FILES

Минимум о работе с файлами. Текстовые и бинарные файлы

К. Владимиров, Intel, 2019
mail-to: konstantin.vladimirov@gmail.com

Минимум о работе с файлами

- Файл (`FILE`) это синоним для implementation defined структуры

```
typedef struct _File_t { /* нечто */ } FILE;
```

- Даже `sizeof(FILE)` не определён. Мы всегда оперируем с `FILE*`
- Открыть файл можно с помощью функции `fopen`

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

- Базовые режимы открытия: "`r`", "`w`", "`a`" для текстовых файлов означают запись, перезапись и дозапись в конец
- Закрыть файл можно с помощью функции `fclose`

```
int fclose (FILE * stream);
```

Обработка ошибок

- Посмотрим ещё раз на fopen

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

- Что если файла нет? Что если он есть, но доступ запрещён?
- Понятно, что вернётся NULL, но как определить что именно пошло не так?

Обработка ошибок

- Посмотрим ещё раз на fopen

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

- В языке C принято обратаывать ошибки в таких функциях через глобальную переменную `errno`
- Чтобы распечатать сообщение, соответствующее коду errno, используется функция perror, позволяющая частично кастомизировать сообщение

```
f = fopen("myfile", "r");

if (f == NULL)
  perror("Error opening file: ");
```

Минимум о работе с файлами

- Ввод и вывод бывают форматированными и не форматированными
- Форматированный ввод/вывод это `fprintf` и `fscanf`

```
int fprintf (FILE * stream, char const * format, ...);
int fscanf  (FILE * stream, char const * format, ...);
```

- Форматные модификаторы все как в обычных `printf` и `scanf`
- Неформатированный ввод и вывод это посимвольное считывание

```
int fputc (int character, FILE * stream);
int fputs (char const * str, FILE * stream);
int fgetc (FILE * stream);
```

- Специальные файлы: `stdin`, `stdout`, `stderr` – все тоже имеют тип `FILE*`

Problem FO – вывод файла на экран

- Создайте произвольный текстовый файл на жёстком диске
- Пример содержимого файла `myfile.txt`

```
quick brown fox jumps over lazy dog
```

- Напишите программу, которая принимает его имя как аргумент, считывает его и выводит на экран как текстовый файл

```
> gcc outtext.c -o outtext
> ./outtext myfile.txt
quick brown fox jumps over lazy dog
```

Текстовые и бинарные файлы

- Файл может быть интерпретирован как содержащий текст (и это и делается по умолчанию). В этом случае его можно использовать с функциями fprintf, fscanf и т.п.
- Но файл может быть интерпретирован как содержащий просто какие-то байты
- В этом случае мы говори о бинарных файлах и используем "rb", "wb", "ab" для их открытия
- Для работы с бинарными файлами используются функции

```
size_t fread(void *p, size_t sz, size_t cnt, FILE *f);

size_t fwrite(const void * ptr, size_t sz, size_t cnt, FILE *f);
```

Problem FO2 – вывод бинарного файла

- Создайте произвольный текстовый файл на жёстком диске
- Пример файла `myfile.in`

```
1234
```

- Напишите программу, которая выводит файл, являющийся её аргументом `argv[1]` на экран как последовательность 16-ричных чисел

```
> gcc outbin.c -o outbin
> ./outbin myfile.in
0x30 0x31 0x32 0x33
```

- Выведите саму эту программу с помощью неё же

```
> ./outbin a.out
```

Перемещение внутри файла

- Для навигации внутри файла используются

```
long int ftell (FILE *stream); // текущее положение

int fseek (FILE *stream, long int offset, int origin);
```

SEEK_SET	Начало файла
SEEK_CUR	Текущее положение в файле
SEEK_END	Конец файла

```
pFile = fopen ("example.txt", "wb"); // бинарная запись
fputs ("This is an apple", pFile);
fseek (pFile, 9, SEEK_SET);
fputs (" sam", pFile); // → This is a sample
```

Problem FS – поиск внутри файла

- Необходимо написать функцию, которая принимая в себя имя файла, начальную позицию и строчку, возвращает первую найденную позицию этой строки в файле (в символах от начала файла) или `-1` если строчка не была найдена

```
int strstrf (const char *name, int offset, const char *substr);
```

- Решая эту задачу, предположите, что файл гигантский и полностью считать его в память нельзя
- Можно воспользоваться любой из стандартных функций из `<string.h>`

Problem SF – сортировка файла

- Необходимо написать функцию, которая принимая в себя имя текстового файла и сортировала слова в нём, выводя результат в другой файл

```
int sortf(char const * in, char const * out);
```

- Например входной файл:

```
quick brown fox jumps over lazy dog
```

- Выходной файл

```
brown dog fox jumps lazy over quick
```

- Решая эту задачу, предположите, что файл гигантский и полностью считать его в память нельзя, но вы можете создавать временные файлы
- В файле все слова разделены пробелами и переносом строк