

Отчёт по лабораторной работе №10

Текстовый редактор Emacs

Матвеева Анастасия Сергеевна

Содержание

1	Цель работы	4
2	Задачи лабораторной работы	5
3	Выполнение лабораторной работы	6
4	Выводы	23
5	Ответы на контрольные вопросы	24

List of Figures

3.1	Открываем редактор	6
3.2	Создание файла	6
3.3	Набираем текст	7
3.4	Вырезаем строку	8
3.5	Вставляем строку	8
3.6	Выделяем текст	9
3.7	Вставляем текст	10
3.8	Выделяем область	11
3.9	Вырезаем данную область	12
3.10	отменяем последнее действие	12
3.11	Перемещаем курсор в начало строки	13
3.12	Перемещаем курсор в начало строки	14
3.13	Перемещаем курсор в конец строки	15
3.14	Перемещаем курсор в начало буфера	15
3.15	Перемещаем курсор в конец буфера	16
3.16	Список активных буферов	16
3.17	Переключаемся на другой буфер	17
3.18	Закрываем окно	17
3.19	Разделяем фрейм на 4 части	18
3.20	Вводим текст	18
3.21	Ищем слова	19
3.22	Ищем слова	19
3.23	Переключаемся между результатами поиска	20
3.24	Выходим из режима поиска	20
3.25	Заменяем слово в тексте	21
3.26	Заменяем слово в тексте	21
3.27	Заменяем слово в тексте	21

1 Цель работы

Познакомиться с операционной системой Linux. Получить практические навыки работы с редактором Emacs.

2 Задачи лабораторной работы

Задачи:

1. Ознакомиться с операционной системой Linux.
2. Ознакомиться с редактором Emacs.
3. Изучить основные команды редактора Emacs.
4. В ходе работы использовать эти команды и интерпретировать их вывод.

3 Выполнение лабораторной работы

1. Откроем редактор Emacs с помощью команды «emacs&» (рис. 3.1).

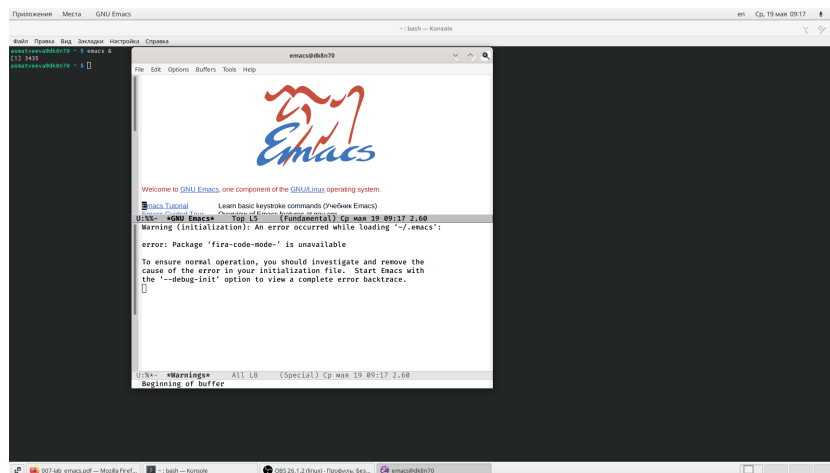


Figure 3.1: Открываем редактор

2. Создадим файл lab07.sh с помощью комбинации «Ctrl-x»«Ctrl-f» (рис. 3.2).

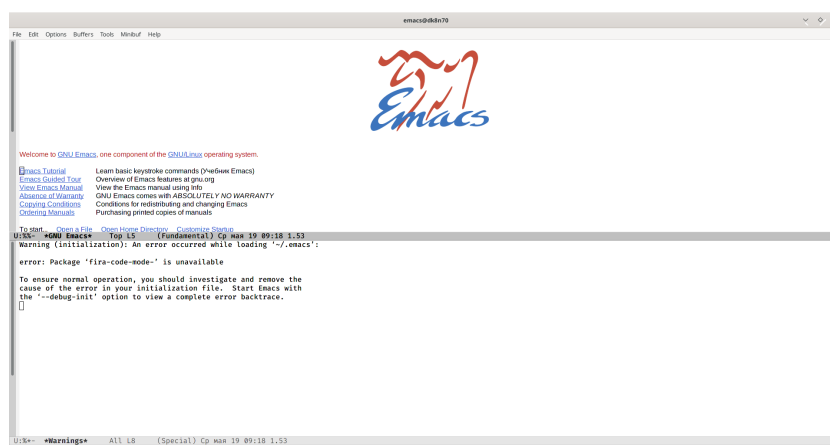
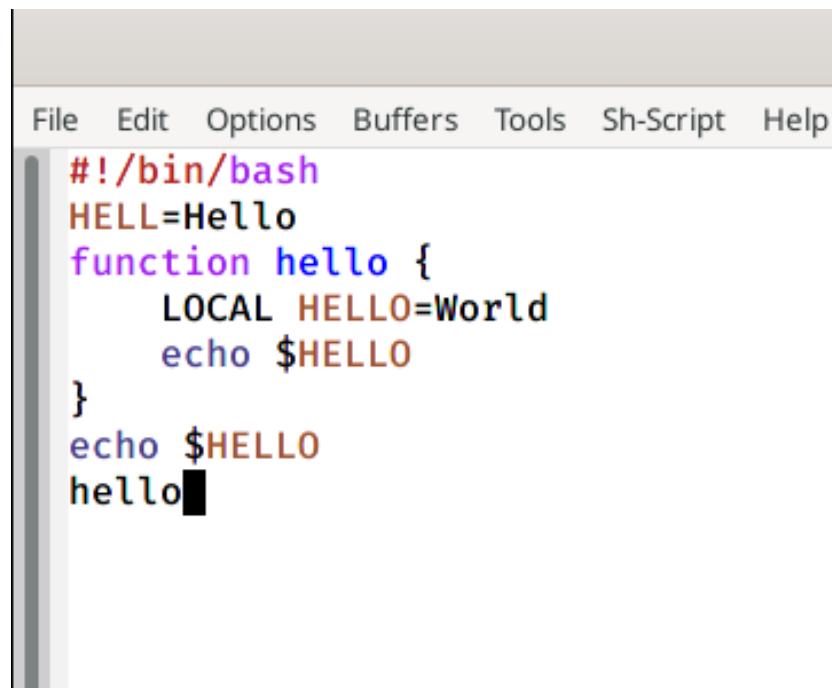


Figure 3.2: Создание файла

3. В открывшемся буфере наберем необходимый текст (рис. 3.3).

A screenshot of a text editor window. The window has a menu bar with the following items: File, Edit, Options, Buffers, Tools, Sh-Script, and Help. The text area contains a shell script with the following content:

```
#!/bin/bash
HELL=Hello
function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
```

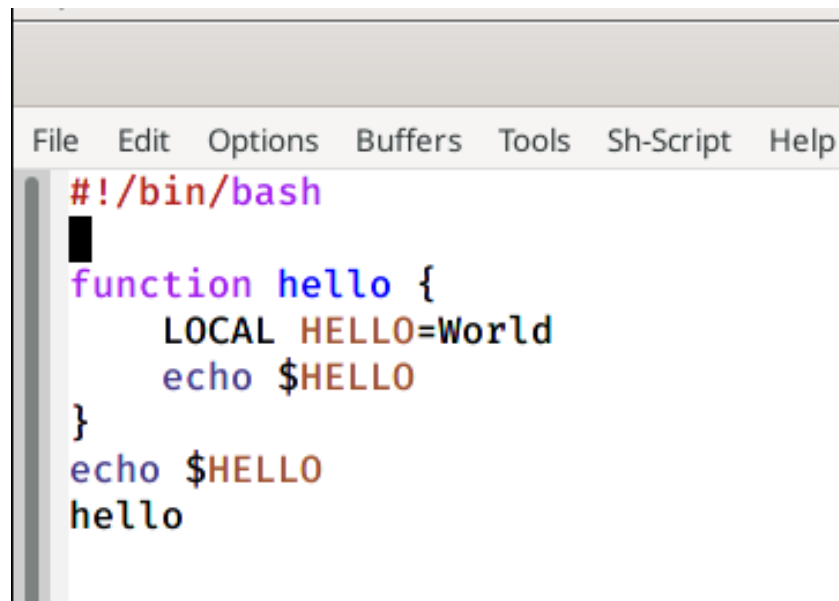
The text is color-coded: the shebang is purple, variable assignments are brown, function definitions are purple, and function calls are brown. A black cursor is positioned at the end of the last line, "hello".

Figure 3.3: Набираем текст

4. Сохраним файл с помощью комбинации «Ctrl-x»«Ctrl-s».

5.

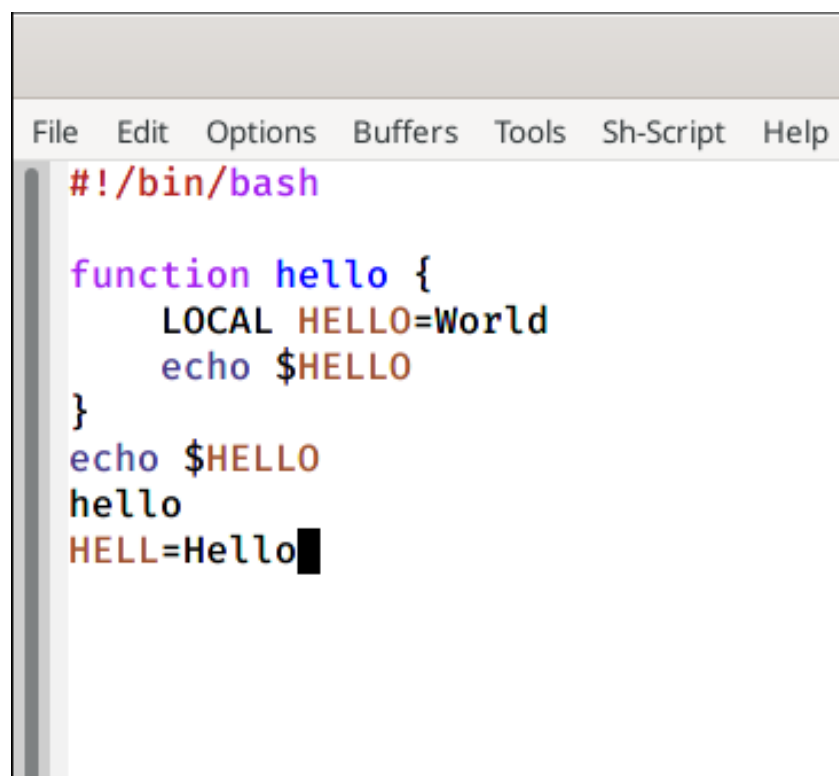
1) Вырежем одной командой целую строку («Ctrl-k») (рис. 3.4).



```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
```

Figure 3.4: Вырезаем строку

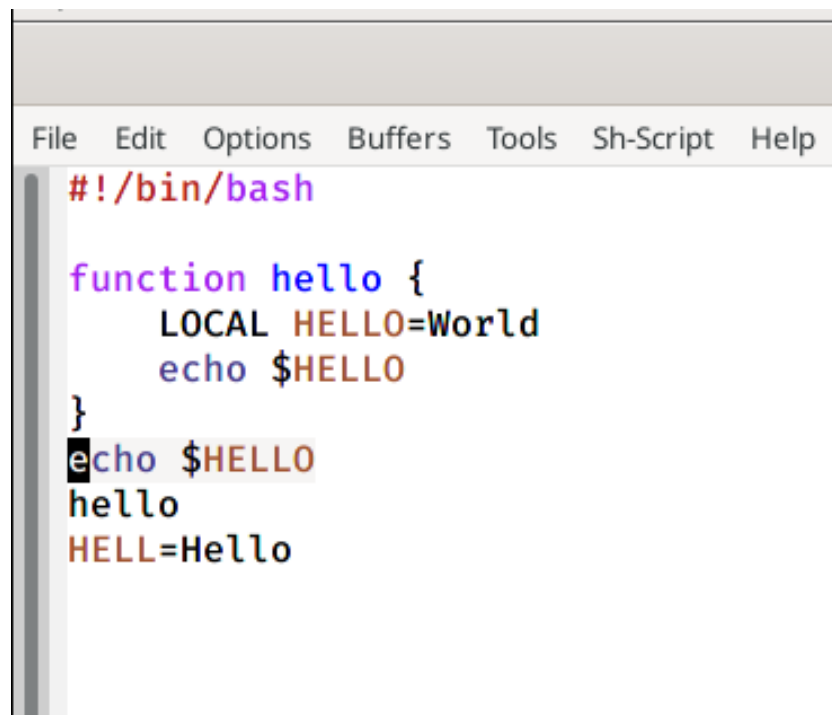
2) Вставим эту строку в конец файла («Ctrl-y») (рис. 3.5).



```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
HELL=Hello
```

Figure 3.5: Вставляем строку

3) Выделим область текста («Ctrl-space») (рис. 3.6).



The image shows a terminal window with a menu bar at the top containing 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The terminal content is a shell script with the following lines: `#!/bin/bash`, `function hello {`, `LOCAL HELLO=World`, `echo $HELLO`, `}`, `echo $HELLO`, `hello`, and `HELL=Hello`. A black rectangular selection highlight is positioned over the text `echo $HELLO` on the line following the closing brace of the function.

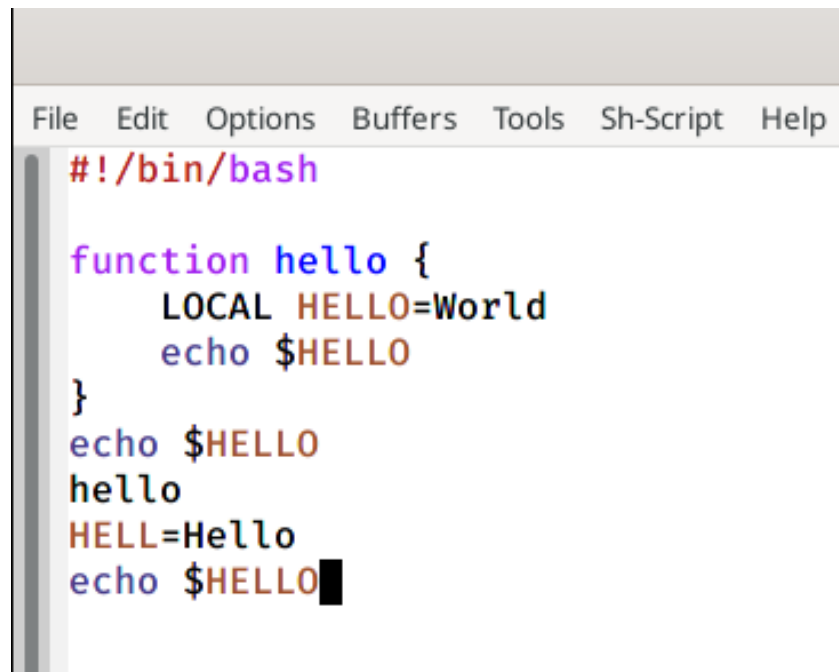
```
#!/bin/bash

function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
HELL=Hello
```

Figure 3.6: Выделяем текст

4) Скопируем область в буфер обмена («Alt-w»).

5) Вставим область в конец файла («Ctrl-y») (рис. 3.7).

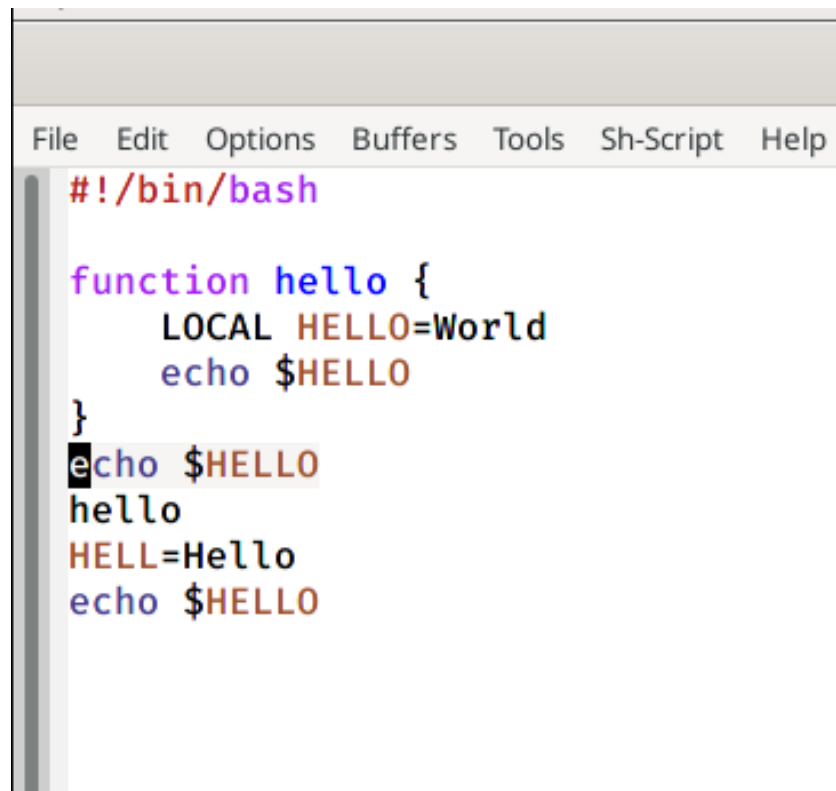


```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash

function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
HELL=Hello
echo $HELLO
```

Figure 3.7: Вставляем текст

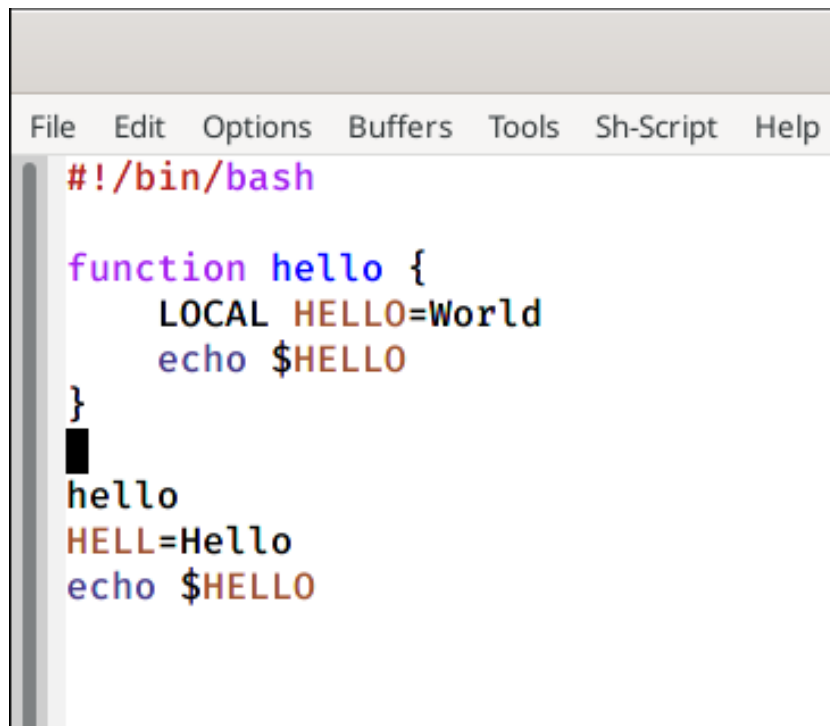
- 6) Вновь выделим эту область («Ctrl-space») и на этот раз вырежем её («Ctrl-w») (рис. 3.8, 3.9).



```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash

function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
HELL=Hello
echo $HELLO
```

Figure 3.8: Выделяем область

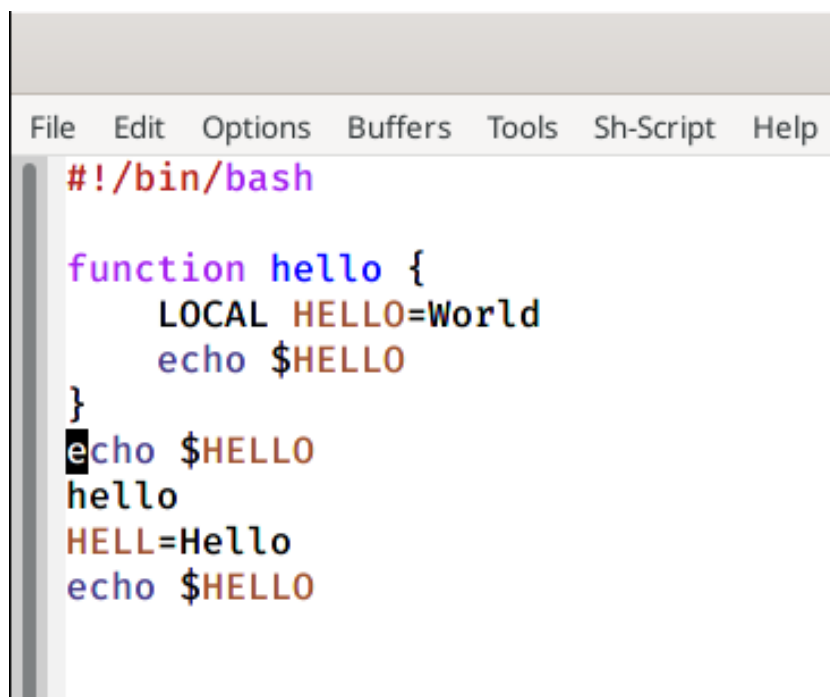


```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash

function hello {
    LOCAL HELLO=World
    echo $HELLO
}
hello
HELL=Hello
echo $HELLO
```

Figure 3.9: Вырезаем данную область

7) Отменим последнее действие («Ctrl-/») (рис. 3.10).



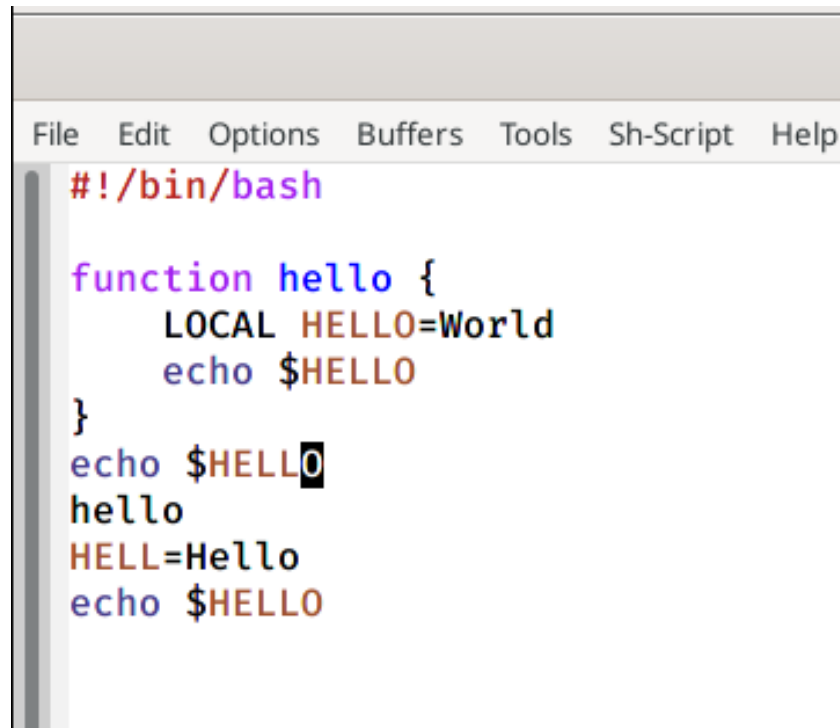
```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash

function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
HELL=Hello
echo $HELLO
```

Figure 3.10: отменяем последнее действие

6.

1) Переместим курсор в начало строки («Ctrl-a») (рис. 3.11, 3.12).

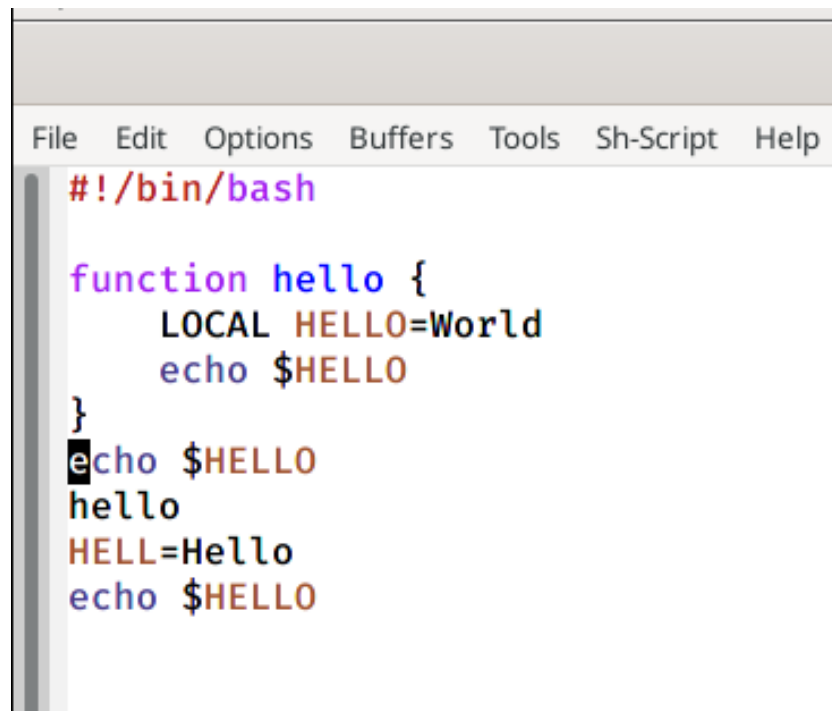


The image shows a terminal window with a menu bar at the top containing 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The terminal content is a shell script with the following lines: `#!/bin/bash`, `function hello {`, `LOCAL HELLO=World`, `echo $HELLO`, `}`, `echo $HELLO`, `hello`, `HELL=Hello`, and `echo $HELLO`. The cursor is positioned at the end of the `echo $HELLO` line, just before the `hello` command on the next line.

```
#!/bin/bash

function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
HELL=Hello
echo $HELLO
```

Figure 3.11: Перемещаем курсор в начало строки



The image shows a terminal window with a menu bar containing 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The terminal content is as follows:

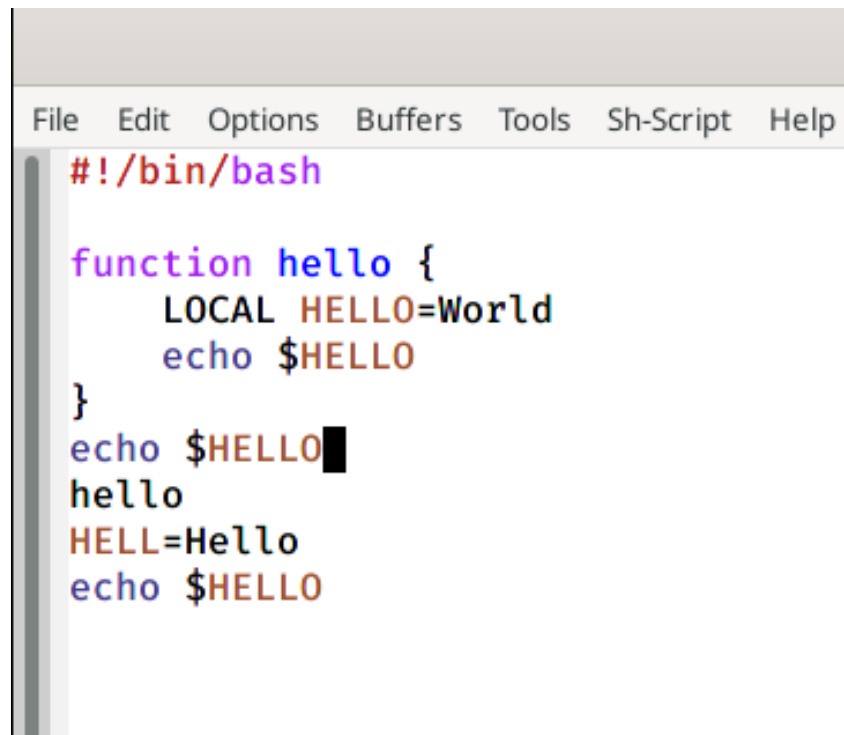
```
#!/bin/bash

function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
HELL=Hello
echo $HELLO
```

A black cursor is positioned at the beginning of the line containing 'echo \$HELLO'.

Figure 3.12: Перемещаем курсор в начало строки

2) Переместим курсор в конец строки («Ctrl-e») (рис. 3.13).



A screenshot of a terminal window with a menu bar containing 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The terminal content is as follows:

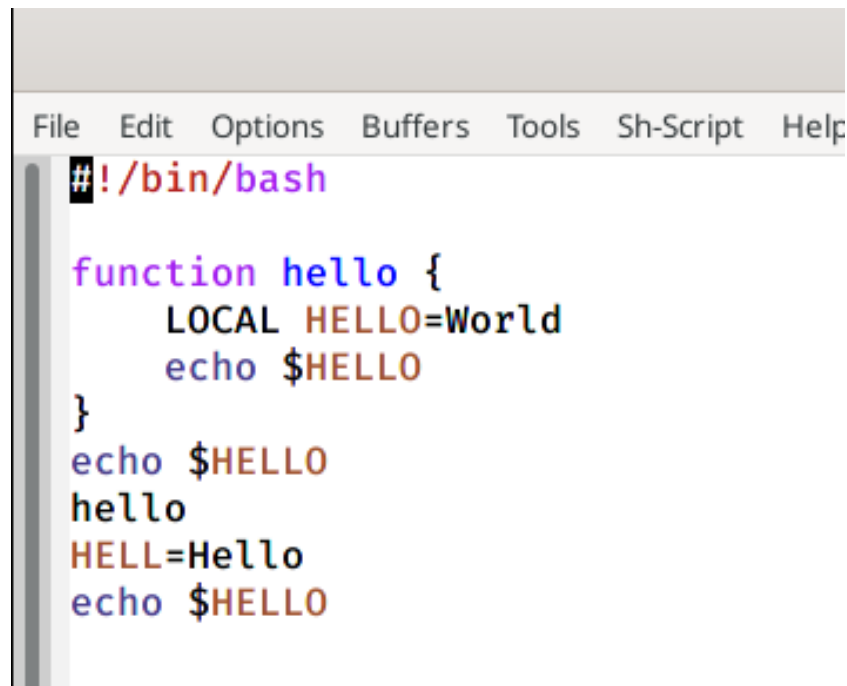
```
#!/bin/bash

function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
HELL=Hello
echo $HELLO
```

The cursor is positioned at the end of the line 'echo \$HELLO'.

Figure 3.13: Перемещаем курсор в конец строки

3) Переместим курсор в начало буфера («M-<») (рис. 3.14).



A screenshot of a terminal window with a menu bar containing 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The terminal content is as follows:

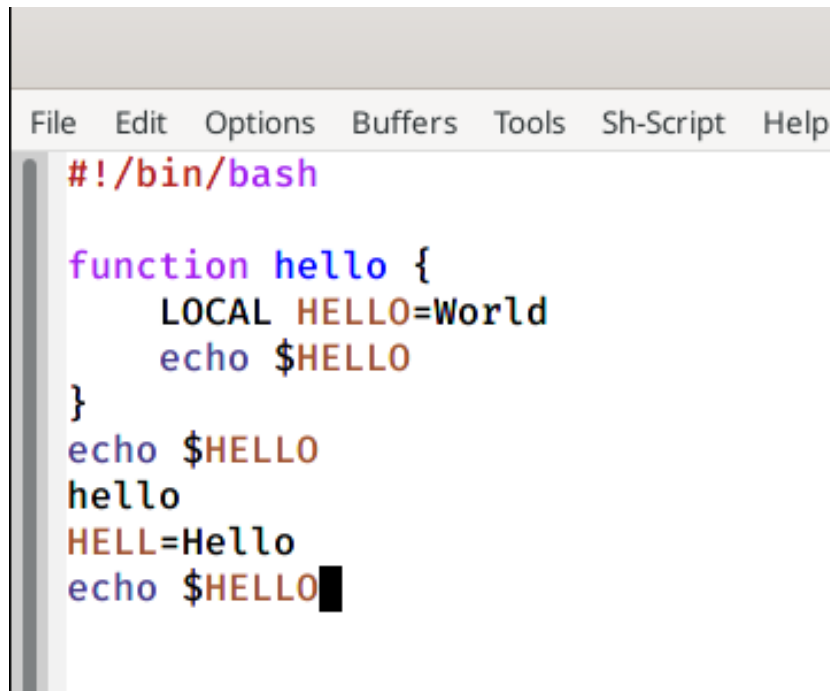
```
#!/bin/bash

function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
HELL=Hello
echo $HELLO
```

The cursor is positioned at the beginning of the line 'echo \$HELLO'.

Figure 3.14: Перемещаем курсор в начало буфера

4) Переместим курсор в конец буфера («М->») (рис. 3.15).



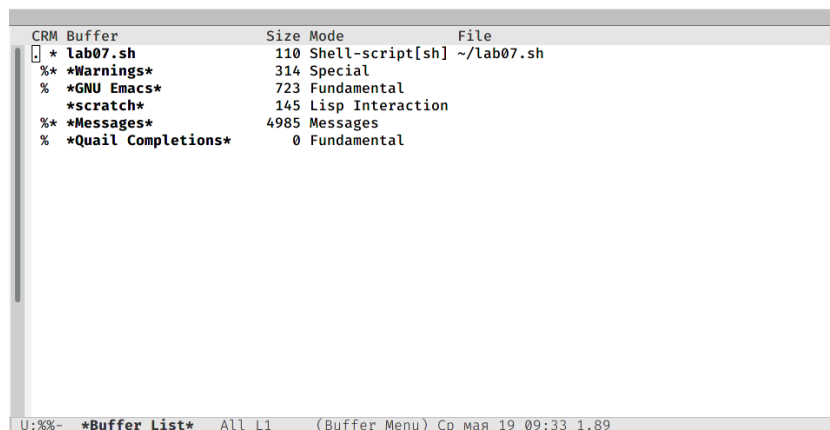
```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash

function hello {
    LOCAL HELLO=World
    echo $HELLO
}
echo $HELLO
hello
HELL=Hello
echo $HELLO
```

Figure 3.15: Перемещаем курсор в конец буфера

7.

1) Выведем список активных буферов на экран («Ctrl-x»«Ctrl-b») (рис. 3.16).



CRM	Buffer	Size	Mode	File
	* lab07.sh	110	Shell-script[sh]	~/lab07.sh
%	*Warnings*	314	Special	
%	*GNU Emacs*	723	Fundamental	
%	*scratch*	145	Lisp Interaction	
%	*Messages*	4985	Messages	
%	*Quail Completions*	0	Fundamental	

U:%%- *Buffer List* All L1 (Buffer Menu) Ср мая 19 09:33 1.89

Figure 3.16: Список активных буферов

- 2) Переместимся во вновь открытое окно («Ctrl-x o») со списком открытых буферов и переключимся на другой буфер (для этого необходимо нажать на «enter» после выбора необходимого буфера) (рис. 3.17).

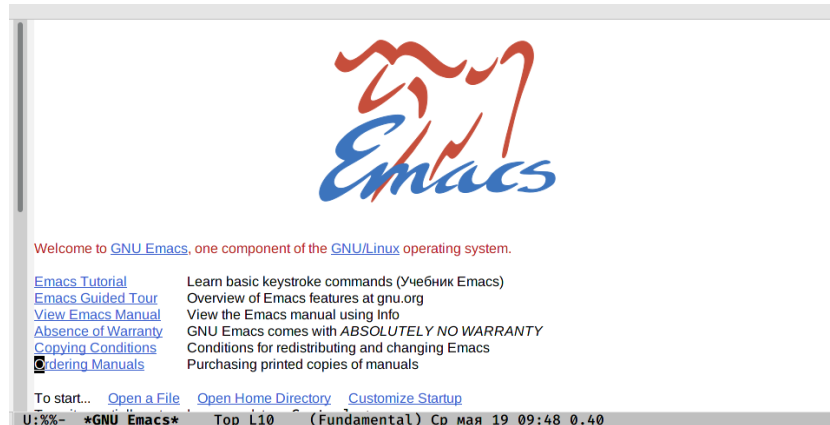


Figure 3.17: Переключаемся на другой буфер

- 3) Закроем это окно («Ctrl-x0») (рис. 3.18).

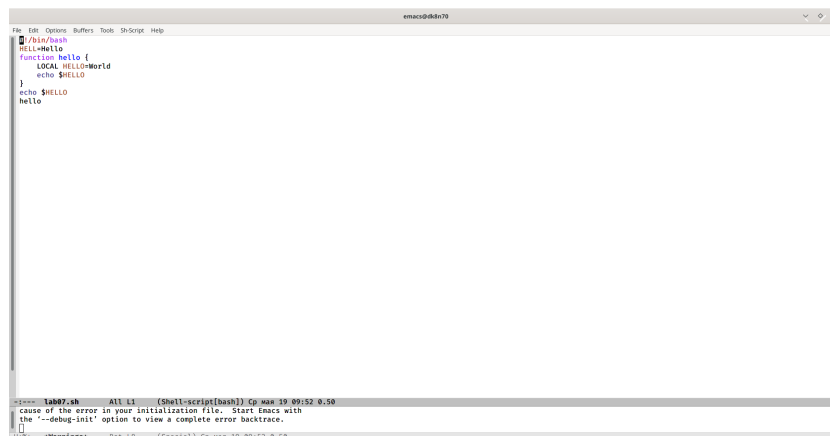
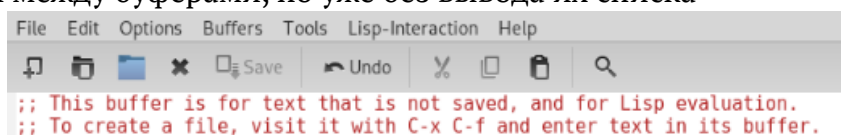


Figure 3.18: Закрываем окно

- 4) Теперь вновь переключимся между буферами, но уже без вывода их списка



на экран («Ctrl-x b») (рис. ??).

8.

- 1) Поделим фрейм на 4 части: разделим фрейм на два окна по вертикали («Ctrl-x 3»), а затем каждое из этих окон на две части по горизонтали («Ctrl-x 2») (рис. 3.19).

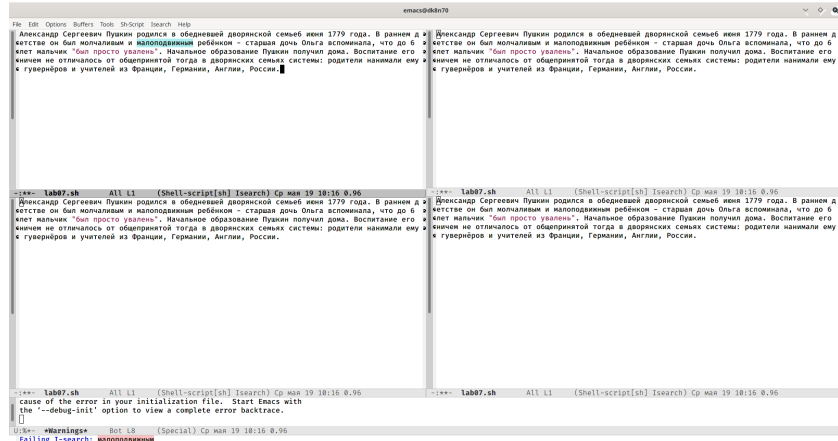


Figure 3.19: Разделяем фрейм на 4 части

- 2) В каждом из четырёх созданных окон откроем новый буфер и введем несколько строк текста (рис. 3.20).

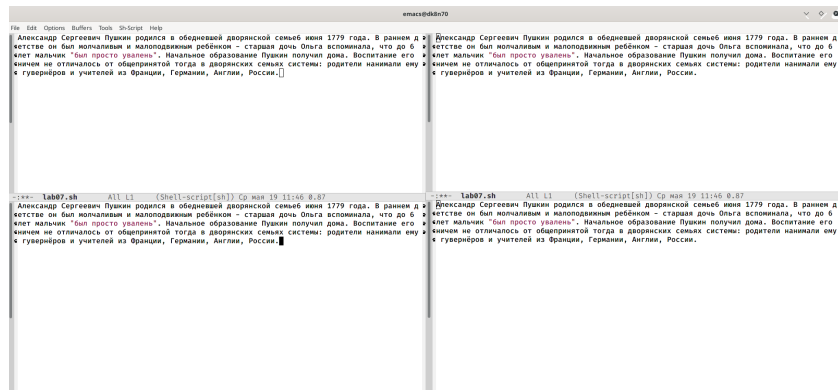


Figure 3.20: Вводим текст

9.

- 1) Переключимся в режим поиска («Ctrl-s») и найдем несколько слов, присутствующих в тексте (рис. 3.21, 3.22).

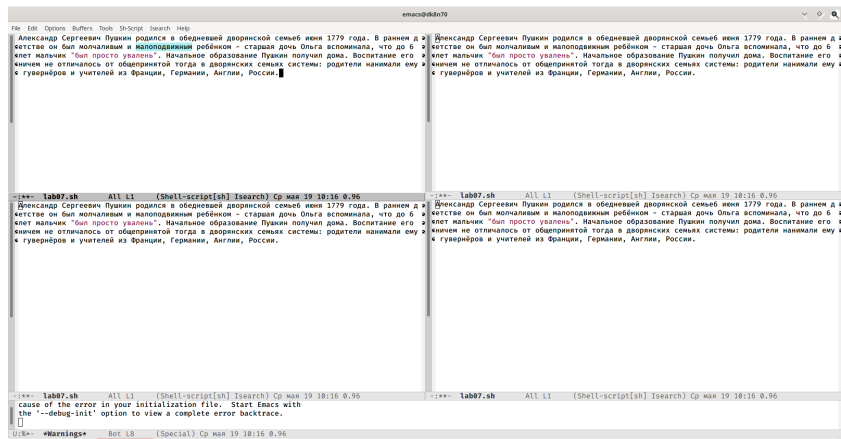


Figure 3.21: Ищем слова

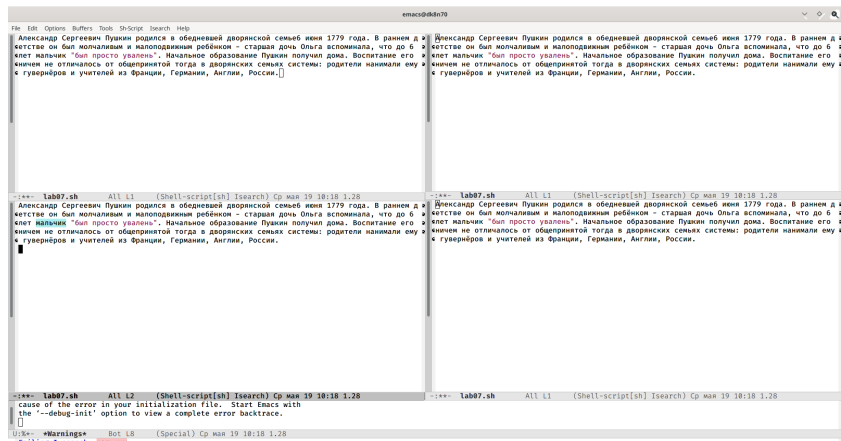


Figure 3.22: Ищем слова

2) Переключимся между результатами поиска, нажимая «Ctrl-s» (рис. 3.23).

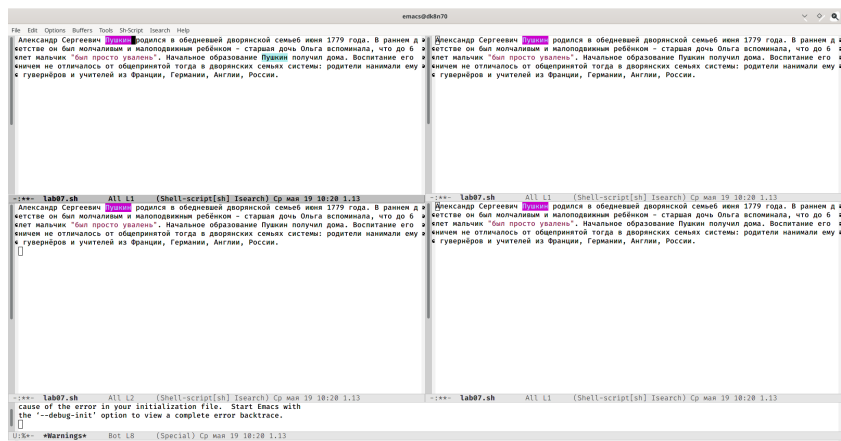


Figure 3.23: Переключаемся между результатами поиска

3) Выйдем из режима поиска, нажав «Ctrl-g» (рис. 3.24).

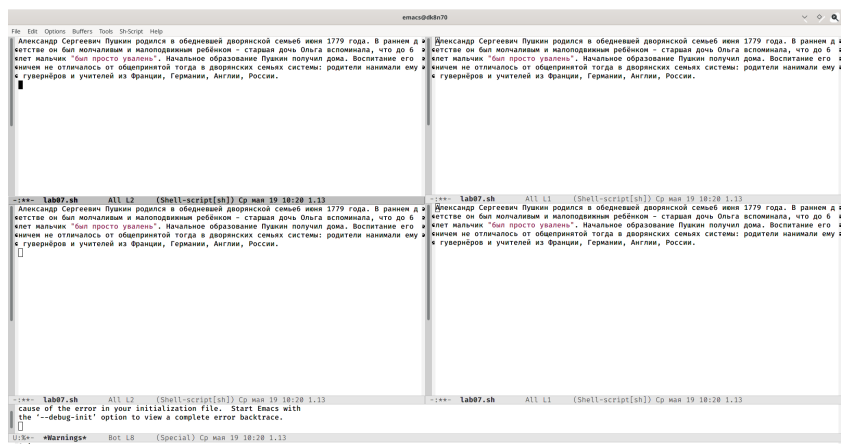


Figure 3.24: Выходим из режима поиска

4) Перейдем в режим поиска и замены («M-%»), введем текст, который следует найти и заменить, нажмем «enter», затем введем текст для замены. После того как будут подсвечены результаты поиска, нажмем «!» для подтверждения замены (рис. 3.25, 3.26, 3.27).

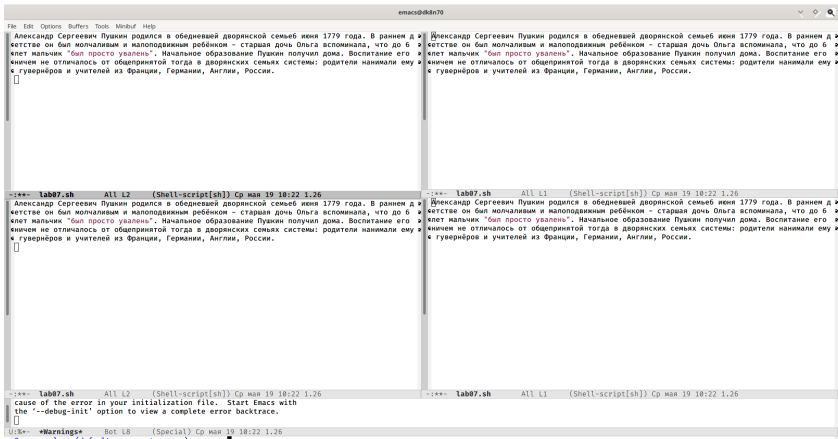


Figure 3.25: Заменяем слово в тексте

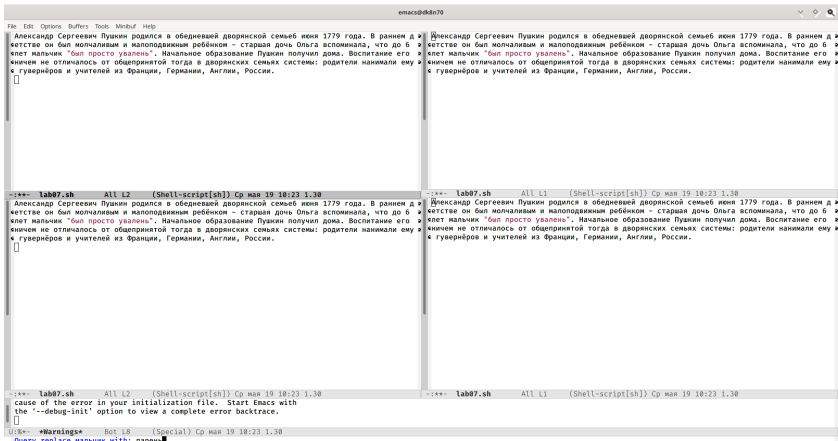


Figure 3.26: Заменяем слово в тексте

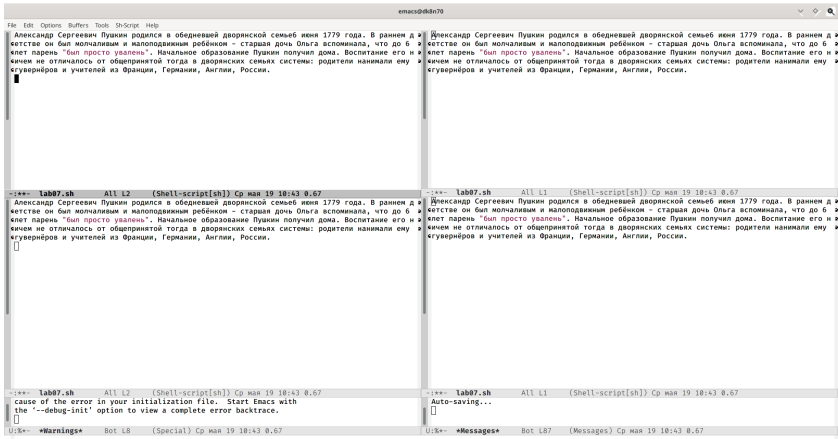


Figure 3.27: Заменяем слово в тексте

- 5) К сожалению, на данном компьютере не удалось с помощью команды («M-s o») испробовать другой режим поиска, поэтому на предыдущем этапе была завершена лабораторная работа. Однако данный вид поиска отличается от обычного тем, что тут считывается строка поиска, которая трактуется как регулярное выражение, и не осуществляется поиск точного совпадения в тексте буфера. Регулярное выражение – это образец, который обозначает набор строк, возможно, и неограниченный набор.

4 Выводы

В ходе выполнения данной лабораторной работы я познакомилась с операционной системой Linux и получила практические навыки работы с редактором Emacs.


5 Ответы на контрольные вопросы

1. Emacs – один из наиболее мощных и широко распространённых редакторов, используемых в мире Unix. По популярности он соперничает с редактором vi и его клонами. В зависимости от ситуации, Emacs может быть:

- текстовым редактором;
- программой для чтения почты и новостей Usenet;
- интегрированной средой разработки (IDE);
- операционной системой и т.д.

Всё это разнообразие достигается благодаря архитектуре Emacs, которая позволяет расширять возможности редактора при помощи языка Emacs Lisp. На языке C написаны лишь самые базовые и низкоуровневые части Emacs, включая полнофункциональный интерпретатор языка Lisp. Таким образом, Emacs имеет встроенный язык программирования, который может использоваться для настройки, расширения и изменения поведения редактора. В действительности, большая часть того редактора, с которым пользователи Emacs работают в наши дни, написана на языке Lisp.

2. Основную трудность для новичков при освоении данного редактора могут составлять большое количество команд, комбинаций клавиш, которые не получится все запомнить с первого раза и поэтому придется часто обращаться к справочным материалам.

3. Буфер –это объект, представляющий собой текст. Если имеется несколько буферов, то редактировать можно только один. Обычно буфер считывает данные из файла или записывает в файл данные из буфера.Окно –это область экрана, отображающая буфер. При запуске редактора отображается одно окно, но при обращении к некоторым функциям могут открыться дополнительные окна. Окна Emacs и окна графической среды XWindow– разные вещи. Одно окно XWindow может быть разбито на несколько окон в смысле Emacs, в каждом из которых отображается отдельный буфер.
4. Да, можно.
5. При запуске Emacs по умолчанию создаются следующие буферы:
- «scratch»(буфер для несохраненного текста)
 - «Messages»(журнал ошибок, включающий также информацию, которая появляется в области EchoArea)
 - «GNUEmacs»(справочный буфер о редакторе)
6. C-c |сначала, удерживая «ctrl»,нажимаю «с»,после –отпускаю обе клавишии нажимаю «|» C-cC-|сначала, удерживая «ctrl»,нажимаю «с», после –отпускаю обе клавиши и, удерживая «ctrl», нажимаю «|»
7. Чтобы поделить окно на две части необходимо воспользоваться комбинацией «Ctrl-x 3»(по вертикали) или «Ctrl-x 2» (по горизонтали).
8. Настройки Emacs сохраняются в файле .emacs.
9. По умолчанию клавиша «» удаляет символ перед курсором, нов редакторе её можно переназначить. Для этого необходимоизменить конфигурацию файла .emacs.
10. Более удобным я считаю редактор emacs, потому что в нем проще открывать другие файлы, можно использовать сразу несколько окон, нет «Командного

режима», «Режима ввода», «Режима командной строки», которые являются немного непривычными и в какой-то степени неудобным