

Отчёт по лабораторной работе №9

Операционные системы

Сячинова Ксения Ивановна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	18
5	Ответы на контрольные вопросы	19

Список иллюстраций

3.1	Запуск emacs	7
3.2	Создание файла	7
3.3	Текст	8
3.4	Вырезаем строку	8
3.5	Вставляем строку в конец	9
3.6	Вставляем строку в конец	9
3.7	Вставка области	10
3.8	Вырезаем область	10
3.9	Отмена действия	11
3.10	В начало строки	11
3.11	В конец строки	12
3.12	В начало буфера	12
3.13	В конец буфера	13
3.14	Список активных буферов	13
3.15	Переключение на другой буфер	14
3.16	Переключение между буферами	14
3.17	Разделение окна на 4 части	15
3.18	Новые файлы	15
3.19	Поиск слов	16
3.20	Переход	16
3.21	Замена слов	17
3.22	Итог	17

Список таблиц

1 Цель работы

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

2 Задание

1. Ознакомиться с теоретическим материалом.
2. Ознакомиться с редактором емас.
3. Выполнить упражнения.
4. Ответить на контрольные вопросы.

3 Выполнение лабораторной работы

1. Для работы в “Emacs” необходимо установить его. Открываем его с помощью команды “emacs &” (рис. 3.1).

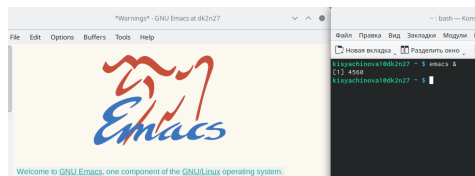


Рис. 3.1: Запуск emacs

2. Создаём файл lab07.sh и с помощью комбинаций “ctrl-x”, “ctrl-f” открываем его.(рис. 3.2).

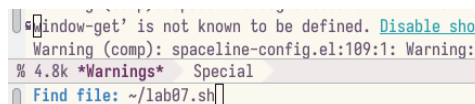
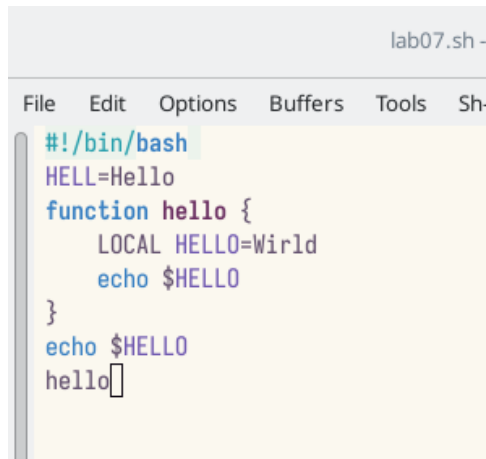


Рис. 3.2: Создание файла

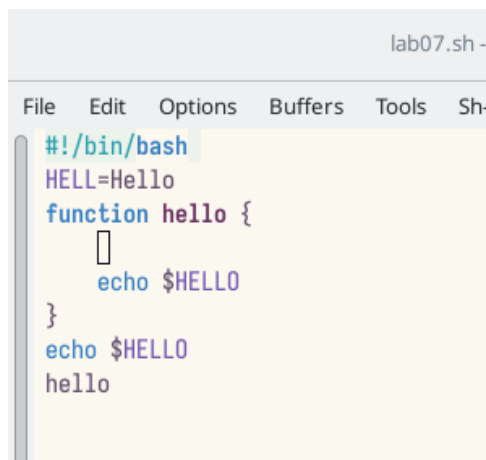
3. Напишем необходимый текст. (рис. 3.3).



```
lab07.sh -
File Edit Options Buffers Tools Sh-
#!/bin/bash
HELL=Hello
function hello {
    LOCAL HELLO=WorlD
    echo $HELLO
}
echo $HELLO
hello
```

Рис. 3.3: Текст

4. Сохраняем файл с помощью комбинаций клавиш “ctrl-x”, “ctrl-s”
5. Выполним ряд действий:
 - 5.1. Вырезать одной командой целую строку (C-k).(рис. 3.4).



```
lab07.sh -
File Edit Options Buffers Tools Sh-
#!/bin/bash
HELL=Hello
function hello {
    echo $HELLO
}
echo $HELLO
hello
```

Рис. 3.4: Вырезаем строку

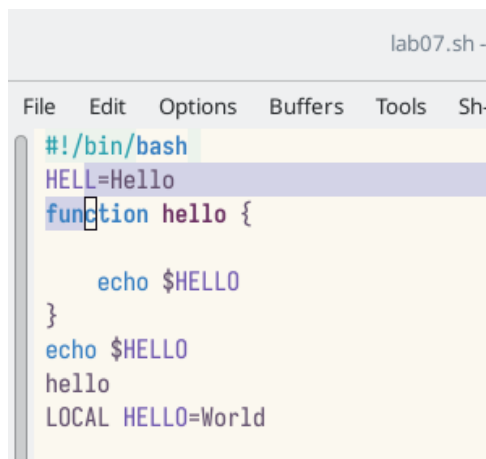
- 5.2. Вставить эту строку в конец файла (C-y).(рис. 3.5).



```
lab07.sh -
File Edit Options Buffers Tools Sh-
#!/bin/bash
HELL=Hello
function hello {
    echo $HELLO
}
echo $HELLO
hello
LOCAL HELLO=World
```

Рис. 3.5: Вставляем строку в конец

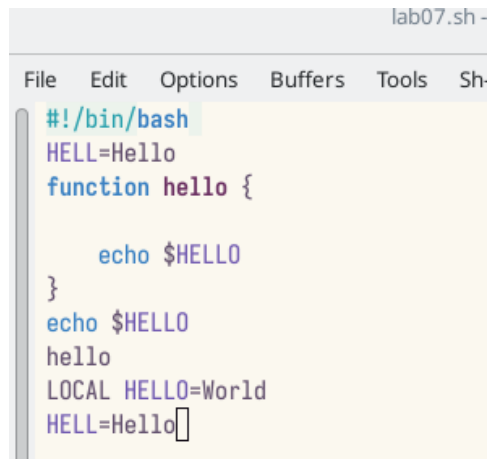
- 5.3. Выделить область текста (C-space).(рис. 3.6).



```
lab07.sh -
File Edit Options Buffers Tools Sh-
#!/bin/bash
HELL=Hello
function hello {
    echo $HELLO
}
echo $HELLO
hello
LOCAL HELLO=World
```

Рис. 3.6: Вставляем строку в конец

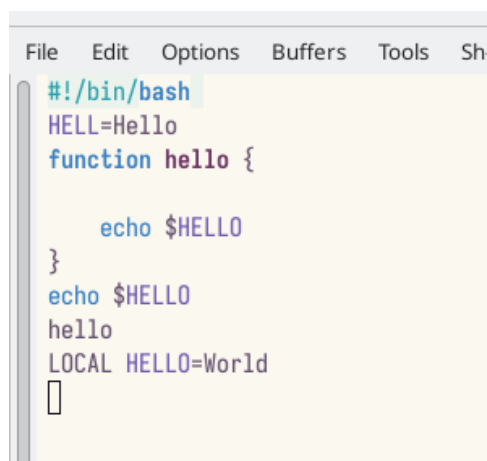
- 5.4. Скопировать область в буфер обмена (M-w).
- 5.5. Вставить область в конец файла.(рис. 3.7)



```
lab07.sh -
File Edit Options Buffers Tools Sh-
#!/bin/bash
HELL=Hello
function hello {
    echo $HELLO
}
echo $HELLO
hello
LOCAL HELLO=World
HELL=Hello
```

Рис. 3.7: Вставка области

- 5.6. Вновь выделить эту область и на этот раз вырезать её (C-w).(рис. 3.8)

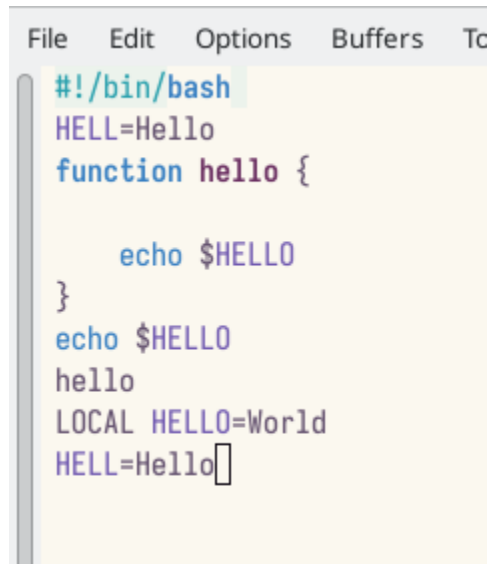


```
File Edit Options Buffers Tools Sh-
#!/bin/bash
HELL=Hello
function hello {
    echo $HELLO
}
echo $HELLO
hello
LOCAL HELLO=World

```

Рис. 3.8: Вырезаем область

- 5.7. Отмените последнее действие (C-/).(рис. 3.9)

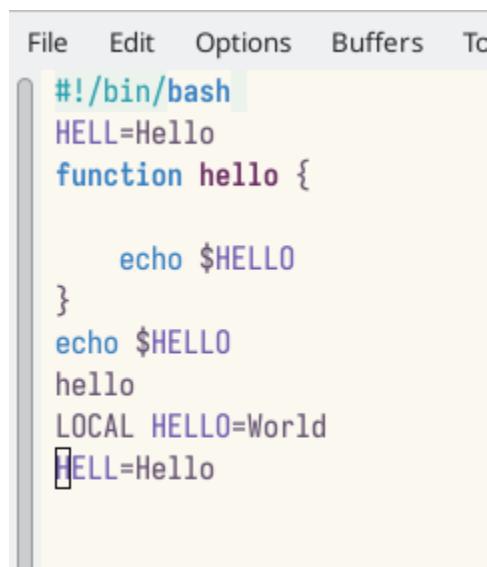


```
File Edit Options Buffers To
#!/bin/bash
HELL=Hello
function hello {
    echo $HELLO
}
echo $HELLO
hello
LOCAL HELLO=World
HELL=Hello
```

Рис. 3.9: Отмена действия

6. Действия с курсором

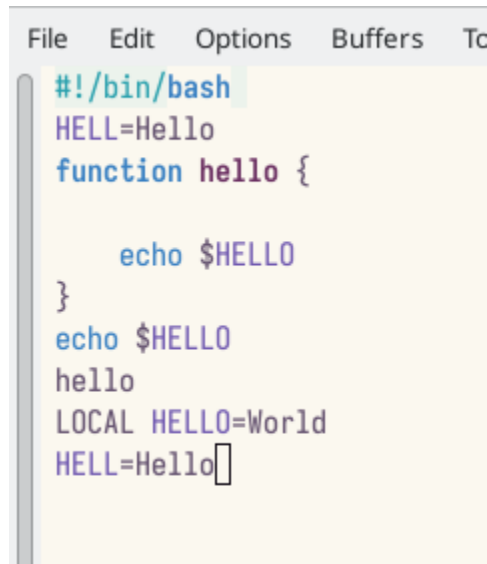
- 6.1. Переместите курсор в начало строки (C-a).(рис. 3.10)



```
File Edit Options Buffers To
#!/bin/bash
HELL=Hello
function hello {
    echo $HELLO
}
echo $HELLO
hello
LOCAL HELLO=World
HELL=Hello
```

Рис. 3.10: В начало строки

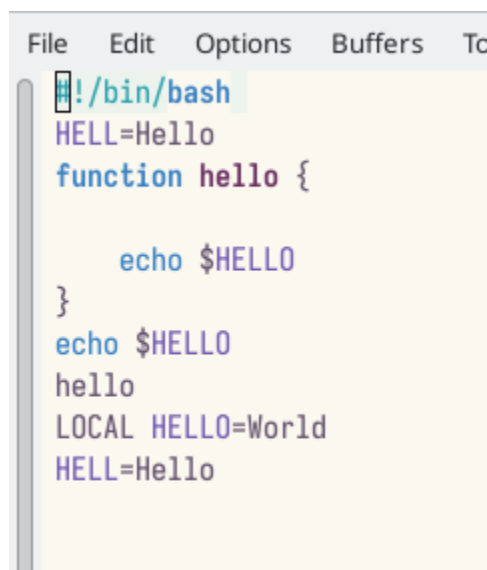
- 6.2. Переместите курсор в конец строки (C-e).(рис. 3.11)



```
File Edit Options Buffers To
#!/bin/bash
HELL=Hello
function hello {
    echo $HELLO
}
echo $HELLO
hello
LOCAL HELLO=World
HELL=Hello
```

Рис. 3.11: В конец строки

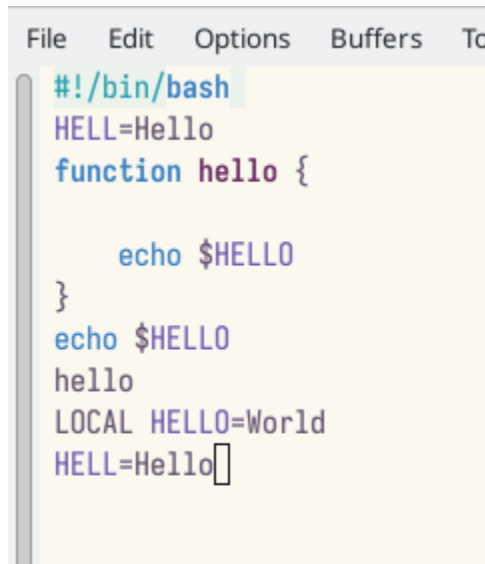
- 6.3. Переместите курсор в начало буфера (M-<).(рис. 3.12)



```
File Edit Options Buffers To
#!/bin/bash
HELL=Hello
function hello {
    echo $HELLO
}
echo $HELLO
hello
LOCAL HELLO=World
HELL=Hello
```

Рис. 3.12: В начало буфера

- 6.4. Переместите курсор в конец буфера (M->).(рис. 3.13)

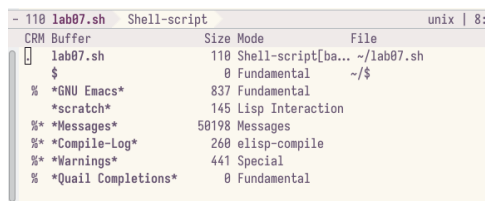


```
#!/bin/bash
HELL=Hello
function hello {
    echo $HELLO
}
echo $HELLO
hello
LOCAL HELLO=World
HELL=Hello
```

Рис. 3.13: В конец буфера

7. Управление буферами.

- 7.1. Вывести список активных буферов на экран (C-x C-b).(рис. 3.14)



CRM Buffer	Size	Mode	File
lab07.sh	110	Shell-script[ba...	~/lab07.sh
\$	0	Fundamental	~/
% *GNU Emacs*	837	Fundamental	
% *scratch*	145	Lisp Interaction	
%* *Messages*	50198	Messages	
%* *Compile-Log*	260	elisp-compile	
%* *Warnings*	441	Special	
% *Quail Completions*	0	Fundamental	

Рис. 3.14: Список активных буферов

- 7.2. Переместим во вновь открытое окно (C-x o) со списком открытых буферов и переключимся на другой буфер (для этого нажмём “enter”)(рис. 3.15)

* 111 lab07.sh Shell-script unix | 11:

CRM	Buffer	Size	Mode	File
	lab07.sh	110	Shell-script[ba...	~/lab07.sh
	\$	0	Fundamental	~/
%	*GNU Emacs*	837	Fundamental	
%	*scratch*	145	Lisp Interaction	
%	*Messages*	50198	Messages	
%	*Compile-Log*	260	elisp-compile	
%	*Warnings*	441	Special	
%	*Quail Completions*	0	Fundamental	

Рис. 3.15: Переключение на другой буфер

- 7.3. Закройте это окно (C-x 0).

7.4. Теперь вновь переключайтесь между буферами, но уже без вывода их списка на экран (C-x b).

File	Edit	Options	Buffers	Tools	Buffer-Menu	Help
CRM	Buffer		Size	Mode	File	
	lab07.sh		110	Shell-script[ba...	~/lab07.sh	
	\$		0	Fundamental	~/	
%	*GNU Emacs*		837	Fundamental		
%	*scratch*		145	Lisp Interaction		
%	*Messages*		50198	Messages		
%	*Compile-Log*		260	elisp-compile		
%	*Warnings*		441	Special		
%	*Quail Completions*		0	Fundamental		

Рис. 3.16: Переключение между буферами

8. Управление окнами.

- 8.1. Поделите фрейм на 4 части: разделите фрейм на два окна по вертикали (C-x 3), а затем каждое из этих окон на две части по горизонтали (C-x 2) (рис. 3.17)

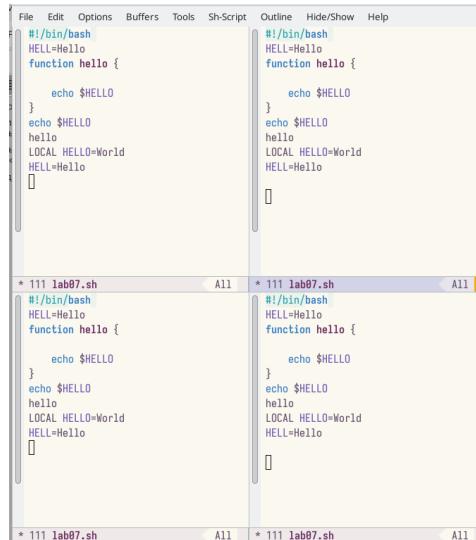


Рис. 3.17: Разделение окна на 4 части

- 8.2. В каждом из четырёх созданных окон откроем новый буфер (файл) и введём несколько строк текста. Для этого я заранее создала 4 файла с разным текстом.(рис. 3.18)

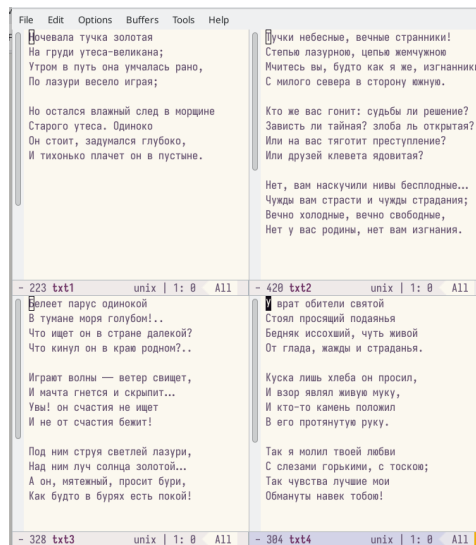


Рис. 3.18: Новые файлы

9. Режим поиска

- 9.1. Переключитесь в режим поиска (C-s) и найдите несколько слов, присутствующих в тексте.(рис. 3.19)

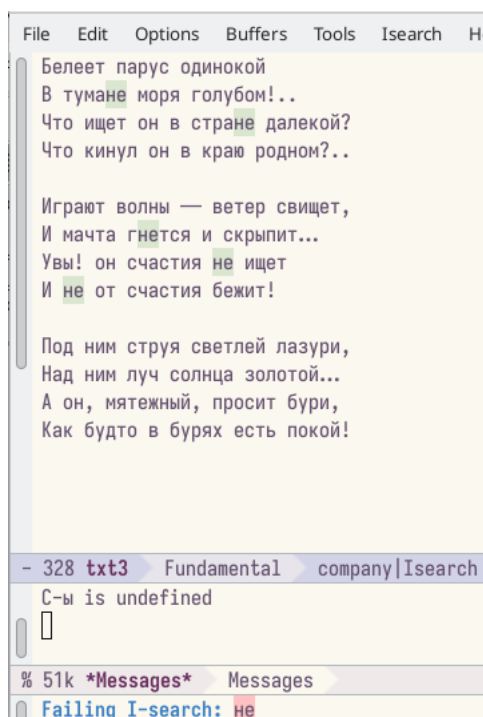


Рис. 3.19: Поиск слов

- 9.2. Переключайтесь между результатами поиска, нажимая C-s(рис. 3.20)

Рис. 3.20: Переход

- 9.3. Выйдите из режима поиска, нажав C-g.
- 9.4. Перейдите в режим поиска и замены (M-%), введите текст, который следует найти и заменить, нажмите Enter , затем введите текст для замены. После того как будут подсвечены результаты поиска, нажмите ! для подтверждения замены.(рис. 3.21),(рис. 3.22)



Рис. 3.21: Замена слов

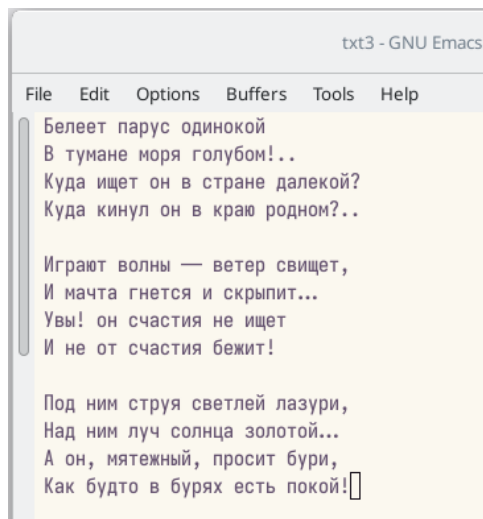


Рис. 3.22: Итог

- 9.5. Попробуем режим поиска (M-s o). Данный поиск отличается тем, что тут считывается строка поиска, которая трактуется как регулярное выражение, и не осуществляем поиск точно 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 по работе 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 X Window – разные вещи. Одно окно X Window 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)
 - «GNU Emacs» (справочный буфер о редакторе)
6. C-c | сначала, удерживая «ctrl», нажимаю «с», после – отпускаю обе клавиши и нажимаю «|» C-c C-| 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 хранятся в файле .emacs.
9. По умолчанию клавиша «» удаляет символ перед курсором, но в редакторе её можно переназначить. Для этого необходимо изменить 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 неудобными.