

Отчет по лабораторной работе №9

Текстовый редактор emacs

Кочкарев “sakochkarev” Станислав

Содержание

1	Цель работы	3
2	Задание	4
3	Выполнение лабораторной работы	5
4	Выводы	27
5	Ответы на контрольные вопросы	28

1 Цель работы

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

2 Задание

- Ознакомление с теоретическим материалом
- Ознакомление с редактором emacs
- Выполнение упражнений с редактором emacs

3 Выполнение лабораторной работы

Предварительно был установлен текстовый редактор emacs.

Первым делом были изучены теоретические материалы и было произведено ознакомление с редактором emacs. После этого мы перешли к выполнению упражнений.

Сперва-наперво мы открыли сам редактор emacs (рис. 3.1).

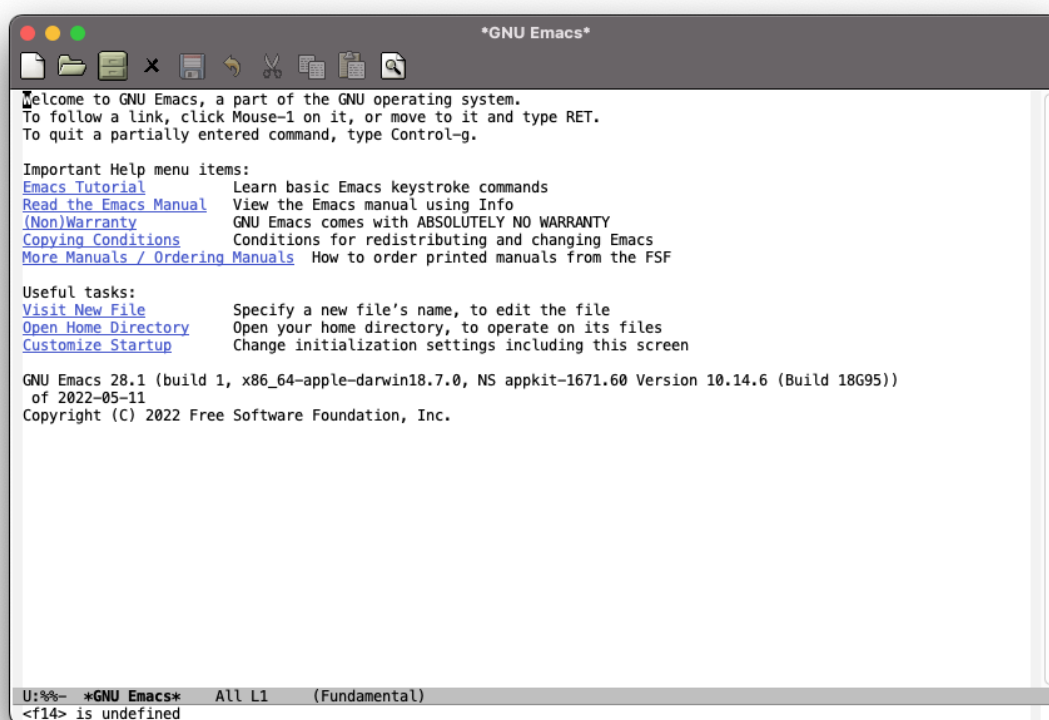


Рис. 3.1: Открытый редактор emacs

Далее был создан файл `lab07.sh` с помощью приведенной комбинации клавиш (рис. 3.2).



Рис. 3.2: Открытый новосозданный файл `lab07.sh`

После этого в открытом файле был написан приведенный текст (рис. 3.3).

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



Рис. 3.3: Файл с записанным текстом

По окончании записи файла мы сохранили его с помощью комбинации клавиш (рис. 3.4).

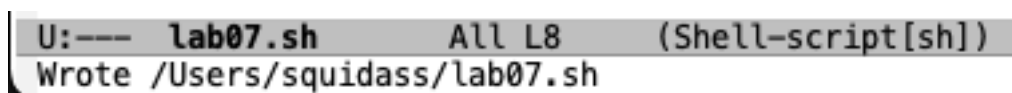


Рис. 3.4: Сохранение файла

Далее шли упражнения по стандартным процедурам редактирования.

Мы вырезали одной командой целую строку (рис. 3.5) и вставили ее в конец файла (рис. 3.6).



Рис. 3.5: Вырезание строки



Рис. 3.6: Вставка вырезанной строки в конец файла

После мы выделили область текста (рис. 3.7), скопировали эту область в буфер обмена и вставили ее в конец файла (рис. 3.8).



Рис. 3.7: Выделение области текста



Рис. 3.8: Вставка выделенной области в конец файла

В конце мы вновь выделили эту область (рис. 3.9) и вырезали ее (рис. 3.10).



Рис. 3.9: Выделение области

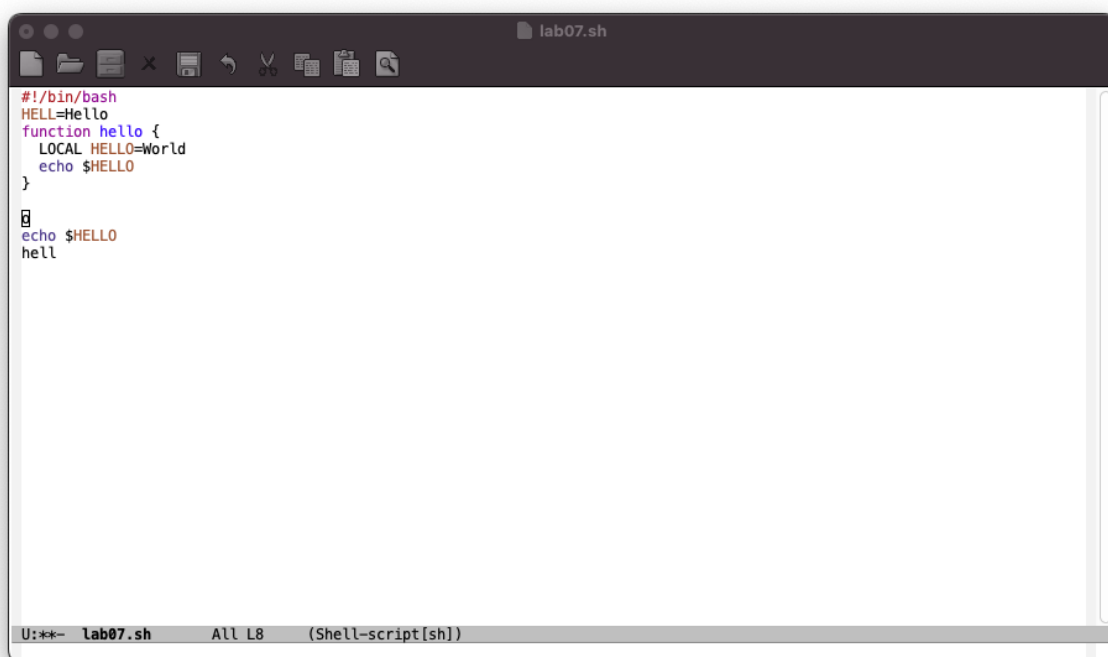
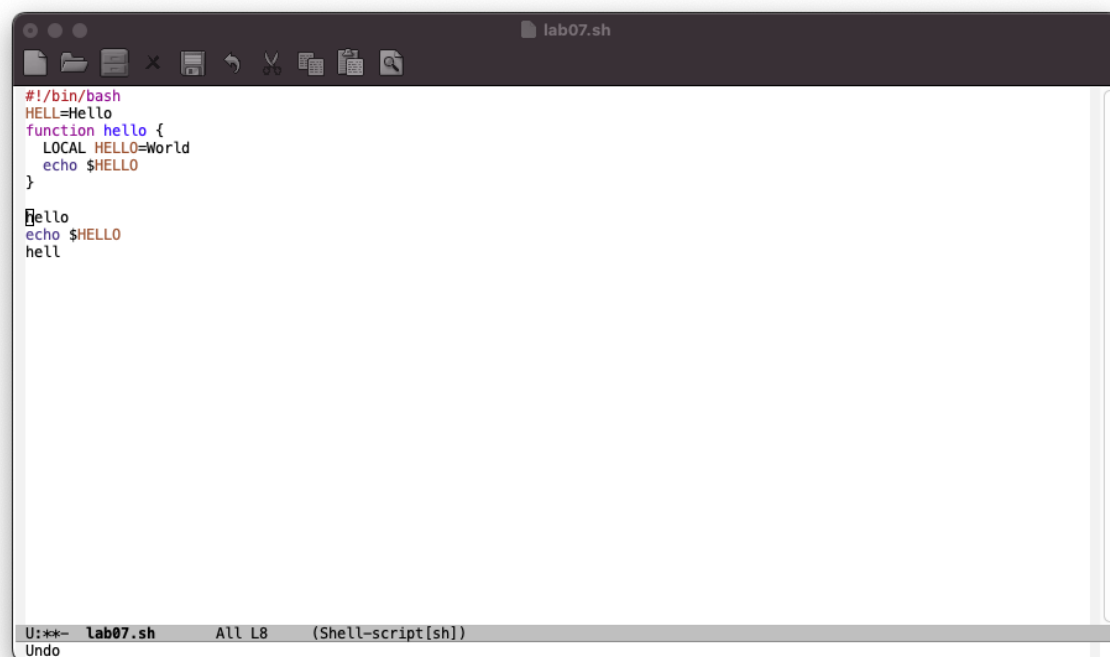


Рис. 3.10: Вырезание выделенной области

Последнее действие мы отменили (рис. 3.11).



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

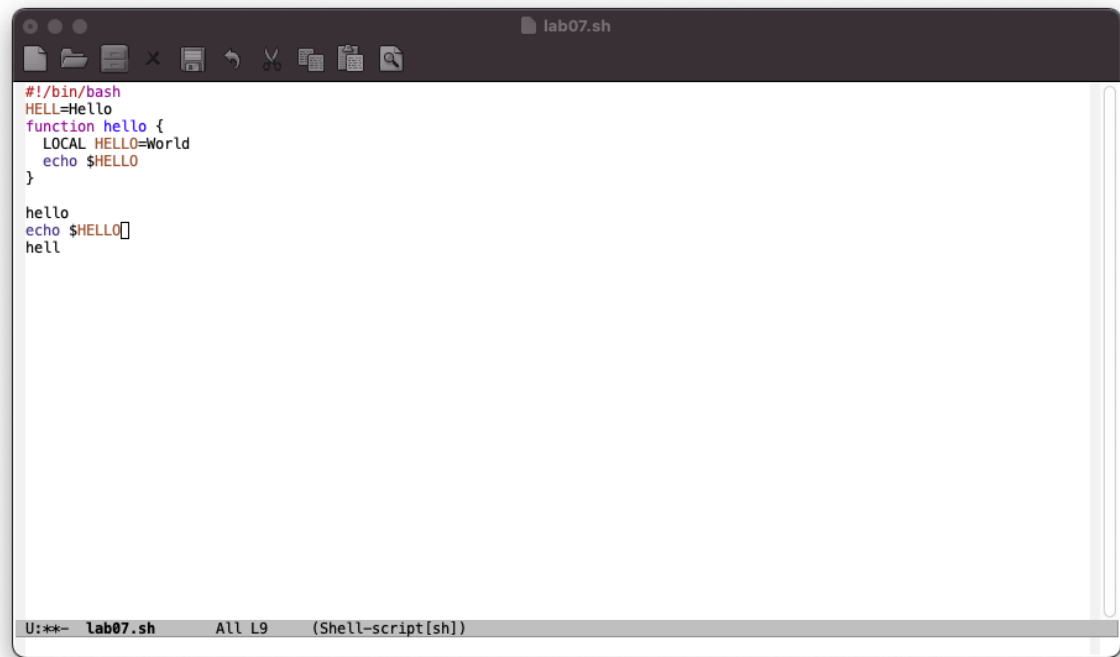
Hello
echo $HELLO
hell
```

U:*** lab07.sh All L8 (Shell-script[sh])
Undo

Рис. 3.11: Отмена последнего действия

После шли упражнения по перемещению курсора.

Сначала мы переместили курсор в начало строки, после этого в конец строки (рис. 3.12).



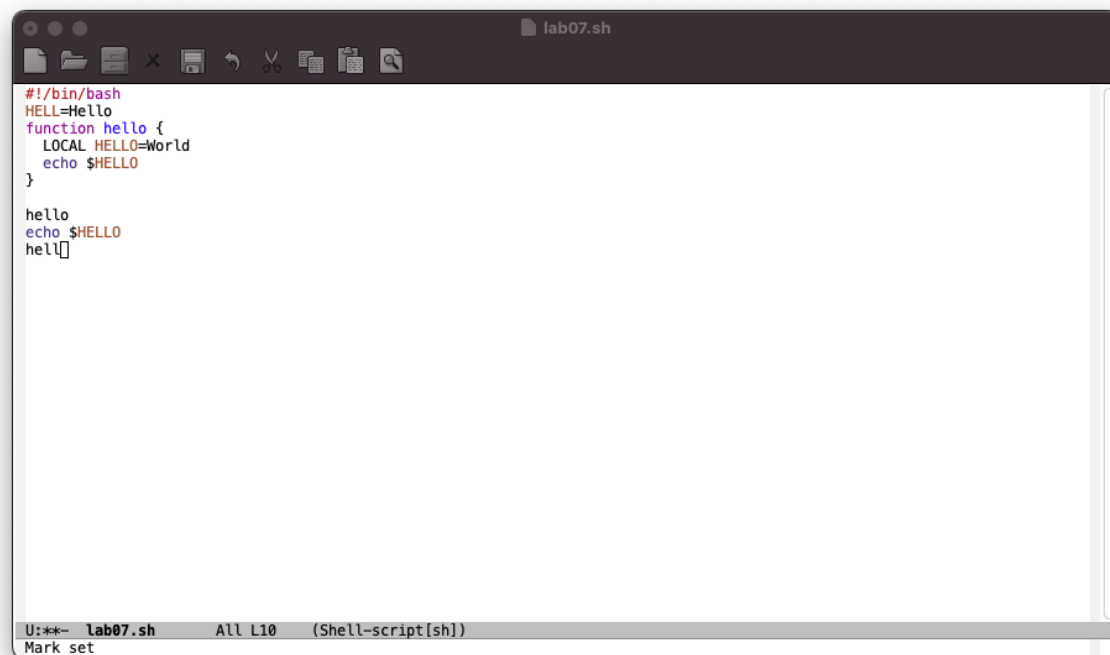
```
#!/bin/bash
HELL=Hello
function hello {
    LOCAL HELLO=World
    echo $HELLO
}

hello
echo $HELLO
hell
```

U:*** lab07.sh All L9 (Shell-script[sh])

Рис. 3.12: Перемещение курсора

А также переместили курсор в начало буфера и в конец (рис. 3.13).



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

hello
echo $HELLO
hell[]
```

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

Рис. 3.13: Перемещение курсора

Следующими шли задания по управлению буферами.

Первым делом мы вывели список активных буферов на экран (рис. 3.14).

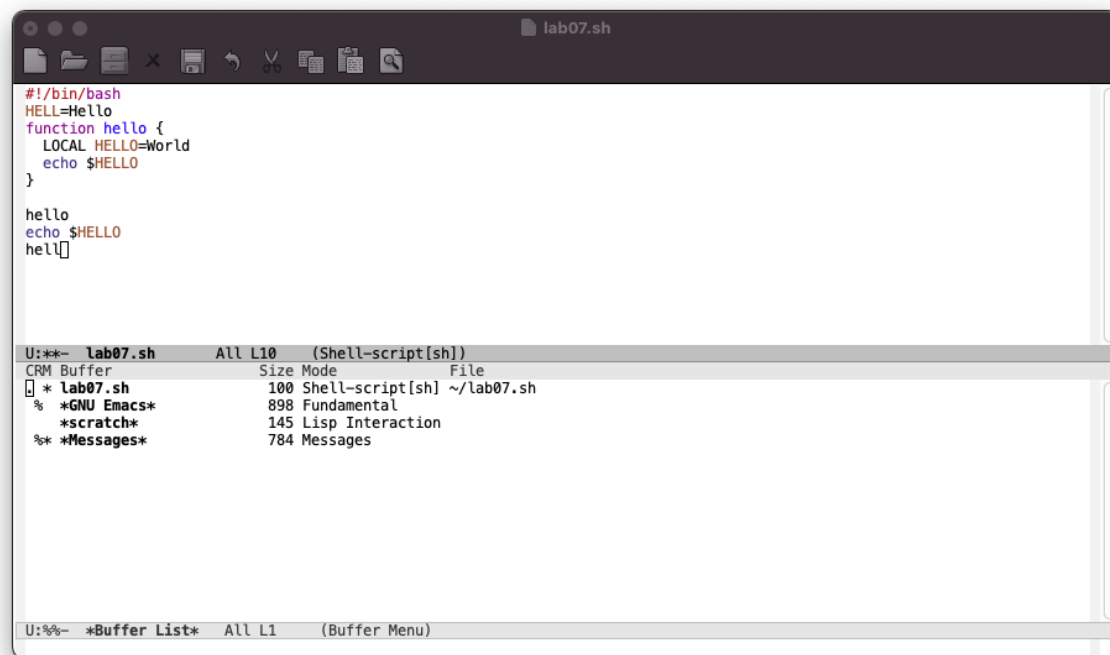


Рис. 3.14: Вывод списка активных буферов на экран

Далее сочетанием клавиш переместились на открытое окно со списком открытых буферов и в нем переключились на другой буфер (рис. 3.15).

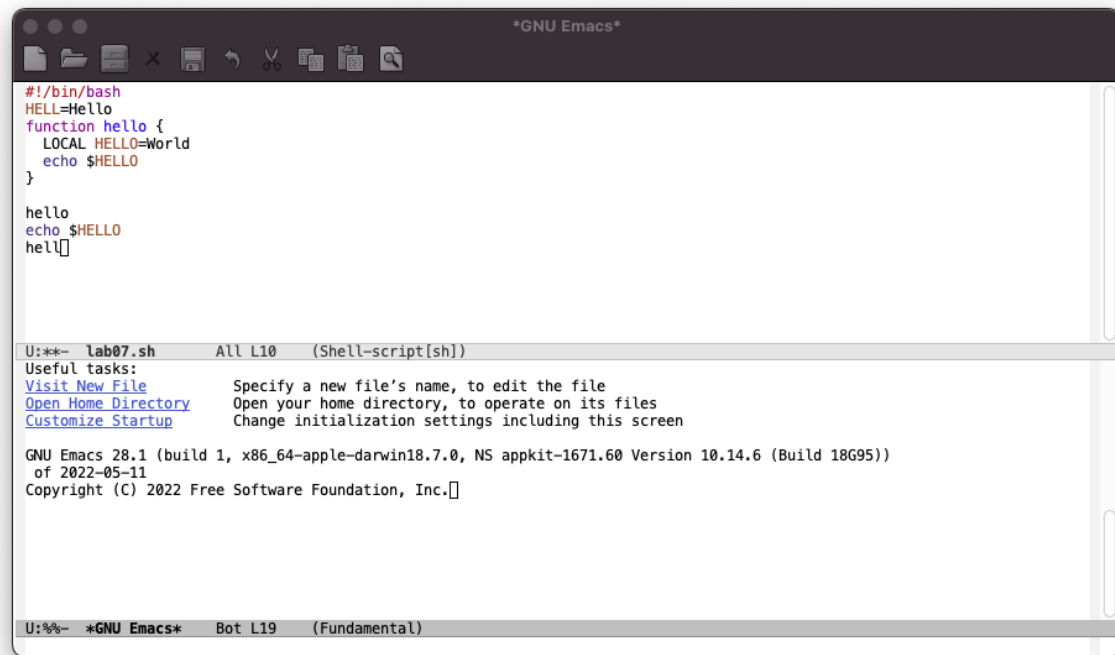


Рис. 3.15: Перемещение в новый буфер

После этого мы закрыли окно с только что открытым буфером (рис. 3.16).

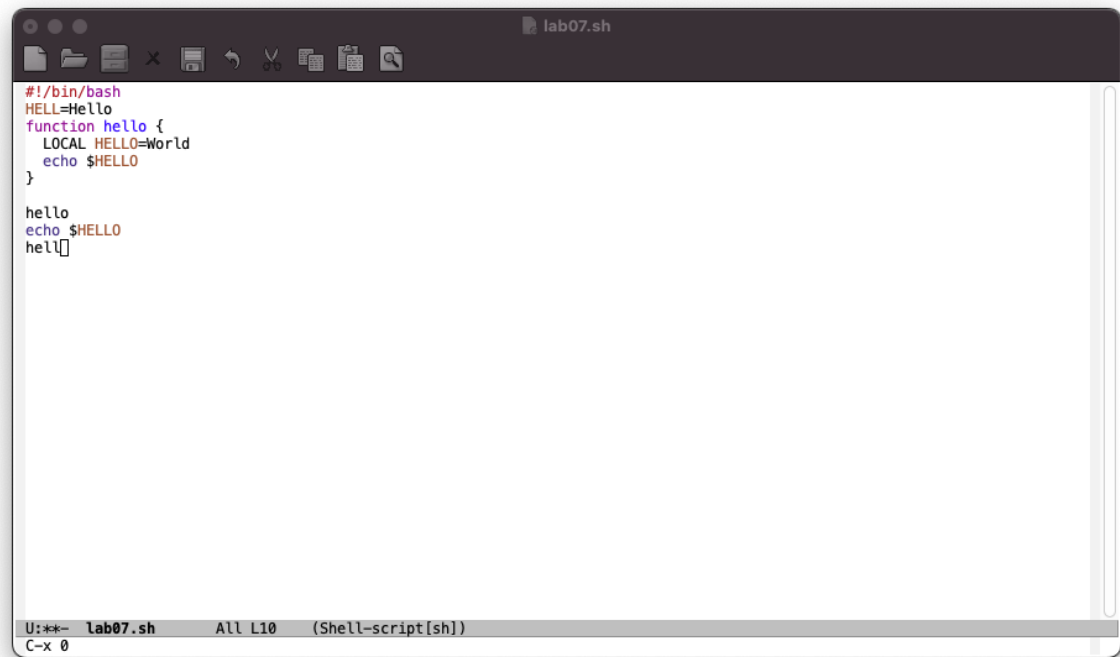


Рис. 3.16: Закрытие окна с буфером

В конце мы также переключились между буферами, однако теперь без использования окна, а только используя сочетание клавиш (рис. 3.17).

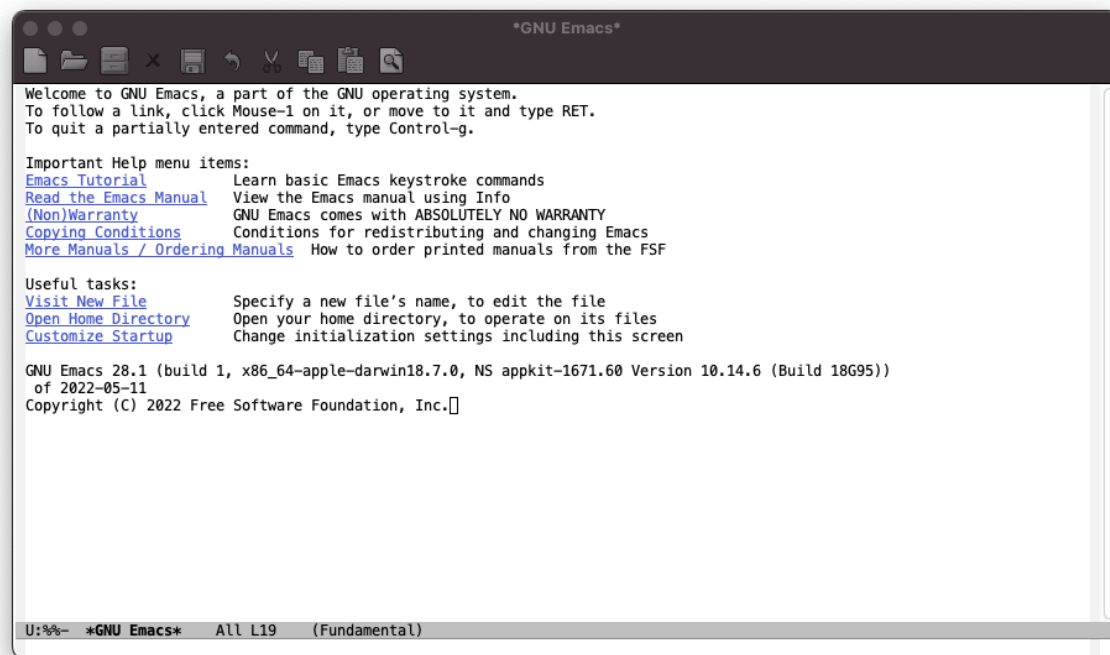


Рис. 3.17: Перемещение в буфер

Предпоследним блоком заданий были задания по управлению окнами.

Первым делом сочетаниями клавиш мы поделили фрейм на 4-е части: два окна по вертикали и два по горизонтали (рис. 3.18).

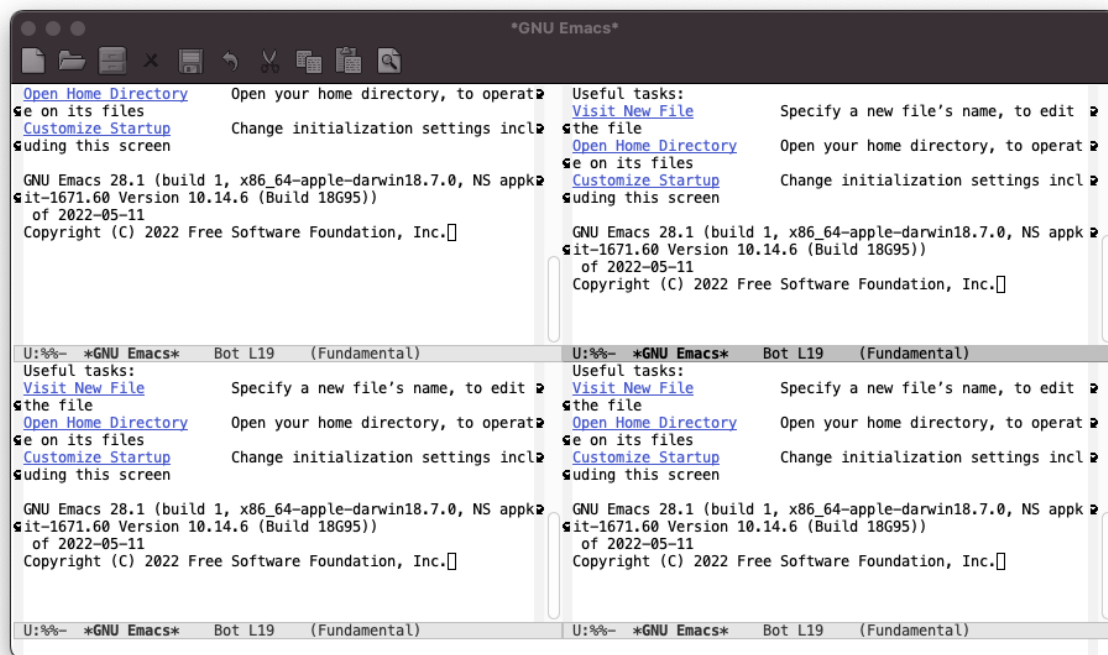


Рис. 3.18: Деление фрейма на 4-е части

После этого в каждом из созданных окон мы создали новый файл и напечатали в нем несколько строк текста (рис. 3.19).

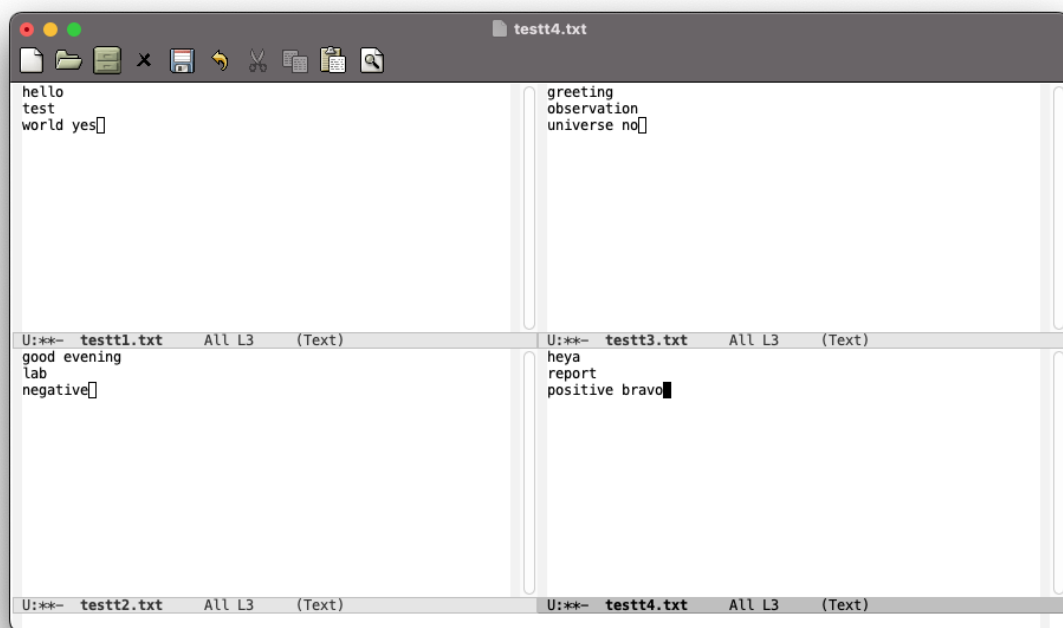


Рис. 3.19: Напечатанные текста в файлах

Последним блоком заданий были задания связанные с режимом поиска.

Для начала мы переключились в обычный режим поиска и нашли несколько слов в тексте (рис. 3.20). По результатам можно было переключаться сочетанием клавиш.



Рис. 3.20: Нахождение вхождений в тексте

Далее мы вышли из этого режима поиска и перешли в режим замены текста (рис. 3.21).

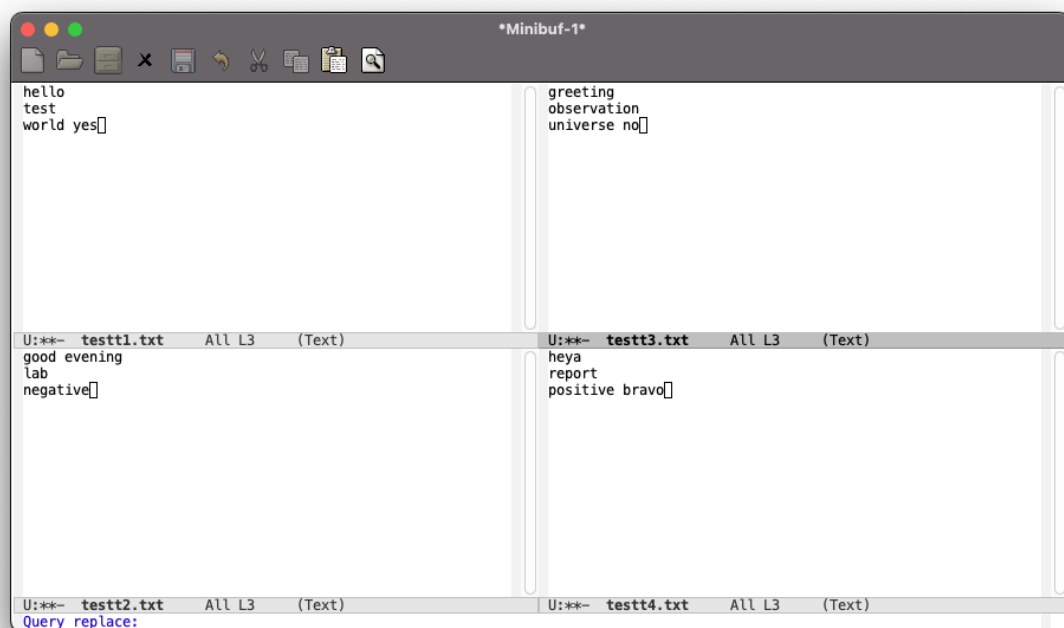


Рис. 3.21: Режим замены текста

В этом режиме первым делом мы ввели текст, который следует найти и заменить, нажали Enter и затем ввели текст для замены (рис. 3.22).

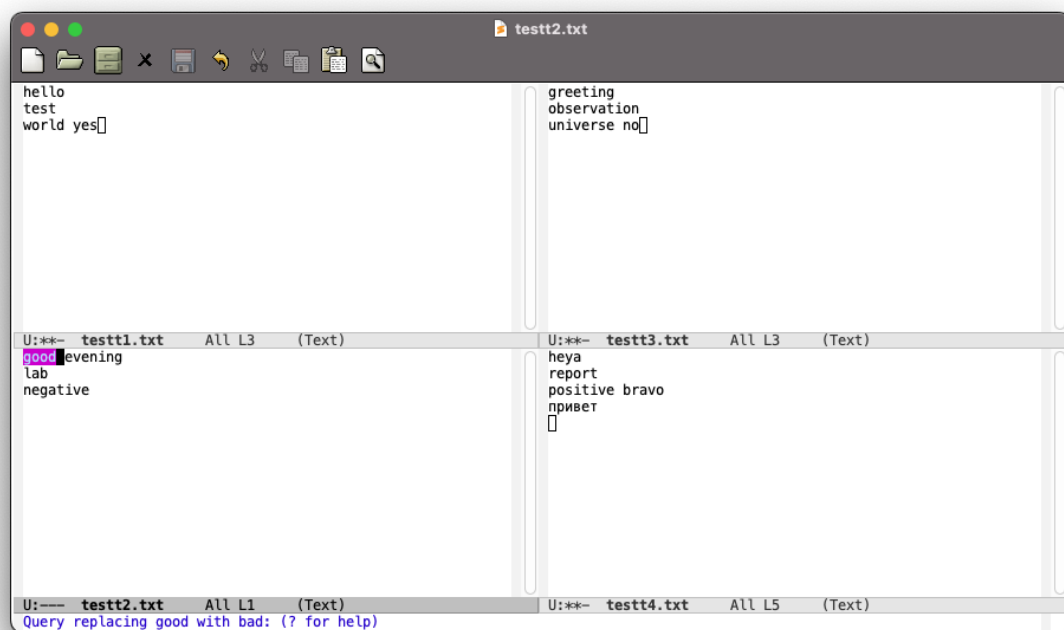


Рис. 3.22: Предложение замены

В буфере подсветился текст для замены и для подтверждения замены мы нажали ! (рис. 3.23).

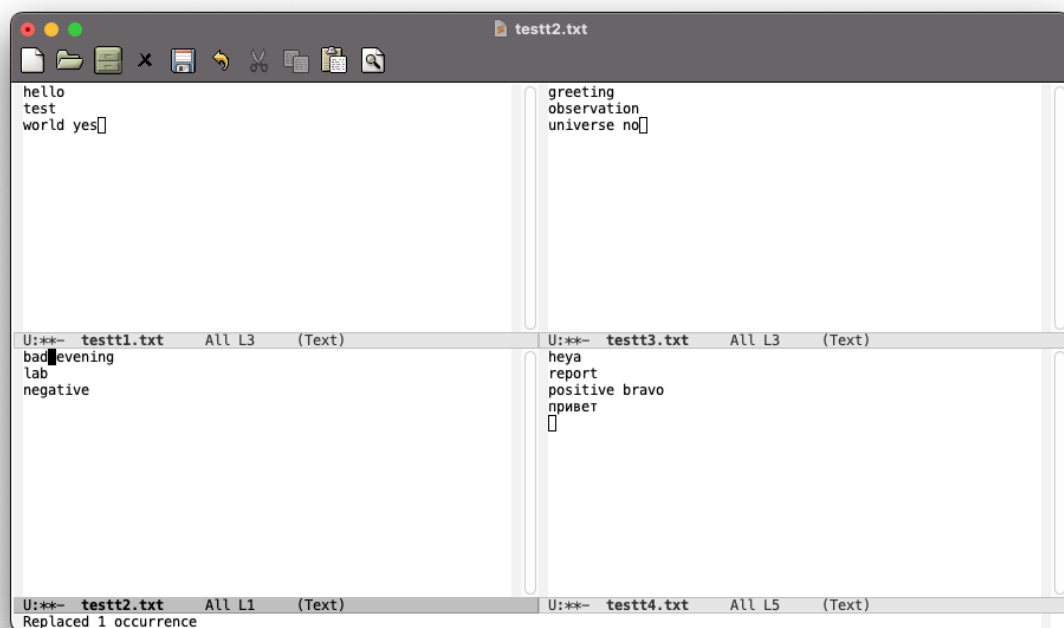


Рис. 3.23: Подтверждение замены

В конце мы испробовали другой режим поиска, нажав M-s o (рис. 3.24).

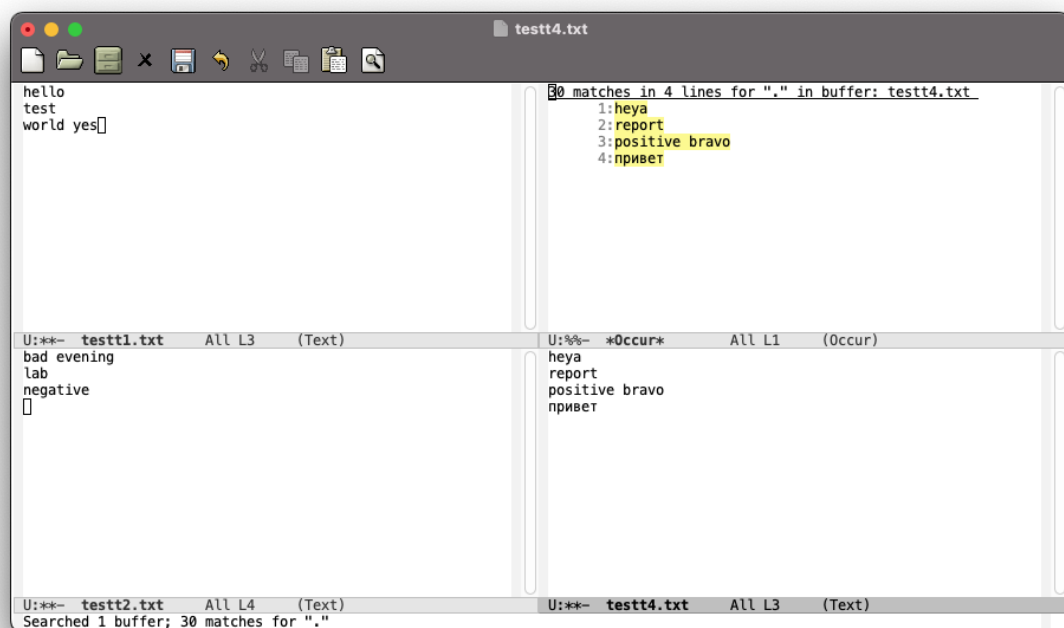


Рис. 3.24: Использование режима поиска по регулярным выражениям

Данный режим является режимом поиска с использованием регулярных выражений.

4 Выводы

По выполнении лабораторной работы мы получили практические навыки работы с редактором emacs, а также немного дополнительно познакомились с операционной системой Linux.

5 Ответы на контрольные вопросы

1. Редактор `emacs` представляет из себя графический (GUI) редактор текстовых файлов с особенностями буферов, фреймов и окон. Они позволяют ускорить и усовершенствовать работу с текстовыми файлами при их эффективном использовании.
2. Те же особенности, которые делают его мощным (система буферов, фреймов и окон) делают его сложным для освоения новичкам. Также сложности добавляет взаимодействие, которое больше углублено в использование горячих клавиш.
3.
 - Буфер – это некий объект, в котором содержится текст. Он может быть как буфером обмена, который чаще всего скрыт, так и буфером рабочим, который отображает файл на устройстве.
 - Окно – не окно приложения в обычном его понимании. В терминологии `emacs` окно – это область фрейма (окна в обычном понимании), которая отображает содержимое буфера.
4. Нет, нельзя, так как окно – это область, “отображающая **один** из буферов”.
5.
 - *GNU Emacs* – содержит в себе “встречный текст” с ссылками на tutorиалы и мануалы.
 - *scratch* – буфер для текста, который еще не сохранен и для обработки Lisp
 - *Messages* – история выведенных сообщений в нижней части фрейма

6.

- C-c | – CtrlShift</kbd>
- C-c C-| – CtrlCtrlShift</kbd>

- 7.
- C-x 2 – для деления по горизонтали
 - C-x 3 – для деления по вертикали

8. Конфигурационный файл находится по пути `~/.config/emacs`.

9. В редакторе клавиша Backspace имеет стандартное поведения удаления предыдущего символа. Однако действие по нажатию клавиши можно пере- назначить в конфигурации emacs.

10. Мне больше понравился редактор vi (vim), т.к. emacs для меня не обладает какими-либо особенностями, которые оправдали бы установку дополни- тельного GUI приложения и уж тем более переход на его использование. Vi (Vim) встроен в большинстве дистрибутивов и позволяет добиться высокой эффективности.