление информации в конец файла,
- "**r+**" – чтение и запись.

- Функции *fopen()* и *fopen_s()* создают структуру типа *FILE* с информацией о файле; *fopen()* возвращает адрес этой структуры, а *fopen_s()* записывает адрес этой структуры в указатель, передаваемый через первый параметр. При попытке открыть несуществующий файл для чтения или чтения и записи этому указателю будет присвоена пустая ссылка *NULL*. Если открывается несуществующий файл для записи или добавления в конец файла, файл создается.

Если файл открывается для добавления информации в конец файла, то текущая позиция в файле устанавливается в конец файла. В остальных случаях устанавливается в начало файла.

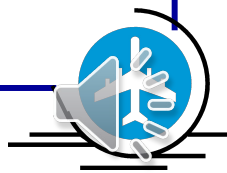


- Дополнительно в параметре ***mode*** можно указать символ ***t*** (текстовый режим) или ***b*** (двоичный режим). Например, **“rb+”** означает, что файл открывается для чтения и записи в двоичном режиме. Режимы отличаются только обработкой символов перехода на новую строку. В текстовом режиме при чтении строки комбинация символов «возврат каретки» и «перевод строки» заменяется одним символом перевода строки (при записи в файл выполняется обратное преобразование). В двоичном режиме эти преобразования не выполняются. По умолчанию файл открывается в текстовом режиме.



Пример

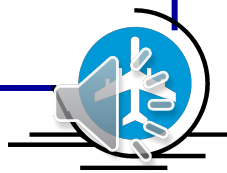
```
FILE  *f;
fopen_s (&f, "E:\\DATA\\r10.dbl", "rb+");
// f = fopen(“E:\\DATA\\r10.dbl”, “rb+”);
if  (f==NULL)
{  puts("Файл E:\\DATA\\r10.dbl  не найден");
   return 1;
}
```



2. ***fclose*** – закрытие файла.

- Функция имеет один параметр – указатель на файл.

- При закрытии файла информация из выходного буфера выгружается в файл. Поэтому следует все файлы закрывать явно.



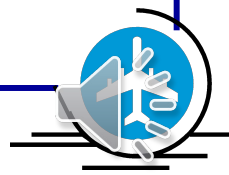
3. *fgets* – чтение строки файла.

- Прототип функции:

  char * fgets (char *s, int n, FILE *f);

- Функция считывает символы из файла, на который ссылается указатель ***f,*** в строку с указателем ***s***. Параметр ***n*** задает предельную длину считываемой строки: читается не более (n-1) символов. В конец строки добавляется нулевой байт (‘\0’).

- Функция возвращает адрес строки ***s*** или значение ***NULL*** при достижении конца файла.



Пример.

```
char  str[81];
FILE  *f;
   …
while (fgets(str,81,f))
{      /* обработка строки */
   …
}
```

4. fputs – запись строки в файл.

- Прототип функции:

  int fputs (char *s, FILE *f);

- Функция записывает строку с указателем ***s*** в файл с указателем ***f*** (без завершающего нуль-символа).

- Функция возвращает код последнего записанного символа.