Лекция №11

Файлы

Файлы

Точного определения файла не существует.

Файлом может быть:

- Объект файловой системы (файлы, папки и т.д.)
- Устройство ввода/вывода (клавиатура, принтер, дисплей)
- Служебные объекты операционной системы

Функции для работы с файлами описаны в файле stdio.h

Способы работы с файлами

Существует два способа работы с файлами:

1) Используя файлы как потоки
2) Используя произвольный доступ к данным файла

Потоковый ввод/вывод

Операции над потоковыми файлами:

- Открытие файла
- Чтение из файла (ввод из файла)
- Запись в файл (вывод в файл)
- Проверка достижения конца файла
- Закрытие файла

Открытие файла

```
FILE* fopen(char *name, char *mode);
```

`name` – имя файла

`mode` – режим открытия файла

Возвращаемое значение: указатель на файловый тип или NULL, если произошла ошибка.

mode	режим	Действие	
		Файл существует	Файл не существует
“r”	чтение	Открытие файла	Ошибка!
“w”	запись	Содержимое файла уничтожится	Файл будет создан
“a”	добавление	Запись будет происходить в конец файла	Файл будет создан

Закрытие файла

Функция закрытия файла:

```
void fclose(FILE *fp);
```

`fp` – указатель на переменную файлового типа, связанную с открытым файлом

Пример:

```
void main() {
    /* открываем файл 1.txt */
    FILE *fp = fopen("1.txt", "r");
    if (fp == NULL) {
        printf("Ошибка открытия файла!");
        return;
    }
   ... /* производим какие-то действия над файлом*/
   /* закрываем файл */
   fclose(fp);
}
```

Вывод в файл символа и строки

Запись символа в файл:

```
int fputc(int c, FILE *fp);
```

`c` – символ, который необходимо записать

`fp` – файл, в который производится запись

Возвращаемое значение: записанный символ или EOF в случае ошибки записи.

Запись строки в файл:

```
int fputs(char* s, FILE *fp)
```

`s` – строка, которую необходимо записать в файл

`fp` – файл, в который производится запись

Возвращаемое значение: неотрицательное значение или EOF в случае ошибки записи.

Вывод форматируемого текста в файл

```
int fprintf(FILE *fp, char* fmt, …);
```

`fp` – файл, в который производится запись

`fmt` – форматная строка

`...` – форматируемые значения

Примечание: fprintf идентична функции printf с той лишь разницей, что в качестве первого параметра передается файл, в который необходимо вывести значения.

Пример записи в файл

```
#include <stdio.h>
void main()
 {
    /* открываем файл на запись */
    FILE *fp = fopen("hello.txt", "w");
    /* проверяем на ошибки */
    if (fp == NULL) {
        printf("Ошибка создания файла\n");
        return;
    }
    /* записываем в файл строку */
    fprintf(fp, "Hello, file world!\n");
    /* закрываем файл */
    fclose(fp);
 }
```

Чтение из файла

Чтение символа из файла:

```
int fgetc(FILE* fp);
```

`fp` – файл

Возвращаемое значение: считанный символ или EOF если произошла ошибка чтения или достигнут конец файла.

Чтение строки из файла:

```
char* fgets(char* line, int maxline, FILE* fp);
```

`line` – строка, в которую будет считана строка из файла

`maxline` – максимальная длина строки

`fp` – файл

Возвращаемое значение: считанная строка или NULL если произошла ошибка чтения или достигнут конец файла.

Пример считывания строк из файла

Данный пример выводит на экран содержимое файла

```
#include <stdio.h>
void main()
 {
     char s[100]; /* считываемая строка */
     FILE *fp = fopen("hello.txt", "r");
    /* проверяем на ошибки */
    if (fp == NULL) {
        printf("Ошибка открытия файла\n");
        return;
    }
    /* читаем построчно файл */
    while (fgets(s, 100, fp) != NULL) {
       /* выводим считанную строку на экран */
       puts(s);
    }
    fclose(fp);
 }
```

Форматный ввод данных из файла

```
int fscanf(FILE* fp, char* fmt, …);
```

`fp` – файл

`fmt` – форматная строка

… - указатели на переменные для ввода значений

Возвращаемое значение: количество успешно введенных элементов.

Примечание: fscanf идентична функции scanf с той лишь разницей, что в качестве первого параметра передается файл, из которого необходимо считывать значения.

Стандартный ввод/вывод

В языке Си есть три преопределенных стандартных файла:
 stdin – стандартный ввод (клавиатура)
 stdout – стандартный вывод (дисплей)
 stderr – вывод сообщений об ошибках (дисплей)

puts(<строка>) аналогично fputs(<строка>, stdout)
scanf(<формат>, …) аналогично fscanf(stdin, <формат>, …)
Недопустимо:
fputs(<строка>, stdin);
fgets(<строка>, <длина>, stdout);

Дополнительные функции работы с файлами

Определение наличия ошибки в файле:

```
int ferror(FILE *fp)
```

`fp` – файл

Возвращает ненулевое значение, если при работе с данным файлом произошла ошибка.

Определение достижения конца файла:

```
int feof(FILE *fp)
```

`fp` – файл

Возвращает ненулевое значение, если достигнут конец файл.

Переопределение стандартных вводов/выводов

Переопределение через командную строку в качестве параметров:

>имя-файла – переопределение стандартного вывода на вывод в файл «имя-файла»

<имя-файла – переопределение стандартного ввода на ввод из файла «имя-файла»

>>имя-файла – переопределение вывода ошибочных сообщений на вывод в файл «имя-файла»

Пример:

Myprog.exe <input.txt >output.txt >>errors.txt