|  |  |
| --- | --- |
| Коллеги здравствуйте.  За предыдущие занятия мы с вами познакомились почти со всеми существующими в языке С типами данных, как примитивными, так и ссылочными. Довольно подробно рассмотрели работу почти всех операторов языка. Пришло время поговорить о взаимодействии программы с операционной системой, а именно - о чтении и записи в файловую систему компьютера.  Файловая система любого компьютера - это структура. Для языка С файл - это тоже структура. Структура, хранящая данные о положении курсора в файле, его название, буферы, флажки и прочие свойства. |  |
| Файлы делятся на два основных типа - текстовые и  бинарные. Мы рассмотрим работу с текстовыми.  Опишем переменную, хранящую указатель на нашу структуру. . Вся основная работа будет проходить через неё. Для того, чтобы присвоить этой переменной указатель на какой-то реальный файл воспользуемся функцией fopen, которая возвращает указатель на адрес в памяти. Функция принимает в качестве аргументов имя файла в двойных кавычках и режим его открытия.  Основных используемых режимов шесть - чтение, запись, добавление, двоичное  чтение, двоичную запись и двоичное добавление. Функции записи и добавления создают файл в случае его отсутствия. А функция записи стирает файл, если он существует и не пустой.  Итак создадим текстовый файл с каким-то неожиданным названием вроде filename.txt, и скажем нашей программе, что нужно будет его создать, если его не существует, перезаписать, если существует, а дальше мы будем в него записывать данные.  Имя файла в аргументе может быть как полным, вроде C:\FILE.TXT тогда файл будет создан в корне диска C, так и относительным, каким мы его указали сейчас. Это значит, что файл будет создан в той папке, в которой запускается наша программа. | #include<stdio.h>  int main (int argc, const char\*argv[]){  FILE \*f;  f = fopen(“filename.txt”, “w”); // r = read, w = write, a = append, rb, wb, ab  return 0;  } |
| В случае, если файл не найден или по какой-то причине не создался, в переменную file запишется нулевой указатель, поэтому перед тем, как начать работу с файлом, нужно проверить, смогла-ли программа его открыть, *для этого запишем условие если в наш указатель записался нулевой указатель, то дальнейшее выполнение функции Мэйн не имеет смысла.* | if(file == NULL) return 1; |
| Если всё хорошо, можем записывать в файл данные. Для записи в файл есть несколько функций, мы воспользуемся самой простой и очевидной  fprintf(); . В неё в качестве первого аргумента обязательно нужно передать указатель на файл, в который мы собираемся писать, а дальше можно использовать как знакомый нам printf() со всеми его удобствами, заполнителями, экранированными последовательностями и дополнительными аргументами. После того как мы закончили запись в файл его необходимо  закрыть, вызвав функцию fclose(); | fprintf(f, “Hello, files! %s”, “we did it! \n”);  fclose(f); |
| *Запустим наш проект и посмотрим что у нас получилось. Перейдем в проводник и увидим что в папке проекта появился файл filename.txt, в котором написано наше содержимое, откроем его с помощью блокнота и убедимся в этом.* |  |
| Теперь давайте рассмотрим не менее животрепещущую тему, а именно - чтение из файла. Для этого нужно его открыть в режиме чтения. Далее мы можем воспользоваться неожиданно похожей функцией - fscanf() чтобы прочитать форматированные значения из файла. *Создадим массив из переменных типа char, назовем его word и, при помощи функции fscanf() считаем из файла некоторую строку, которую положим в этот массив. Далее выведем в консоль строку которую прочитали, для этого воспользуемся привычной нам функцией printf, а затем выведем пустую строку. Запустим нашу программу и увидим, что в консоль вывелось слово Hello, - т.е. до пробела, функция fscanf отлично* отработала. | char word[256];  f = fopen(“filename.txt”, “r”);  fscanf(f, “%s”, &word);  printf(“%s”, word);  puts(“”); |
| Но сколько данных читать? Как узнать, что достигнут конец файла? Для этого придумали функцию feof() (FILE END OF FILE) возвращающую ноль, если конец файла не достигнут, и единицу если достигнут.  Опишем цикл, который выведет в консоль все значения типа строка из нашего файла. *Для этого мы циклически пройдемся по всему файлу и пока не будет достигнут конец файла будем выводить считанные строки в консоль, в этом же цикле.*  *Запустим наш проект и убедимся, что вывод в консоль полностью соответствует содержимому файла, и это было не сложно.*  *Не забудем в конце закрыть файл.* | char word[256];  f = fopen(“filename.txt”, “r”);  while(!feof(file)){  fscanf(f, “%s”, &word);  printf(“%s”, word);  }  fclose(f);  puts(“”); |
| На следующем уроке поговорим о распределении памяти. До скорой встречи! |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |