

Лабораторная работа №9

Дисциплина - Операционные системы

Оширова Юлия Николаевна

Содержание

1 Цель работы	5
2 Задание	6
3 Теоретическое введение	8
4 Выполнение лабораторной работы	10
5 Выводы	20
6 Контрольные вопросы	21
Список литературы	23

Список иллюстраций

4.1 Редактор emacs	10
4.2 Редактор emacs	11
4.3 Текст в редакторе	11
4.4 Вырезание в строчку	12
4.5 Вставка строчки в конец файла	13
4.6 Вставка скопированной области в конец файла	13
4.7 Вырезание области текста	14
4.8 Отмена последнего действия	14
4.9 Курсор в начале строки	15
4.10 Курсор в конце строки	15
4.11 Список активных буферов	16
4.12 Другой буфер	16
4.13 4 окна	17
4.14 Новые файлы в каждом окне	17
4.15 Режим поиска	18
4.16 Режим поиска	18
4.17 Режим поиска и замены	19
4.18 Другой режим поиска	19

Список таблиц

1 Цель работы

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

2 Задание

Последовательность выполнения работы 1. Ознакомиться с теоретическим материалом. 2. Ознакомиться с редактором emacs. 3. Выполнить упражнения. 4. Ответить на контрольные вопросы.

Основные команды emacs 1. Открыть emacs. 2. Создать файл lab07.sh с помощью комбинации Ctrl-x Ctrl-f (C-x C-f). 3. Наберите текст, который дан. 4. Сохранить файл с помощью комбинации Ctrl-x Ctrl-s (C-x C-s). 5. Проделать с текстом стандартные процедуры редактирования, каждое действие должно осуществляться комбинацией клавиш. 5.1. Вырезать одной командой целую строку (C-k). 5.2. Вставить эту строку в конец файла (C-y). 5.3. Выделить область текста (C-space). 5.4. Скопировать область в буфер обмена (M-w). 5.5. Вставить область в конец файла. 5.6. Вновь выделить эту область и на этот раз вырезать её (C-w). 5.7. Отмените последнее действие (C-/). 6. Научитесь использовать команды по перемещению курсора. 6.1. Переместите курсор в начало строки (C-a). 6.2. Переместите курсор в конец строки (C-e). 6.3. Переместите курсор в начало буфера (M-). 6.4. Переместите курсор в конец буфера (M->). 7. Управление буферами. 7.1. Вывести список активных буферов на экран (C-x C-b). 7.2. Переместитесь во вновь открытое окно (C-x) о со списком открытых буферов и переключитесь 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). 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). 8.2. В каждом из четырёх созданных окон откройте новый буфер (файл) и введите несколько строк текста.

9. Режим поиска 9.1. Переключитесь в режим поиска (`C-s`) и найдите несколько слов, присутствующих в тексте. 9.2. Переключайтесь между результатами поиска, нажимая `C-s`. 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. 9.5. Испробуйте другой режим поиска, нажав `M-s` о. Объясните, чем он отличается от обычного режима?

3 Теоретическое введение

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;

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.

Первая версия редактора Emacs была написана в 70-х годах 20-го столетия Richard Stallman (Ричардом Столманом) как набор макросов для редактора TECO. В дальнейшем, уже будучи основателем Фонда Свободного программного обеспечения Free Software Foundation и проекта GNU, Stallman разработал GNU Emacs в развитие оригинального Emacs и до сих пор сопровождает эту программу. Emacs является одним из старейших редакторов. Он использовался тысячами программистов на протяжении последних 20 с лишним лет, для него создано

много дополнительных пакетов расширений. Эти дополнения позволяют делать с помощью Emacs такие вещи, которые Stallman , вероятно, даже не считал возможными в начале своей работы над редактором. [1]

4 Выполнение лабораторной работы

1. Открыть emacs.

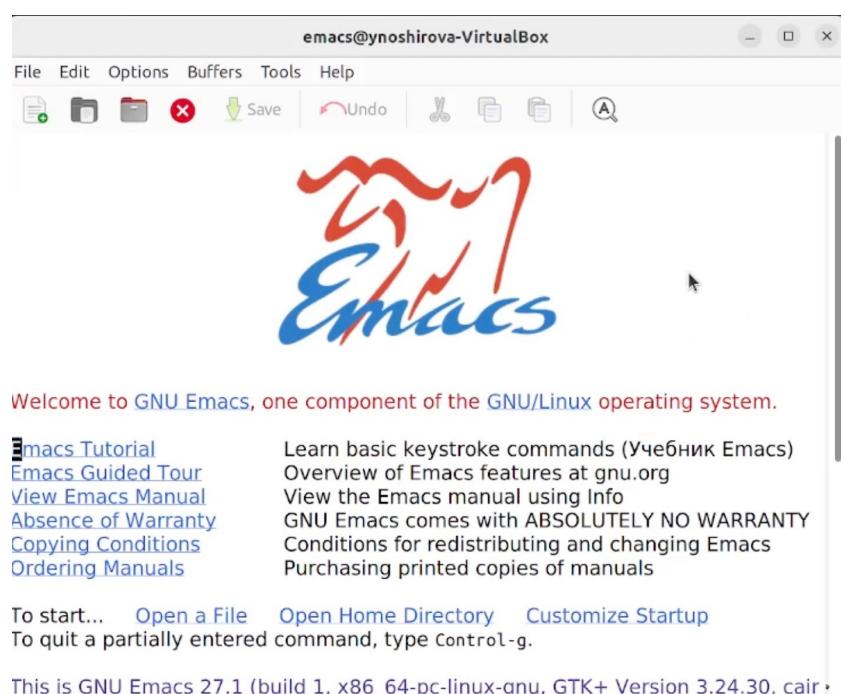


Рис. 4.1: Редактор emacs

2. Создать файл lab07.sh с помощью комбинации Ctrl-x Ctrl-f (C-x C-f).

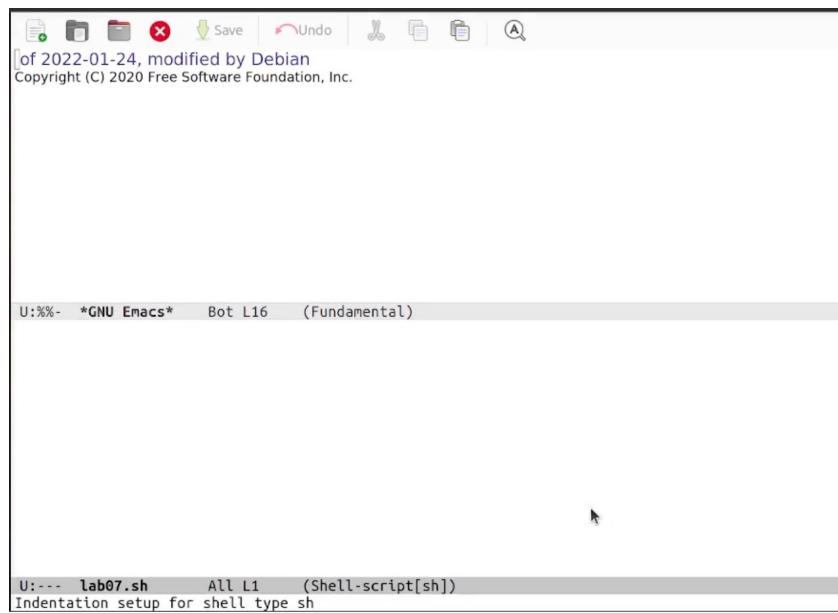


Рис. 4.2: Редактор emacs

3. Наберите текст, который дан.

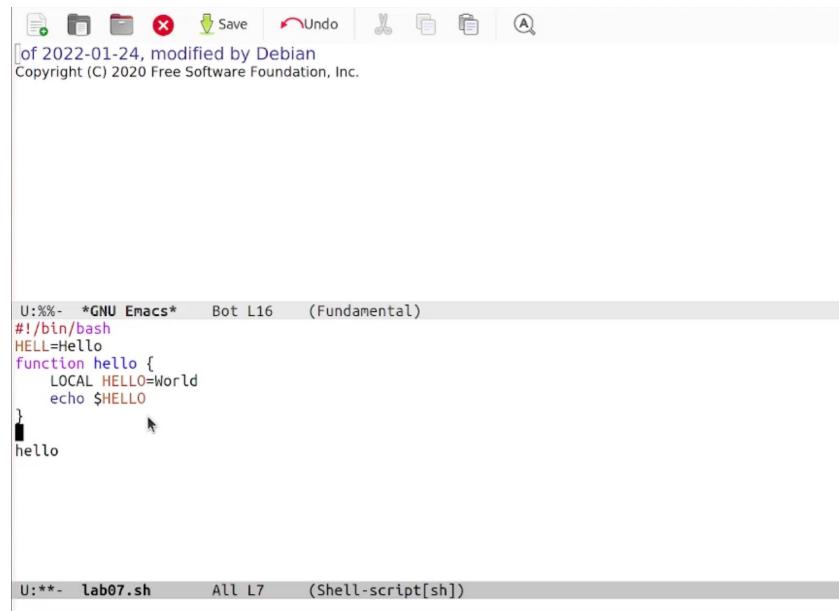
4. Сохранить файл с помощью комбинации Ctrl-x Ctrl-s (C-x C-s).

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

The screenshot shows the same Emacs interface as Figure 4.2, but with a buffer containing a shell script. The script defines a variable \$HELLO and a function hello that prints \$HELLO. It then prints \$HELLO again and calls the hello function. The cursor is at the end of the word "hello". The status bar at the bottom indicates the file is being auto-saved.

Рис. 4.3: Текст в редакторе

5. Проделать с текстом стандартные процедуры редактирования, каждое действие должно осуществляться комбинацией клавиш. 5.1. Вырезать одной командой целую строку (C-k).



The screenshot shows the Emacs editor interface. The top bar displays file icons (document, folder, etc.), a save button, an undo button, and other mode-specific icons. Below the bar, a status message reads "U:%%- *GNU Emacs* Bot L16 (Fundamental)" and "Copyright (C) 2020 Free Software Foundation, Inc.". The main editing area contains the following code:

```
U:%%- *GNU Emacs* Bot L16 (Fundamental)
#!/bin/bash
HELLO=Hello
function hello {
    LOCAL HELLO=World
    echo $HELLO
}
hello
```

A cursor is visible at the end of the word "hello". The bottom status bar indicates "U:**- lab07.sh All L7 (Shell-script[sh])".

Рис. 4.4: Вырезание в строчку

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

The screenshot shows the Emacs text editor interface. The top status bar indicates the buffer is named 'lab07.sh', contains 9 lines, and is a shell script. The bottom status bar shows 'Mark set'. The main window displays the following code:

```
U:%%- *GNU Emacs* Bot L16 (Fundamental)
#!/bin/bash
HELLO=Hello
function hello {
    LOCAL HELLO=World
    echo $HELLO
}

hello
echo $HELLO
```

Рис. 4.5: Вставка строчки в конец файла

5.3. Выделить область текста (C-space). 5.4. Скопировать область в буфер обмена (M-w). 5.5. Вставить область в конец файла.

The screenshot shows the Emacs text editor interface. The top status bar indicates the buffer is named 'lab07.sh', contains 14 lines, and is a shell script. The bottom status bar shows 'Mark set'. The main window displays the following code, with the last two lines of code from the previous screenshot now repeated at the end:

```
U:%%- *GNU Emacs* Bot L16 (Fundamental)
#!/bin/bash
HELLO=Hello
function hello {
    LOCAL HELLO=World
    echo $HELLO
}

hello
echo $HELLO
function hello {
    LOCAL HELLO=World
    echo $HELLO
}
U:**- lab07.sh All L14 (Shell-script[sh])
Mark set
```

Рис. 4.6: Вставка скопированной области в конец файла

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

The screenshot shows the Emacs interface with two windows. The top window is titled "U:%%- *GNU Emacs* Bot L16 (Fundamental)" and contains a shell script. The bottom window is titled "U:**- lab07.sh All L10 (Shell-script[sh])". In the top window, the text "function hello {" is selected. A cursor is visible at the end of the line "echo \$HELLO". The bottom window shows the script with the same line.

```
U:%%- *GNU Emacs* Bot L16 (Fundamental)
#!/bin/bash
HELLO=Hello
function hello {
    LOCAL HELLO=World
    echo $HELLO
}

hello
echo $HELLO

U:**- lab07.sh All L10 (Shell-script[sh])
```

Рис. 4.7: Вырезание области текста

5.7. Отмените последнее действие (C-/).

The screenshot shows the Emacs interface with two windows. The top window is titled "U:%%- *GNU Emacs* Bot L16 (Fundamental)" and contains a shell script. The bottom window is titled "U:**- lab07.sh All L12 (Shell-script[sh])". The text "function hello {" is now highlighted in purple, indicating it has been modified or is selected. The bottom window shows the script with the same line.

```
U:%%- *GNU Emacs* Bot L16 (Fundamental)
#!/bin/bash
HELLO=Hello
function hello {
    LOCAL HELLO=World
    echo $HELLO
}

hello
echo $HELLO
function hello {
    LOCAL HELLO=World
    echo $HELLO
}

U:**- lab07.sh All L12 (Shell-script[sh])
```

Рис. 4.8: Отмена последнего действия

6. Научитесь использовать команды по перемещению курсора. 6.1. Переместите курсор в начало строки (C-a).

The screenshot shows the Emacs interface with the following details:

- Toolbar icons: Save, Undo, Cut, Copy, Paste, Find, Replace.
- Buffer status bar: U:%%- *GNU Emacs* Bot L16 (Fundamental)
- File status bar: /of 2022-01-24, modified by Debian
- Copyright notice: Copyright (C) 2020 Free Software Foundation, Inc.
- Code buffer content:

```
U:%%- *GNU Emacs* Bot L16 (Fundamental)
#!/bin/bash
HELLO=Hello
function hello {
    LOCAL HELLO=World
    echo $HELLO
}

hello
echo $HELLO
function hello {
    LOCAL HELLO=World
    echo $HELLO
}

U:**- lab07.sh All L12 (Shell-script[sh])
```

Рис. 4.9: Курсор в начале строки

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

The screenshot shows the Emacs interface with the following details:

- Toolbar icons: Save, Undo, Cut, Copy, Paste, Find, Replace.
- Buffer status bar: U:%%- *GNU Emacs* Bot L16 (Fundamental)
- File status bar: /of 2022-01-24, modified by Debian
- Copyright notice: Copyright (C) 2020 Free Software Foundation, Inc.
- Code buffer content:

```
U:%%- *GNU Emacs* Bot L16 (Fundamental)
#!/bin/bash
HELLO=Hello
function hello {
    LOCAL HELLO=World
    echo $HELLO
}

hello
echo $HELLO
function hello {
    LOCAL HELLO=World
    echo $HELLO
}

U:**- lab07.sh All L12 (Shell-script[sh])
```

Рис. 4.10: Курсор в конце строки

6.3 Переместите курсор в начало буфера (M-<). 6.4. Переместите курсор в конец буфера (M->). 7. Управление буферами. 7.1. Вывести список активных буферов на

экран (**C-x C-b**).

The screenshot shows the Emacs interface with the Buffer List buffer active. The top menu bar includes 'File', 'Save', 'Undo', and other standard Emacs icons. Below the menu is a table showing buffer statistics:

CRM Buffer	Size	Mode	File
.* lab07.sh	157	Shell-script[sh]	~/lab07.sh
Messages	1095	Messages	
GNU Emacs	743	Fundamental	
scratch	145	Lisp Interaction	

The main area displays the content of the 'lab07.sh' buffer, which contains a shell script with a function 'hello' that prints 'Hello World'. The bottom status bar indicates 'U:%%- *Buffer List* All L1 (Buffer Menu)' and 'U:**- lab07.sh All L14 (Shell-script[sh])'.

Рис. 4.11: Список активных буферов

7.2. Переместитесь во вновь открытое окно (**C-x**) о со списком открытых буферов и переключитесь на другой буфер.

The screenshot shows the Emacs interface with the Buffer List buffer active. The top menu bar includes 'File', 'Save', 'Undo', and other standard Emacs icons. Below the menu is a table showing buffer statistics:

CRM Buffer	Size	Mode	File
.* lab07.sh	157	Shell-script[sh]	~/lab07.sh
Messages	1095	Messages	
GNU Emacs	743	Fundamental	
scratch	145	Lisp Interaction	

The main area displays the content of the 'lab07.sh' buffer, which contains a shell script with a function 'hello' that prints 'Hello World'. The bottom status bar indicates 'U:%%- *Buffer List* All L1 (Buffer Menu)' and 'U:**- lab07.sh All L13 (Shell-script[sh])'.

Рис. 4.12: 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). 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).

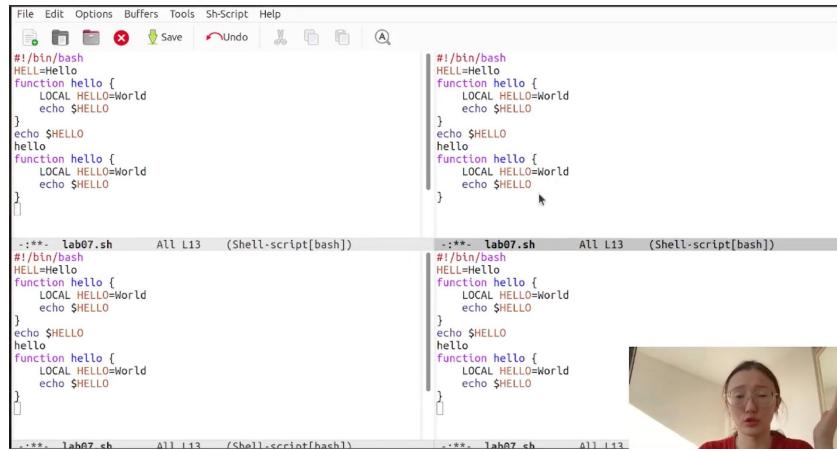


Рис. 4.13: 4 окна

8.2. В каждом из четырёх созданных окон откройте новый буфер (файл) и введите несколько строк текста.

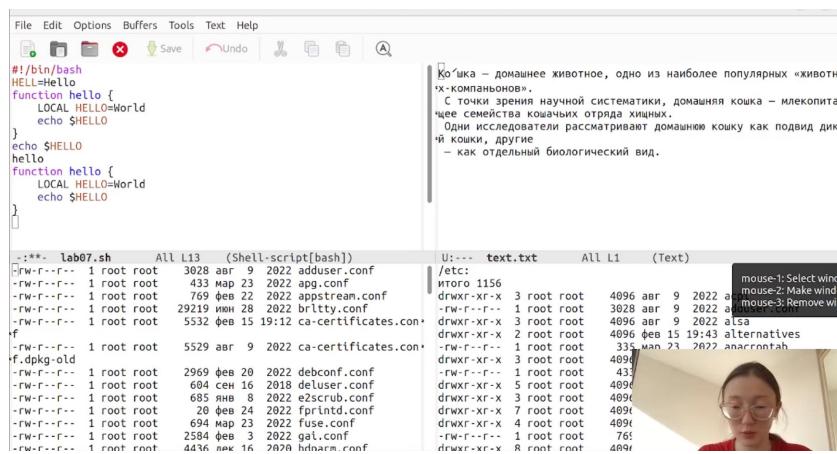


Рис. 4.14: Новые файлы в каждом окне

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

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

-:--- lab07.sh All L2 (Shell-script[bash] Isearch)
I-search: hello

Рис. 4.15: Режим поиска

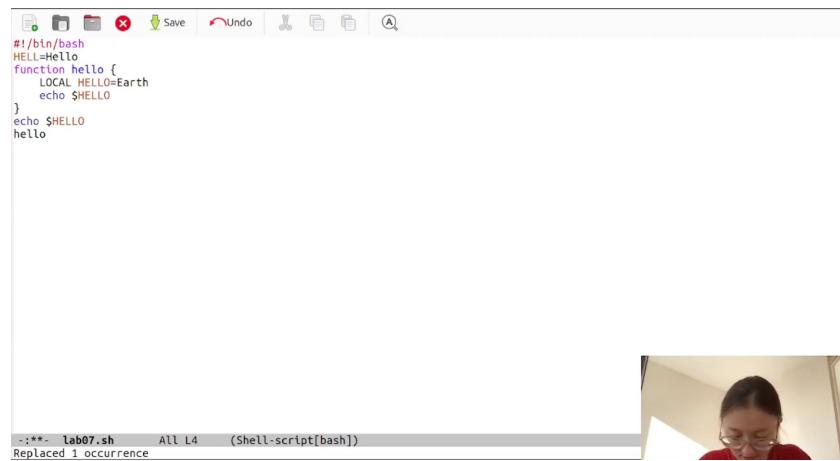
9.2. Переключайтесь между результатами поиска, нажимая C-s.

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

-:--- lab07.sh All L7 (Shell-script[bash] Isearch)
I-search: hello

Рис. 4.16: Режим поиска

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.



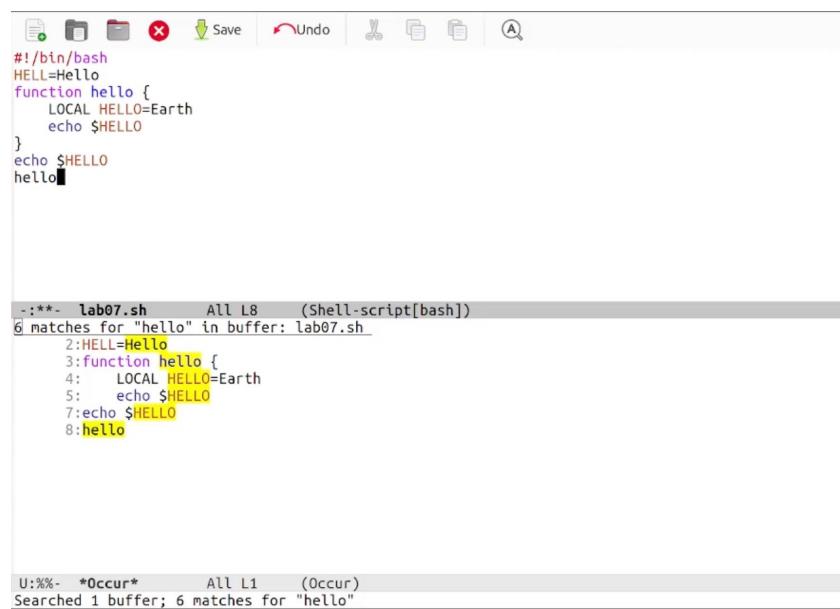
```
#!/bin/bash
HELL=Hello
function hello {
    LOCAL HELLO=Earth
    echo $HELLO
}
echo $HELLO
hello
```

-:*- lab07.sh All L4 (Shell-script[bash])
Replaced 1 occurrence



Рис. 4.17: Режим поиска и замены

9.5. Испробуйте другой режим поиска, нажав M-s o. Объясните, чем он отличается от обычного режима?



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

-:*- lab07.sh All L8 (Shell-script[bash])
6 matches for "hello" in buffer: lab07.sh
2:HELL=Hello
3:function hello {
4: LOCAL HELLO=Earth
5: echo \$HELLO
7:echo \$HELLO
8:hello

U:%%- *0curr* All L1 (0curr)
Searched 1 buffer; 6 matches for "hello"

Рис. 4.18: Другой режим поиска

Отличие от обычного режима в том, что тут появляется отдельное окно с текстом из файла с выделенными словами, которые нужно было найти.

5 Выводы

В процессе выполнения лабораторной работы я получила практические навыки работы в редакторе Emacs.

6 Контрольные вопросы

1. Кратко охарактеризуйте редактор emacs.

Emacs — один из наиболее мощных и широко распространённых редакторов, используемых в мире UNIX. Написан на языке высокого уровня Lisp.

2. Какие особенности данного редактора могут сделать его сложным для освоения новичком?

Большое разнообразие сложных комбинаций клавиш, которые необходимы для редактирования файла и в принципе для работы с Emacs.

3. Своими словами опишите, что такое буфер и окно в терминологии emacs'а.

Буфер - это объект в виде текста. Окно - это прямоугольная область, в которой отображен буфер.

4. Можно ли открыть больше 10 буферов в одном окне?

Да, можно.

5. Какие буфера создаются по умолчанию при запуске emacs?

Emacs использует буфера с именами, начинающимися с пробела, для внутренних целей. Отчасти он обращается с буферами с такими именами особым образом – например, по умолчанию в них не записывается информация для отмены изменений.

6. Какие клавиши вы нажмёте, чтобы ввести следующую комбинацию С-с | и С-с С-|?

Ctrl + c, а потом | и Ctrl + c Ctrl + |

7. Как поделить текущее окно на две части?

С помощью команды Ctrl + x 3 (по вертикали) и Ctrl + x 2 (по горизонтали).

8. В каком файле хранятся настройки редактора emacs?

Настройки emacs хранятся в файле .emacs, который хранится в домашней директории пользователя. Кроме этого файла есть ещё папка .emacs.

9. Какую функцию выполняет клавиша и можно ли её переназначить?

Выполняет функцию стереть, думаю можно переназначить.

10. Какой редактор вам показался удобнее в работе vi или emacs? Поясните почему.

Для меня удобнее был редактор Emacs, так как у него есть командная оболочка. А vi открывается в терминале, и выглядит своеобразно.

Список литературы

1. Emacs для начинающих [Электронный ресурс]