

Программирование

Факультет безопасности информационных технологий
Университет ИТМО

Осень 2023 - Весна 2024

Лабораторная работа № 4

Работа с файлами и структуры данных на основе указателей

Разработать на языке C для ОС Linux программу, которая читает из файла заданного формата состояние заданной структуры данных, выполняет манипуляции со структурой в памяти, считывая команды из стандартного потока ввода, и сохраняет результат в файл.

Программа должна представлять собой консольное приложение, настройка работы которого осуществляется путем передачи аргументов в строке запуска:

```
lab4abcNXXXXX [-v] имя_файла
```

Имя программы должно начинаться на lab4, далее должен следовать уникальный для варианта суффикс. Уникальный суффикс составляется из первых букв имени, отчества (если есть) и фамилии студента, выполняющего лабораторную работу. Далее следует номер группы студента. Используются строчные латинские буквы и арабские (в традиционном понимании, т. е. 0..9) цифры. Например, если студента, выполняющего лабораторную, зовут Петр Сергеевич Иванов, его группа — N32451, то имя программы должно быть lab4psiN32451.

Программа должна поддерживать опцию -v, при указании которой следует вывести ФИО и группу студента, который выполнил работу, и информацию о варианте задания, после чего завершиться:

```
$ ./lab4psiN32451 -v
Петр Сергеевич Иванов, гр. N32451
Вариант: 2-3-4-5
```

Номер варианта имеет вид W-X-Y-Z, где W — номер варианта из Табл. 1, X — номер варианта из Табл. 2, Y — номер варианта из Табл. 3, Z — номер варианта из Табл. 4.

Обязательный аргумент имя_файла задает путь к файлу, в котором содержится список строк в формате, заданном вариантом из Табл. 4. Если файла с указанным именем нет, список считается пустым. После запуска программа считывает команды, позволяющие изменять список, из стандартного потока ввода. Например (наклонный шрифт означает ввод с клавиатуры, `↵` — нажатие на Enter, `␣` — комбинация клавиш Ctrl-D (окончание ввода)):

```
$ ./lab4psiN32451 test/CATS
pop_front↵
push_front "All your base are belong to us."↵
push_back "Move\"ZIG\"." "For great justice."↵
sort↵
dump overmind.txt↵
␣
```

Команды могут быть добавлены в файл, содержимое которого затем может быть передано на стандартный ввод программе, в этом случае все должно работать аналогично вводу команд из консоли:

```
$ cat commands.txt
pop_front
push_back "Что за ужас тут творится?!"
```

```
sort
```

```
$ cat commands.txt | ./lab4psiN32451 test/doka2
```

```
$ ./lab4psiN32451 test/doka2 < commands.txt
```

В данном примере две последние команды по-разному передают содержимое файла `commands.txt` на стандартный ввод программы, но функционально одинаковы (делают одно и то же).

Если пользователь при запуске указывает неподдерживаемые опции, лишние аргументы, аргументы неправильного формата, вместо чисел нужного формата строки, которые не являются числами или не могут поместиться в заданный тип, либо совершает какие-то другие ошибки, программа должна сообщить об этом, например:

```
$ ./lab4psiN32451
```

```
Ошибка: не задано имя файла.
```

```
$ ./lab4psiN32451 /no/such/path.txt
```

```
Ошибка: не удалось создать файл.
```

```
$ ./lab4psiN32451 test/bad.boy
```

```
Ошибка: некорректный формат файла.
```

```
$ ./lab4psiN32451 mr.bean
```

```
magic↵
```

```
Ошибка: неподдерживаемая команда: 'magic'.
```

```
$ ./lab4psiN32451 mr.bean < commands.txt
```

```
Ошибка: неподдерживаемая команда: 'magic'.
```

Если программа завершилась успешно, то код завершения (значение, которое возвращается из `main()` или передается в функцию `exit()`) должен быть нулевым. В случае возникновения ошибок во время выполнения программы, код завершения должен быть ненулевым. Проверить код завершения последней запущенной в терминале команды можно с помощью специальной переменной `$?`. Конкретные значения кодов ошибок выбираются разработчиком. Например:

```
$ ./lab4psiN32451 test/ok_file < commands.txt
```

```
$ echo $?
```

```
0
```

```
$ ./lab4psiN32451 test/corrupted_file < commands.txt
```

```
Ошибка: некорректный формат файла.
```

```
$ echo $?
```

```
1
```

Программа должна прочитать содержимое файла, если файл существует, и создать в памяти список, тип которого определяется вариантом из Табл. 1. Если файла с указанным именем не существует или размер файла равен нулю, в памяти создается "пустой" список (не содержащий элементов). После выполнения команд программа должна сохранить измененный список в файл в формате, заданном вариантом из Табл. 4. **При завершении (когда не удалось прочитать очередную команду из потока ввода, например, потому, что пользователь нажал Ctrl-D) программа должна сохранить результат в файл, если список подвергся каким-либо модификациям.** Если в результате выполнения команд получен "пустой" список, программа должна создать файл нулевого размера (если файл существовал, то его размер нужно урезать до нуля). Программа должна поддерживать обязательные для всех вариантов команды и одну команду, определяемую вариантом из Табл. 3.

Обязательные для всех вариантов команды:

`push_front строка1 [строка2 ... строкаN]`

Добавить строки в начало списка в порядке, обратном перечислению (строка1 будет начальным элементом списка, строка2 вторым и т.д.). Строки отделяются друг от друга пробелами. В случае, если очередная строка не подходит по формату, определенному вариантом из Табл. 2, в стандартный поток ошибок должно быть выведено сообщение об ошибке. Корректные строки, указанные в команде, должны быть добавлены в список.

`push_back строка1 [строка2 ... строкаN]`

Добавить строки в конец списка в порядке перечисления. Строки отделяются друг от друга пробелами. В случае, если очередная строка не подходит по формату, определенному вариантом из Табл. 2, в стандартный поток ошибок должно быть выведено сообщение об ошибке. Корректные строки, указанные в команде, должны быть добавлены в список.

`pop_front`

Удалить первый элемент списка. Если список пустой, команда ничего не делает.

`pop_back`

Удалить последний элемент списка. Если список пустой, команда ничего не делает.

`dump [имя_файла]`

Вывод содержимого списка в стандартный поток вывода или в файл, если указано имя_файла, в следующем формате: "адрес_элемента связи_элементов строка". Связи элементов — данные, зависящие от типа списка (адрес следующего элемента для односвязного списка, адреса следующего и предыдущего элементов для двусвязного списка, результат XOR'a адресов предыдущего и следующего элементов для XOR-связного списка). Пример вывода для двусвязного списка из трех элементов, тип строк — IPv4 адреса:

```
0x58050000f740288a 0x0000000000000000 0x58050000f7402896 192.168.8.1
0x58050000f740289c 0x58050000f740288a 0x58050000f7402928 1.1.1.1
0x58050000f7402928 0x58050000f740289c 0x0000000000000000 93.240.0.3
```

Проект (исходные коды, заголовочные файлы, Makefile и прочие файлы, которые могут понадобиться для сборки) должен содержаться в отдельном каталоге с именем, совпадающим с названием программы (lab4abcNXXXXX), и собираться с помощью стандартной утилиты make. Исходные файлы программы на языке C должны компилироваться с помощью gcc. Makefile должен поддерживать как минимум цели all и clean. Пример заготовки проекта ЛР № 4 содержится на [гугл-диске](#) в папке "лабораторные" (архив lab4abcNXXXXX.tar.gz, для распаковки можно использовать команду `tar -xzvf lab4abcNXXXXX.tar.gz`)

Порядок выполнения и сдачи лабораторной работы:

1. Скачать заготовку проекта, изменить название каталога на правильное (соответствующее вашей группе и ФИО), скорректировать содержимое Makefile'a.
2. Выполнить задание, подготовить все файлы проекта, скомпилировать программу с флагами `-Wall -Wextra -Werror` и устранить все предупреждения и ошибки.
3. Протестировать программу на различных входных данных, убедиться, что ошибок нет, в противном случае вернуться к пункту 2.
4. Удалить все исполняемые и промежуточные файлы из папки проекта (`make clean`). В архиве должны остаться только файлы *.c, *.h, Makefile, README.txt.
5. Заархивировать папку проекта, используя формат *.tar.gz. (`tar -czvf lab4abcNXXXXX.tar.gz lab4abcNXXXXX/`).
6. Подготовить отчет по лабораторной работе в формате pdf, на титульной странице отчета не забыть поставить подпись. Файл отчета должен иметь название NXXXXX_ФамилияИО_ЛР4.pdf. Состав отчета описан ниже.
7. Отправить архив и отчет в формате pdf на почту преподавателя, который ведет лабораторные, письмом с темой «Программирование ЛР4 Фамилия Имя Отчество NXXXXX Вариант A-B-C-D».

8. Дождаться ответа по почте или на лабораторном занятии, устранить возможные замечания (повторить с пункта 1).
9. Получить некоторое количество вопросов от преподавателя по отчету и темам, связанным с лабораторной, и дать на них ответы (а может и не получить, если лабораторная выполнена на хорошем уровне и сомнений в знаниях студента у преподавателя не возникает). Получить от преподавателя подтверждение, что работа выполнена успешно и отчет принят.
10. Немного отдохнуть и приступить к выполнению следующей лабораторной :-)

Отчет должен быть подготовлен в формате pdf и содержать:

- правильно оформленную титульную страницу (с подписью студента);
- задание;
- Make-файл;
- примеры работы программ на различных исходных данных (скриншоты);
- исходный текст программы с комментариями.

Замечание 1. При выполнении лабораторной работы следует использовать только функции стандартной библиотеки C и системные вызовы операционной системы. Использовать C++, ввод-вывод в стиле C++ (классы `ifstream/ofstream/...`), контейнеры и алгоритмы STL (`<string>`, `<vector>`, `<map>`, ...) и сторонние библиотеки **запрещено**.

Замечание 2. В программе должна присутствовать обработка ошибок: в случаях, если пользователь передал некорректные аргументы или ввел недопустимые значения, программа должна выдавать диагностическое сообщение на консоль (в стандартный поток ошибок), прежде чем завершиться.

Замечание 3. Индексы в списках и структурах в файлах начинаются с нуля.

Замечание 4. Программа должна успешно компилироваться и выполняться в 64-разрядной ОС Linux с ядром версии ≥ 5.0 , glibc версии ≥ 2.0 , gcc версии ≥ 10.0 .

Таблица 1. Тип списка

№ варианта	Тип списка
1	Односвязный список
2	Двусвязный список
3	XOR-связный список

Таблица 2. Формат данных в строке

№ варианта	Формат данных в строке
1	IPv4-адрес
2	IPv6-адрес
3	MAC-адрес
4	Доменное имя
5	Адрес электронной почты
6	Строка в двойных кавычках (с экранированием двойных кавычек внутри)
7	Строка в одинарных кавычках (с экранированием одинарных кавычек внутри)
8	Дата и время
9	ISBN-13
10	Автомобильный номер РФ



Таблица 3. Команда

№ варианта	Команда
1	Команда: <code>shuffle</code> Описание: Переупорядочить элементы списка случайным образом. Каждое

	применение команды должно приводить к новому случайному порядку.
2	<i>Команда:</i> filter строка <i>Описание:</i> Удалить из списка элементы, в которых встречается строка.
3	<i>Команда:</i> delete N <i>Описание:</i> Удалить из списка элемент с индексом N.
4	<i>Команда:</i> delete_odd <i>Описание:</i> Удалить из списка элементы на нечетных позициях.
5	<i>Команда:</i> delete_even <i>Описание:</i> Удалить из списка элементы на четных позициях.
6	<i>Команда:</i> del_start строка <i>Описание:</i> Удалить из списка первый слева элемент, содержащий заданную строку.
7	<i>Команда:</i> del_end строка <i>Описание:</i> Удалить из списка первый справа элемент, содержащий заданную строку.
8	<i>Команда:</i> reorder_asc <i>Описание:</i> Расположить элементы списка по возрастанию.
9	<i>Команда:</i> reorder_dsc <i>Описание:</i> Расположить элементы списка по убыванию.
10	<i>Команда:</i> min_first <i>Описание:</i> Переместить элемент с минимальным значением (или все элементы с минимальным значением, если их в списке несколько) в начало списка, не изменяя порядок других элементов.
11	<i>Команда:</i> max_first <i>Описание:</i> Переместить элемент с максимальным значением (или все элементы с максимальным значением, если их в списке несколько) в начало списка, не изменяя порядок других элементов.
12	<i>Команда:</i> min_last <i>Описание:</i> Переместить элемент с минимальным значением (или все элементы с минимальным значением, если их в списке несколько) в конец списка, не изменяя порядок других элементов.
13	<i>Команда:</i> max_last <i>Описание:</i> Переместить элемент с максимальным значением (или все элементы с максимальным значением, если их в списке несколько) в конец списка, не изменяя порядок других элементов.
14	<i>Команда:</i> rot_left N <i>Описание:</i> Выполнить циклический сдвиг списка влево на N элементов.
15	<i>Команда:</i> rot_right N <i>Описание:</i> Выполнить циклический сдвиг списка вправо на N элементов.
16	<i>Команда:</i> unique <i>Описание:</i> Удалить из списка одинаковые элементы, в результирующем списке все элементы должны быть уникальными.
17	<i>Команда:</i> swap K L <i>Описание:</i> Обменять местами элементы списка с индексами K и L.
18	<i>Команда:</i> reverse <i>Описание:</i> Изменить порядок следования элементов списка на обратный.
19	<i>Команда:</i> insert N S <i>Описание:</i> Вставить в список на позицию N строку S.
20	<i>Команда:</i> ins_before S1 S2 <i>Описание:</i> Если в списке встречается элемент S1, вставить перед ним элемент S2.
21	<i>Команда:</i> ins_after S1 S2 <i>Описание:</i> Если в списке встречается элемент S1, вставить после него элемент S2.
22	<i>Команда:</i> rot_odd_right N <i>Описание:</i> Выполнить циклический сдвиг элементов списка, находящихся на нечетных позициях, вправо на N элементов.

23	<p>Команда: rot_even_right N</p> <p>Описание: Выполнить циклический сдвиг элементов списка, находящихся на четных позициях, вправо на N элементов.</p>
24	<p>Команда: rot_odd_left N</p> <p>Описание: Выполнить циклический сдвиг элементов списка, находящихся на нечетных позициях, влево на N элементов.</p>
25	<p>Команда: rot_even_left N</p> <p>Описание: Выполнить циклический сдвиг элементов списка, находящихся на четных позициях, влево на N элементов.</p>
26	<p>Команда: shuffle_odd</p> <p>Описание: Переупорядочить элементы списка на четных позициях случайным образом. Каждое применение команды должно приводить к новому случайному порядку.</p>
27	<p>Команда: shuffle_even</p> <p>Описание: Переупорядочить элементы списка на нечетных позициях случайным образом. Каждое применение команды должно приводить к новому случайному порядку.</p>

Таблица 4. Формат файла

№ варианта	Формат файла
1	<p>Начало файла Конец файла</p>  <p>Содержимое файла:</p> <p>Поле cnt (целое без знака [4 байта]). Количество строк в области строк.</p> <p>Область строк s_0, \dots, s_{N-1}. Строки в кодировке UTF-8 располагаются одна за другой без промежутков. Каждая строка заканчивается нулевым байтом.</p> <p>Область произвольных данных data. Область произвольного размера (возможно, нулевого).</p> <p>Область индексов i_0, \dots, i_{N-1}. Индексы соответствующих строк s_0, \dots, s_{N-1}, определяющие порядок строк в списке (N целых без знака [2 байта]). Область индексов представляет собой массив, расположенный в конце файла.</p>
2	<p>Начало файла Конец файла</p>  <p>Содержимое файла:</p> <p>Поле offset (целое без знака [4 байта]). Смещение области индексов в байтах от начала файла.</p> <p>Область строк s_0, \dots, s_{N-1}. Строки в кодировке UTF-8 располагаются одна за другой без промежутков. Каждая строка заканчивается нулевым байтом.</p> <p>Область произвольных данных data. Область произвольного размера (возможно, нулевого).</p> <p>Область индексов i_0, \dots, i_{N-1}. Индексы соответствующих строк s_0, \dots, s_{N-1}, определяющие порядок строк в списке (N целых без знака [2 байта]). Область индексов представляет собой массив, расположенный в конце файла. Количество индексов определяет количество строк в файле.</p>

3	<div data-bbox="316 62 1444 241"> <div>Начало файла</div> <div>Конец файла</div> <div> <div>n_0</div> <div>s_0</div> <div>n_1</div> <div>s_1</div> <div>...</div> <div>n_{N-1}</div> <div>s_{N-1}</div> </div> </div> <p><i>Содержимое файла:</i> В файле последовательно располагаются размеры строк и сами строки. Порядок следования строк в файле соответствует порядку следования строк в списке. Поля размеров строк n_0, \dots, n_{N-1}(целые без знака [4 байта]). Размеры строк в байтах. Строки s_0, \dots, s_{N-1}. Строки в кодировке UTF-8 без нулевого байта в конце.</p>
4	<div data-bbox="316 488 1444 645"> <div>Начало файла</div> <div>Конец файла</div> <div> <div>cnt</div> <div>i_0</div> <div>o_0</div> <div>i_1</div> <div>o_1</div> <div>...</div> <div>i_{N-1}</div> <div>o_{N-1}</div> <div>data</div> <div>s_0</div> <div>...</div> <div>data</div> <div>s_{N-1}</div> </div> </div> <p><i>Содержимое файла:</i> Поле cnt (целое без знака [4 байта]). Количество пар полей индексов и смещений строк, расположенных сразу после этого поля. Индексы i_0, \dots, i_{N-1} (целые без знака [2 байта]). Индексы соответствующих строк s_0, \dots, s_{N-1}, определяющие порядок строк в списке. Смещения o_0, \dots, o_{N-1} (целые без знака [4 байта]). Смещения соответствующих строк в файле. Каждый элемент массива o_i содержит смещение в байтах от начала файла строки s_i. Строки s_0, \dots, s_{N-1}. Строки в кодировке UTF-8 с нулевым символом в конце располагаются в файле с возможными промежутками между ними. Области произвольных данных data. Области произвольного размера (возможно, нулевого).</p>
5	<div data-bbox="316 1093 1444 1249"> <div>Начало файла</div> <div>Конец файла</div> <div> <div>cnt</div> <div>i_0</div> <div>o_0</div> <div>...</div> <div>i_{N-1}</div> <div>o_{N-1}</div> <div>data</div> <div>n_0</div> <div>s_0</div> <div>...</div> <div>data</div> <div>n_{N-1}</div> <div>s_{N-1}</div> </div> </div> <p><i>Содержимое файла:</i> Поле cnt (целое без знака [4 байта]). Количество пар полей индексов и смещений строк, расположенных сразу после этого поля. Индексы i_0, \dots, i_{N-1} (целые без знака [2 байта]). Индексы соответствующих строк s_0, \dots, s_{N-1}, определяющие порядок строк в списке. Смещения o_0, \dots, o_{N-1} (целые без знака [4 байта]). Смещения полей длин соответствующих строк в файле. Каждый элемент массива o_i содержит смещение в байтах от начала файла поля n_i, за которым следует строка s_i. Размеры строк n_0, \dots, n_{N-1} (целые без знака [4 байта]). Размеры строк в байтах. Строки s_0, \dots, s_{N-1}. Строки в кодировке UTF-8 без нулевого символа в конце. Размер каждой строки в байтах задается полем n_i. Области произвольных данных data. Области произвольного размера (возможно, нулевого).</p>
6	<div data-bbox="316 1731 1444 1888"> <div>Начало файла</div> <div>Конец файла</div> <div> <div>i_0</div> <div>n_0</div> <div>s_0</div> <div>i_1</div> <div>n_1</div> <div>s_1</div> <div>...</div> <div>i_{N-1}</div> <div>n_{N-1}</div> <div>s_{N-1}</div> </div> </div> <p><i>Содержимое файла:</i> В файле последовательно располагаются группы из трех полей: индекс строки, размер строки и данные строки. Порядок строк в списке определяется индексами. Индексы i_0, \dots, i_{N-1} (целые без знака [2 байта]). Индексы соответствующих строк s_0, \dots, s_{N-1}, определяющие порядок строк в списке. Размеры строк n_0, \dots, n_{N-1} (целые без знака [2 байта]). Размеры строк в байтах. Строки s_0, \dots, s_{N-1}. Строки в кодировке UTF-8 без нулевого байта в конце.</p>

7	<div data-bbox="316 56 1444 224"><div>Начало файла</div><div><div>S₀</div><div>S₁</div><div>...</div><div>S_{N-1}</div><div>data</div><div>i₀</div><div>O₀</div><div>...</div><div>i_{N-1}</div><div>O_{N-1}</div><div>offset</div></div><div>Конец файла</div></div> <p>Содержимое файла:</p> <p>Область строк s_0, \dots, s_{N-1}. Область, в которой располагаются строки в кодировке UTF-8. Каждая строка заканчивается нулевым байтом. Начало строки s_i определяется смещением o_i, окончание строки — нулевым байтом.</p> <p>Области произвольных данных data. Области произвольного размера (возможно, нулевого).</p> <p>Индексы i_0, \dots, i_{N-1} (целые без знака [4 байта]). Индексы соответствующих строк s_0, \dots, s_{N-1}, определяющие порядок строк в списке.</p> <p>Смещения o_0, \dots, o_{N-1} (целые без знака [4 байта]). Смещения соответствующих строк в файле. Каждое поле o_i содержит смещение в байтах от начала файла строки s_i.</p> <p>Смещение первого индекса offset (целое без знака [4 байта]). Последнее поле в файле содержит смещение первого индекса i_0.</p>
8	<div data-bbox="316 719 1444 873"><div>Начало файла</div><div><div>O₀</div><div>...</div><div>O_{N-1}</div><div>NUL</div><div>data</div><div>S₀</div><div>data</div><div>...</div><div>S_{N-1}</div><div>data</div><div>i₀</div><div>...</div><div>i_{N-1}</div></div><div>Конец файла</div></div> <p>Содержимое файла:</p> <p>Область смещений o_0, \dots, o_{N-1}. Массив смещений строк в файле (N целых без знака [4 байта]). Каждый элемент массива o_i (кроме последнего) содержит смещение в байтах от начала файла строки s_i. Последний элемент массива NUL представляет собой маркер конца массива, все байты маркера — нулевые.</p> <p>Области произвольных данных data. Области произвольного размера (возможно, нулевого).</p> <p>Область строк s_0, \dots, s_{N-1}. Область, в которой в произвольном порядке располагаются строки в кодировке UTF-8, заканчивающиеся нулевым байтом. После каждой строки может располагаться область произвольных данных. Начало строки s_i определяется смещением o_i, окончание строки — нулевым байтом.</p> <p>Область индексов i_0, \dots, i_{N-1}. Индексы соответствующих строк s_0, \dots, s_{N-1}, определяющие порядок строк в списке (N целых без знака [2 байта]). Область индексов представляет собой массив, расположенный в конце файла.</p>