##### Лабораторная работа № 4.

ИНТЕРФЕЙС С ОПЕРАЦИОННОЙ СИСТЕМОЙ.

ФАЙЛОВЫЕ ОПЕРАЦИИ

1. Цель работы.

Ознакомиться со стандартными функциями языка СИ, используемыми для организации доступа к файлам.

2. Краткие теоретические сведения.

Для каждого из одновременно открытых файлов в программе на языке СИ должны быть описаны указатели на структуру ***FILE***, содержащую информацию о соответствующем файле. Эта структура заполняется операционной системой при открытии файла и используется затем при вызове функций, выполняющих операции чтения - записи.

Типовая последовательность действий при файловых операциях может быть следующей:

***FILE \*fp;***

***. . .***

***fp=fopen(……); // открыли файл***

***. . .***

***fprintf(fp,……); // запись в файл***

***fputs(fp,……); // аналогично***

***. . .***

***fclose(fp); // закрыли файл***

После закрытия файла указатель ***fp*** может быть использован для открытия другого файла.

Примеры использования стандартных функций, оперирующих с файлами, приведены ниже.

***FILE \*fp; int x; char buf[30], b;***

***fp=fopen(«ABC.TXT», «rt»); //открыли файл для чтения***

***// fp=NULL – ошибка***

***fscanf(fp, «%d», &x); //считали значение переменной х***

***n=fgets(buf, 30, fp); // считали строку из файла***

***// n=0 – конец файла***

***b=getc(fp); // считали символ***

***//b=EOF – конец файла***

***fprintf(fp, «x=%d», x); // записали значение переменной***

***fputs(buf, fp); // записали строку***

***putc(b, fp); // записали символ***

***fclose(fp); // закрыли файл***

При написании программ на языке СИ часто используют передачу параметров в программу непосредственно при ее запуске, например:

***name input.txt output.txt***

где ***name*** - имя исполняемого файла задачи (***name.exe***). При этом функция ***main()*** должна быть оформлена в виде:

***main(int argc,char\* argv[])***

***{***

***…***

При запуске программы с командной строкой параметр ***argc*** получает значение, равное количеству параметров в строке (включая имя задачи), а элементы массива ***argv*** – адреса строк - параметров. Например, для вышеприведенного примера

***argс = 3;***

***argv[0] = "name";***

***argv[1] = "input.txt";***

***argv[2] = "output.txt";***

3. Методические указания

В лабораторной работе требуется написать две программы для обработки текстовых файлов. Одна из них выполняет построчную, другая посимвольную обработку. Ввод параметров должен быть организован в командной строке запуска программы.

Исходный файл должен быть создан с помощью любого текстового редактора. При обработке текста рекомендуется использовать функции из стандартной библиотеки СИ для работы со строками, преобразования и анализа символов.

4. Порядок выполнения работы

4.1. Написать программу, обрабатывающую текстовый файл и записывающую обработанные данные в файл с таким же именем, но с другим типом (табл. 4.1)

4.2. Написать программу, выполняющую посимвольную обработку текстового файла (табл. 4.2.).

Ввод параметров организовать в командной строке запуска программы.

Таблица 4.1

| Вари  ант | Задание | Параметры командной строки |
| --- | --- | --- |
| 1 | Исключить строки с длиной, больше заданной | 1. Имя входного файла 2. Заданная длина строки |
| 2 | Оставить только строки, начинающиеся с латинских букв | 1. Имя входного файла 2. Количество обрабатываемых строк |
| 3 | Исключить строки, начинающиеся с заданного слова | 1. Имя входного файла 2. Заданное слово |
| 4 | Оставить строки, начинающиеся с заданной буквы | 1. Имя входного файла 2. Заданная буква |
| 5 | Исключить строки с количеством пробелов, больше заданного числа | 1. Имя входного файла 2. Заданное количество пробелов |
| 6 | Оставить строки, не содержащие цифры | 1. Имя входного файла 2. Количество обрабатываемых строк |
| 7 | Исключить строки, начинающиеся заданной парой символов | 1. Имя входного файла  2. Заданная пара символов |
| 8 | Оставить строки, заканчивающиеся цифрами | 1. Имя входного файла  2. Максимальная длина строки |
| 9 | Исключить строки, содержащие хотя бы один заданный символ | 1. Имя входного файла  2. Заданный символ |
| 10 | Оставить строки, содержащие количество цифр меньше заданного | 1. Имя входного файла  2. Заданное количество цифр |
| 11 | Исключить строки, содержащее заданное слово | 1. Имя входного файла  2. Заданное слово |
| 12 | Оставить строки, где все слова имеют длину больше указанной | 1. Имя входного файла  2. Длина слова |
| 13 | Исключить строки, начинающиеся и заканчивающиеся заданным символом | 1. Имя входного файла  2. Заданный символ |
| 14 | Оставить строки, заканчивающиеся заданным словом | 1. Имя входного файла  2. Заданное слово |
| 15 | Исключить строки, не содержащие ни одного заданного символа | 1. Имя входного файла  2. Заданный символ |
| 16 | Оставить строки, где все слова имеют длину меньше указанной | 1. Имя входного файла  2. Длина слова |
| 17 | Исключить строки, в которых есть слова короче указанной длины | 1. Имя входного файла  2. Длина слова |
| 18 | Оставить строки, в которых указанное слово встречается более одного раза | 1. Имя входного файла  2. Заданное слово |
| 19 | Исключить строки, в которых есть хотя бы один не алфавитно-цифровой символ | 1. Имя входного файла  2. Количество обрабатываемых строк |
| 20 | Оставить строки с количеством слов, меньшим указанного | 1. Имя входного файла  2. Количество слов |

Таблица 4.2

| Вариант | Задание | Параметры командной строки |
| --- | --- | --- |
| 1 | Удалить из текста заданный символ | 1. Имя входного файла 2. Заданный символ |
| 2 | В конце каждой строки вставить заданный символ | 1. Имя входного файла 2. Заданный символ |
| 3 | Заменить цифры на пробелы | 1. Имя входного файла 2. Количество замен |
| 4 | Заменить знаки на заданный символ | 1. Имя входного файла 2. Заданный символ |
| 5 | Заменить каждый пробел на два | 1. Имя входного файла 2. Количество замен |
| 6 | После каждой точки вставить символ ‘\n’ | 1. Имя входного файла 2. Количество замен |
| 7 | Удалить из текста все пробелы | 1. Имя входного файла  2. Количество замен |
| 8 | Заменить заданные символы на пробелы | 1.Имя входного файла  2. Заданный символ |
| 9 | После каждого пробела вставить точку | 1.Имя входного файла  2. Количество вставок |
| 10 | Заменить все пробелы последним символом текста | 1. Имя входного файла  2. Максимальное количество замен |
| 11 | Во всех парах одинаковых символов второй символ заменить на пробел | 1. Имя входного файла  2. Количество замен |
| 12 | Заменить на пробелы все символы, совпадающие с первым символом в строке | 1. Имя входного файла  2. Количество замен |
| 13 | Заменить заданную пару букв на символы #@ | 1. Имя входного файла  2. Заданная пара букв |
| 14 | Заменить все цифры заданным символом | 1. Имя входного файла  2. Заданный символ |
| 15 | Заменить на пробел все символы, совпадающие с последним символом в строке | 1. Имя входного файла  2. Количество замен |
| 16 | Заменить все символы с кодами меньше 48 на пробелы | 1. Имя входного файла  2. Количество замен |
| 17 | Заменить все символы с кодами больше 48 на пробелы | 1. Имя входного файла  2. Количество замен |
| 18 | Заменить каждый третий символ на пробел | 1. Имя входного файла  2. Количество замен |
| 19 | Заменить все пробелы на заданный символ | 1. Имя входного файла  2. Заданный символ |
| 20 | Заменить все пары одинаковых символов на пробелы | 1. Имя входного файла  2. Количество замен |

5. Содержание отчета

5.1. Титульный лист

5.2. Цель работы.

5.3. Задание.

5.4. Текст(ы) программ.

5.5. Полученные результаты.

5.6. Выводы.

6. Контрольные вопросы

6.1. Назначение указателя ***FILE\****.

6.2. Стандартные функции открытия/закрытия файлов.

6.3. Стандартные функции для построчной обработки файлов.

6.4. Стандартные функции для посимвольной обработки файлов.

6.5. Стандартные функции работы со строками.

6.6. Буферизованный и небуферизованный ввод/вывод.

6.7. Передача параметров в программу при ее запуске.

6.8. Проверка корректности выполнения операций файлового ввода-вывода.

6.9. Режимы открытия файлов.

6.10. Перенаправление стандартного ввода/вывода.

6.11. Чем определяется область видимости и время жизни переменной?