ПРАКТИЧЕСКАЯ РАБОТА №4

Организация файлового ввода/выода.

Цель работы: изучить работу потоков файлового ввода/вывода, понятие дескриптора файла и различные режимы работы с файловым вводом/выводом в Си.

Задание: для ранее созданной программы разработать операции файлового ввода вывода, позволяющие сохранять созданные ранее объекты в файл и загружать их из файла. Также нужно разработать функции получение количества сохраненных объектов в файле и перемещение по файлу и извлечение i-того объекта из файла. Так же разработать поддержу сохранения и загрузки из бинарного файла.

Изменить демонстрационную программу в соответствии с добавленным функционалом. Использовать передачу аргументов через командную строку, запланировать 4 функции:

./main **save** *filename* – где filename (строка) название файла, реализует сохранение в файл сгенерированные объекты.

./main **load** *filename* – где filename (строка) название файла, реализует загрузку из файла сгенерированных объектов.

./main **list** *filename* – где filename (строка) название файла, реализует чтение количества структур хранимых в файле, выводит список прочитанных структур и нумерацию.

./main **get** *I filename* – где I (целое число) i-тый элемент сохраненный в файл, filename (строка)название файла. Выводит содержимое i-того элемента из файла filename.

Предусмотреть выбор режима чтения и записи. (обычный/бинарный). *Ключевой особенностью демонстрации должно быть то, что за раз можно продемонстрировать только одну операцию. То есть после выполнения одной из 4 функций программа полностью закрывается.*

Реализовать все функции отдельным модулем (fio**struct**.h, fio**struct**.c), где struct название структуры по вашему варианту.

Для сборки демонстрационной программы использовать ***makefile.***

Задание на 3:

1. Добавить функцию ***rand_gen_struct(size_t n)***, для генерации 10,000 элементов для своей структуры. Например: 10 000 элементов *point3d*
2. Реализовать сохранение данных в текстовый файл в обычном режиме.
3. Реализовать загрузку данных из текстового файла в обычном режиме.
4. Формат записи данных:
5. Реализовать функцию ***get_element_from_text_file*** (const char* filename, int index): перемещаться по файлу, читая строки до нужного индекса. Вернуть значение или NULL, если индекс выходит за границы файла.

Задание на 4:

Выполнить задание на 3.

1. Реализовать сохранение данных в текстовый файл в двоичном режиме.
2. Реализовать загрузку данных из текстового файла в двоичном режиме. Учесть размер данных при записи/чтении (например, использовать ***sizeof*** для определения размера каждого элемента).

3. Реализовать функцию ***get_element_from_binary_file***(const char* filename, int index, size_t element_size, void* result), вычислить позицию элемента как index * element_size. Использовать ***fseek*** для перемещения курсора к этой позиции. Прочитать элемент с помощью ***fread***.

Задание на 5:

Выполнить задание на 3 и 4.

1. Разработать функцию ***rand_gen_struct_in_container(size_t n),*** которая генерирует n значений вашей структуры и помещает в вашу структуру контейнер. Например: 10 000 элементов *point3d* внутри структуры *dict*.
   - Для *list*/*queue*: каждый элемент на новой строке.
   - Для *dict*: каждая пара ключ-значение записывается через разделитель (например, key:value).
   - Для *vector*: элементы записываются через запятую или пробел.
2. Реализовать сохранение и загрузку данных в текстовом и двоичном режиме Позволяющий сразу перемещать данные их структуру в файл для записи, а также при загрузке из файла в структуру.
3. Продемонстрировать вывод i-того элемента из структуры, которая загружена из файла.