

Программирование

Факультет безопасности информационных технологий
Университет ИТМО

Осень 2024

Лабораторная работа № 3 Строки и ввод-вывод

Разработать на языке C для ОС Linux программу, которая читает данные из стандартного потока ввода или файла, осуществляет преобразование в соответствии с вариантом лабораторной работы и выводит результат в стандартный поток вывода или файл.

Номер варианта лабораторной работы составлен из четырех чисел и имеет вид A-B-C-D, где A — номер варианта из Табл. 1, B — номер варианта из Табл.2., C — номер варианта из Табл. 3, D — номер варианта из Табл. 4.

Программа должна представлять собой консольное приложение, настройка работы которого осуществляется путем передачи аргументов в строке запуска (квадратные скобки показывают, что соответствующие аргументы могут отсутствовать):

```
prg3abcNXXXXX [опции] [имя_вход_файла [имя_выход_файла]]
```

Имя программы должно начинаться на prg3, 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 — N31451, то имя программы должно быть prg3psiN31451.

Программа получает на вход текст, в котором находит последовательности символов, заданные вариантом из Табл. 1, и, в зависимости от опций, указанных пользователем при запуске, модифицирует текст и выводит результат либо в стандартный поток вывода, либо в указанный файл. Модификация текста заключается либо в выделении найденных фрагментов с помощью форматирования, принятого форматом текстовых файлов Markdown [1], либо выделением найденных фрагментов цветом с помощью управляющих последовательностей ANSI [2, 3], если при запуске программы пользователь указал опцию -c. Форматирование Markdown определяется вариантом из Табл. 3, а цвет — вариантом из Табл. 4.

Программа может быть запущена без опций и аргументов. В этом случае все прочитанные из стандартного потока ввода строки после преобразования передаются на стандартный поток вывода. Например (наклонный шрифт означает ввод с клавиатуры, ↵ — нажатие на Enter, ⌘ — комбинацию клавиш Ctrl-D (окончание ввода)):

```
$ ./prg3psiN31451
CATS: "All your base are belong to us."↵
CATS: "You have no chance to survive make your time."↵
Captain: "Move\"ZIG\"."↵
Captain: "For great justice."↵
CATS: **"All your base are belong to us."**
CATS: **"You have no chance to survive make your time."**
Captain: **"Move\"ZIG\"."**
Captain: **"For great justice."**
```

В данном примере преобразование заключалось в поиске в тексте строк, заключенных в двойные кавычки, и обрамлении их разметкой **, что в формате Markdown выделяет эти

строки жирным шрифтом. Строки, в которых заданная последовательность символов не найдена, передаются на вывод без изменений.

Если программа запускается с одним аргументом имя_вход_файла, задающим путь к входному файлу, то строки читаются из него, а вывод выполняется в стандартный поток вывода:

```
$ ./prg3psiN31451 zero_wing.txt
CATS: **"All your base are belong to us."**
CATS: **"You have no chance to survive make your time."**
Captain: **"Move\"ZIG\"."**
Captain: **"For great justice."**
```

Если программа запускается с двумя аргументами — имя_вход_файла и имя_выход_файла — второй аргумент задает имя файла, в который записывается результат преобразований:

```
$ ./prg3psiN31451 zero_wing.txt zero_wing.md
```

Программа должна поддерживать опции -v, -c, -n и опции, определяемые вариантом из Табл. 2.

При указании опции -v программа должна вывести ФИО и группу студента, который выполнил работу, и информацию о варианте задания, после чего завершиться:

```
$ ./prg3psiN31451 -v
Петр Сергеевич Иванов, гр. N31451
Вариант: 3-2-2-2
```

При указании опции -c программа должна вместо выделения найденных фрагментов с помощью разметки Markdown использовать изменение цвета вывода с помощью управляющих последовательностей ANSI. Цвет определяется вариантом из Табл. 4. Например (выполняется поиск MAC-адресов, выделение выполняется зеленым цветом, начиная со второго):

```
$ ./prg3psiN31451 -c -f=2 to_comrade_major.log
11:44:20 21.09.2023 92.163.10.3 (04:44:F3:3D:21:89) accessed
https://kremlin.ru/
11:45:11 21.09.2023 92.163.1.19 (F3:22:91:01:00:A1) accessed
https://en.wikipedia.org/
11:45:12 21.09.2023 91.102.2.200 (30:12:72:39:D2:FE) accessed
https://bitcoincasino.go/ [BLOCKED]
11:45:12 21.09.2023 92.163.10.3 (04:44:F3:3D:21:89) accessed
https://en.cppreference.com/
11:45:15 21.09.2023 91.102.2.200 (30:12:72:39:D2:FE) accessed
https://bighotbuffers.xxx/ [BLOCKED]
```

Последовательность символов, которую требуется найти, может располагаться на нескольких строках. При указании опции -n выполняется поиск и выделение только таких последовательностей, которые расположены в пределах одной строки. Например (наклонный шрифт означает ввод с клавиатуры, ↵ — нажатие на Enter, ⌘ — комбинацию клавиш Ctrl-D (окончание ввода), выполняется поиск строк в одинарных кавычках, преобразование — жирное начертание):

```
$ ./prg3psiN31451
Мощность множества можно рассматривать и как↵
новый объект, называемый 'кардинальным↵
числом' или 'кардиналом'.↵
⌘
Мощность множества можно рассматривать и как
```

новый объект, называемый `**'кардинальным
числом' **` или `**'кардиналом' **`.

```
$ ./prg3psiN31451 -n
```

Мощность множества можно рассматривать и как
новый объект, называемый `'кардинальным
числом'` или `'кардиналом'`.

Мощность множества можно рассматривать и как
новый объект, называемый `'кардинальным
числом'` или `**'кардиналом' **`.

Обратите внимание, что программа должна корректно работать с перенаправлением ввода-вывода, которое выполняется из командного интерпретатора (bash, sh, ...):

```
$ cat zero_wing.txt | ./prg3psiN31451 > /tmp/zero_wing.md
```

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, напечатав сообщение об ошибке в стандартный поток ошибок (stderr).

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 оболочки `$?`. Конкретные значения кодов ошибок выбираются разработчиком. Например:

```
$ ./prg3psiN31451 input_file.txt output_file.md
```

```
$ echo $?  
0
```

```
$ ./prg3psiN31451 -x in.txt out.txt  
Ошибка: неподдерживаемая опция -x.
```

```
$ echo $?  
1
```

```
$ ./prg3psiN31451 -b=?? in.txt out.txt  
Ошибка: ?? не является числом.
```

```
$ echo $?  
1
```

```
$ ./prg3psiN31451 -c /home/peter/prg3/file.md  
Ошибка: не удалось открыть файл /home/peter/prg3/file.md
```

```
$ echo $?  
2
```

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 (prg3abcNXXXXX), 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 № 3 содержится на [гугл-диске](#) в папке "лабораторные"

(архив prg3abcNXXXXXX.tar.gz, для распаковки можно использовать команду tar -xzvf prg3abcNXXXXXX.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 prg3abcNXXXXXX.tar.gz prg3abcNXXXXXX/).
6. Подготовить отчет по лабораторной работе в **формате pdf**, на титульной странице отчета не забыть **поставить подпись** и **указать номер варианта**. Файл отчета должен иметь название ПРГ_ЛР3_NXXXXXX_ФамилияИО.pdf. Состав отчета описан ниже.
7. Отправить архив и отчет в формате pdf на почту преподавателя, который ведет лабораторные, письмом с темой «*Программирование ЛР3 Фамилия Имя Отчество NXXXXXX 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. На этот семестр хватит лабораторных! :-)

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. Программа не должна делать никаких предположений о возможной максимальной длине строк в файле и должна работать с файлами, содержащими строки любой длины.

Замечание 4. Если для облегчения понимания того, как выполняется преобразование, нужно выводить дополнительную информацию, её желательно выводить с помощью переменной окружения LAB3DEBUG.

Замечание 5. Программа должна успешно компилироваться и выполняться в 64-разрядной ОС Linux с ядром версии >= 5.0, glibc версии >= 2.0, gcc версии >= 10.0.

Ссылки:

[1] <https://ru.wikipedia.org/wiki/Markdown>

[2] https://ru.wikipedia.org/wiki/Управляющие_последовательности_ANSI

[3] https://en.wikipedia.org/wiki/ANSI_escape_code

Таблица 1. Данные для поиска

№ варианта	Данные для поиска	Замечания и примеры
1	IPv4-адрес	Десятичный формат с точками. <i>Примеры:</i> 192.168.0.1, 8.8.8.8
2	IPv6-адрес	Полная или сокращенная запись без IPv4-компоненты. <i>Примеры:</i> 2001:0db8:85a3:0000:0000:8a2e:0370:7334, fe80::1ff:fe23:4567:890a
3	MAC-адрес	Разделение октетов через двоеточие. <i>Примеры:</i> fe:57:e6:b8:c5:97, 00:00:00:00:00:00
4	Номер кредитной карты	Вида XXXX-XXXX-XXXX-XXXX с проверкой по алгоритму Луна. <i>Примеры:</i> 4282-4207-4879-9773, 2362-3178-9095-1698
5	Адрес электронной почты	Вида имя@домен. <i>Примеры:</i> pupkin@itmo.ru, rocket+groot@avengers.com
6	Строка в двойных кавычках	С экранированием двойных кавычек внутри с помощью символа \. <i>Примеры:</i> "Здравствуй, Мир!", "Vasya \"Mad dog\" Pupkin"
7	Строка в одинарных кавычках	С экранированием одинарных кавычек внутри с помощью символа \. <i>Примеры:</i> 'Что за ужас тут творится?!', 'The Wizard\'s Last Rhymes'
8	Дата и время	В формате HH:MM:SS dd.mm.yyyy <i>Примеры:</i> 12:24:01 01.02.2001, 00:00:00 31.12.2023
9	ISBN-13	С проверкой по алгоритму ISBN-13. <i>Примеры:</i> 978-0-306-40615-7, 978-2-266-11156-0
10	Автомобильный номер РФ	Можно ограничиться только поддержкой формата xDDDxxDDRUS из ГОСТ Р 50577-2018. <i>Примеры:</i> o335но78RUS, c227на47RUS
11	Федеральный телефонный номер РФ	Вида +7 8{({DEF})}XXX{-}XX{-}XX, где означает выбор одного из вариантов, {} - элемент может отсутствовать. <i>Примеры:</i> +7(921)3368715, 89013368584, 8(971)201-11-91.

Таблица 2. Опции, с помощью которых задается ограничение преобразования

№ варианта	Опции, ограничивающие преобразование
1	<code>-b=M</code> Опция <code>-b</code> задает номер начальной строки (строки нумеруются с единицы). Преобразование выполняется, начиная со строки M. Если опция <code>-b</code> не указана, то преобразование начинается с первой строки. <code>-e=N</code> Опция <code>-e</code> задает номер конечной строки. Преобразование выполняется до строки N включительно. Если опция <code>-e</code> не указана, преобразование продолжается до последней строки.
2	<code>-f=M</code> Опция <code>-f</code> задает номер объекта данных, начиная с которого нужно осуществлять заданное вариантом преобразование. Объекты данных нумеруются с единицы. Если опция <code>-f</code> не указана, то преобразование выполняется, начиная с первого найденного объекта. <code>-t=N</code> Опция <code>-t</code> задает номер последнего объекта данных, над которым нужно осуществлять заданное вариантом преобразование. Если опция <code>-t</code> не указана, преобразование выполняется до последнего найденного объекта данных. При указании опции <code>-n</code> нумеруются только те объекты данных, которые отвечают критериям поиска, то есть располагаются на одной строке.

Таблица 3. Разметка Markdown, используемая для выделения найденных фрагментов текста

№ варианта	Разметка	Код разметки
1	Курсив	<code>*Пример*</code>
2	Жирное начертание	<code>**Пример**</code>
3	Код	<code>`Пример`</code>
4	Зачеркнутый текст	<code>~~Пример~~</code>

Таблица 4. Цвет для выделения найденных фрагментов текста при выводе с опцией `-c`

№ варианта	Цвет	Код цвета (foreground color ANSI code)
1	Красный (red)	31
2	Зеленый (green)	32
3	Желтый (yellow)	33
4	Синий (blue)	34
5	Пурпурный (magenta)	35
6	Голубой (cyan)	36