

# **Лабораторная работа №10**

**Российский университет дружбы народов**

Тимур Андреевич Дарижапов

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
<b>4</b>	<b>Выводы</b>	<b>28</b>
<b>5</b>	<b>Ответы на контрольные вопросы</b>	<b>29</b>

## **Список таблиц**

# Список иллюстраций

3.1	Открытие . . . . .	7
3.2	Создание файла . . . . .	7
3.3	Текст . . . . .	8
3.4	Вырезание . . . . .	9
3.5	Вставка . . . . .	10
3.6	Выделение . . . . .	11
3.7	Вставка в конец . . . . .	12
3.8	Вырезание области . . . . .	13
3.9	Отмена действия . . . . .	14
3.10	Перемещение в начало . . . . .	15
3.11	Перемещение в конец . . . . .	16
3.12	Перемещение в начало буфера . . . . .	17
3.13	Перемещение в конец буфера . . . . .	18
3.14	Активные буферы . . . . .	19
3.15	Перемещение в окно . . . . .	19
3.16	Перемещение в другой буфер . . . . .	20
3.17	Переключение без списка . . . . .	20
3.18	Переключение . . . . .	20
3.19	Фрейм на 4 части . . . . .	21
3.20	Новые буферы . . . . .	22
3.21	Режим поиска . . . . .	23
3.22	Переключение . . . . .	24
3.23	Выход из режима поиска . . . . .	25
3.24	Режим поиска и замены . . . . .	26
3.25	Другой режим поиска . . . . .	27

# 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 etacs.
3. Выполнить упражнения.
4. Ответить на контрольные вопросы.

## 3 Выполнение лабораторной работы

1.Открываем emacs(Рисунок 3.1).

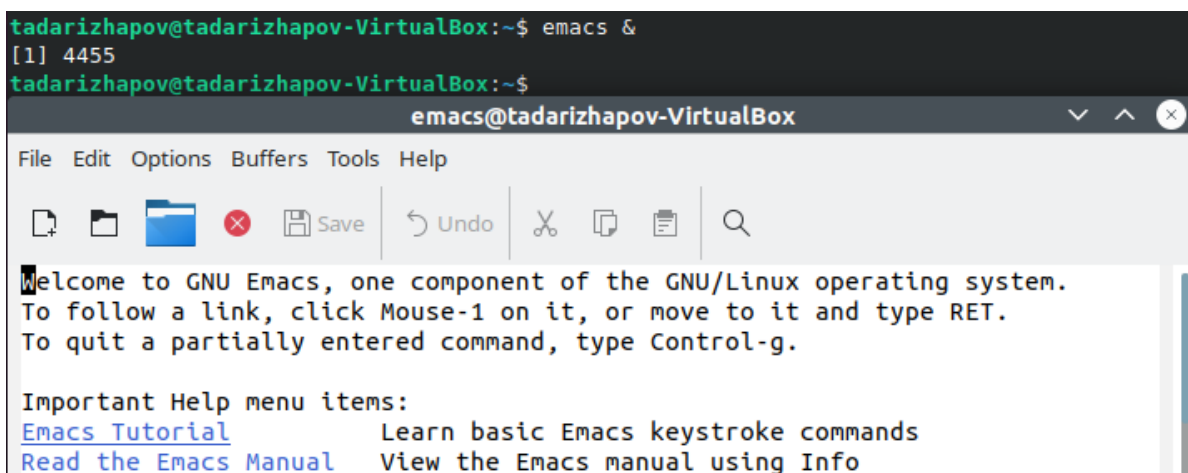


Рис. 3.1: Открытие

2.Создаём файл lab07.sh с помощью комбинации Ctrl-x и Ctrl-f(Рисунок 3.2).

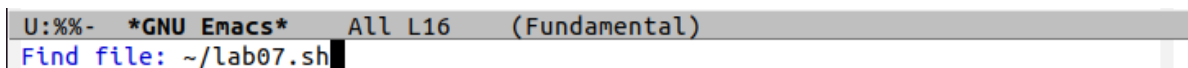


Рис. 3.2: Создание файла

3.Набираем текст(Рисунок 3.3).

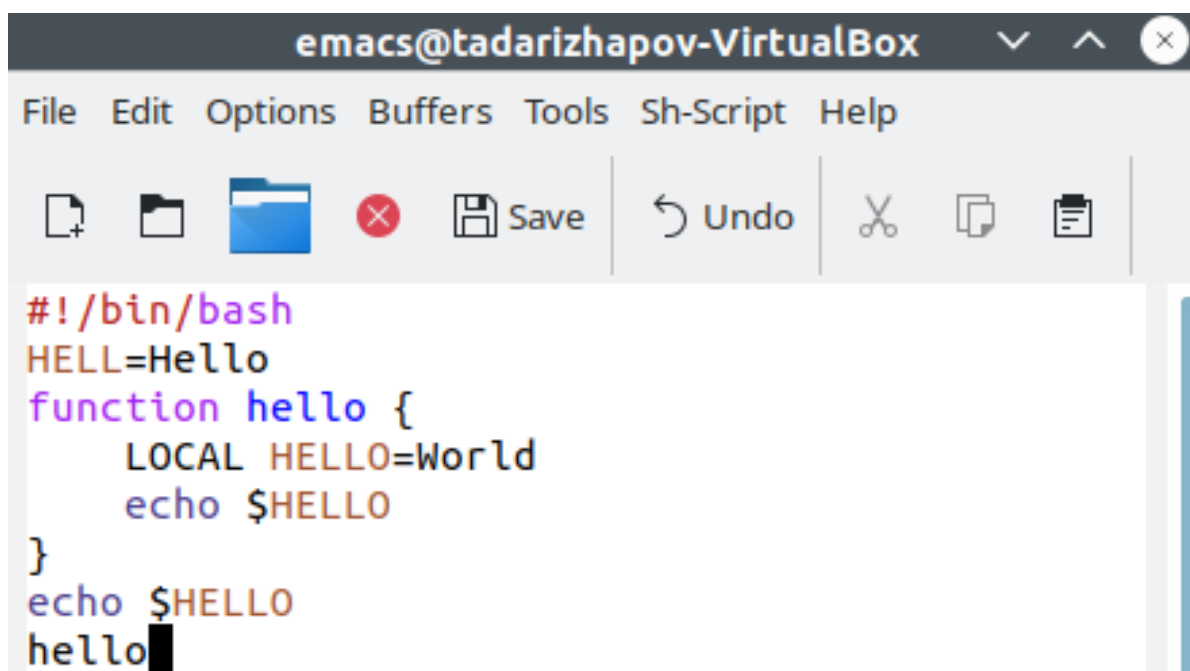


Рис. 3.3: Текст

4. Сохраняем файл с помощью комбинации Ctrl-x Ctrl-s.
5. Проделываем с текстом стандартные процедуры редактирования. (Примечание: клавиша Ctrl обозначена как C, а клавиша Alt обозначена как A).
- 5.1. Вырезаем одной командой целую строку (C-k) (Рисунок 3.4).



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

Рис. 3.4: Вырезание

5.2. Вставляем эту строку в конец файла (C-y) (Рисунок 3.5).

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

Рис. 3.5: Вставка

5.3.Выделяем область текста (C-space)(Рисунок 3.6).

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

Рис. 3.6: Выделение

5.4.Копируем область в буфер обмена (A-w).

5.5.Вставляем область в конец файла(Рисунок 3.7).

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

Рис. 3.7: Вставка в конец

5.6. Выделяем эту область второй раз и на этот раз вырезаем её (C-w) (Рисунок 3.8).

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

    echo $HELLO
```

Рис. 3.8: Вырезание области

5.7.Отменяем последнее действие (C-/)(Рисунок 3.9).

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

Рис. 3.9: Отмена действия

6. Учимся использовать команды по перемещению курсора.

6.1. Перемещаем курсор в начало строки (C-a) (Рисунок 3.10).

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

Рис. 3.10: Перемещение в начало

6.2.Перемещаем курсор в конец строки (C-e)(Рисунок 3.11).

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

Рис. 3.11: Перемещение в конец

6.3.Перемещаем курсор в начало буфера (A-<)(Рисунок 3.12).



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

Рис. 3.12: Перемещение в начало буфера

6.4.Перемещаем курсор в конец буфера (A->)(Рисунок 3.13).

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

Рис. 3.13: Перемещение в конец буфера

7.Учимся управлять буферами.

7.1.Выводим список активных буферов на экран (С-х С-b)(Рисунок 3.14).

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

-:--- lab07.sh Top L1 (Shell-script[bash])				
CRM	Buffer	Size	Mode	File
	lab07.sh	131	Shell-script[... ~/lab07.sh	
%	*GNU Emacs*	898	Fundamental	
	*scratch*	145	Lisp Interaction	
%*	*Messages*	1076	Messages	

Рис. 3.14: Активные буферы

7.2.Перемещаемся во вновь открытое окно (С-х о) со списком открытых буферов и переключаемся на другой буфер(Рисунок 3.15, 3.16).

-:--- lab07.sh Top L1 (Shell-script[bash])				
CRM	Buffer	Size	Mode	File
.	lab07.sh	131	Shell-script[... ~/lab07.sh	
%	*GNU Emacs*	898	Fundamental	
█	*scratch*	145	Lisp Interaction	
%*	*Messages*	1076	Messages	

Рис. 3.15: Перемещение в окно

```
-:--- lab07.sh      Top L4      (Shell-script[bash])
Loading /etc/emacs/site-start.d/50latexmk.el (source)...done
Loading /etc/emacs/site-start.d/50texlive-lang-english.el (source)...done
For information about GNU Emacs and the GNU system, type C-h C-a.
Setting up indent for shell type bash
Indentation variables are now local.
Indentation setup for shell type bash
█

U:%*-  *Messages*      Bot L19      (Messages)
```

Рис. 3.16: Перемещение в другой буфер

7.3. Закрываем это окно (C-x 0).

7.4. Теперь вновь переключаемся между буферами, но уже без вывода их списка на экран (C-x b) (Рисунок 3.17, 3.18).

```
-:--- lab07.sh      All L1      (Shell-script[bash])
Switch to buffer (default *GNU Emacs*): *scratch*
```

Рис. 3.17: Переключение без списка

```
;; This buffer is for text that is not saved, and for Lisp
evaluation.
;; To create a file, visit it with C-x C-f and enter text
in its buffer.
█
```

Рис. 3.18: Переключение

8. Учимся управлять окнами.

8.1. Поделим фрейм на 4 части: разделим фрейм на два окна по вертикали (C-x 3), а затем каждое из этих окон на две части по горизонтали (C-x 2) (Рисунок 3.19).



Рис. 3.19: Фрейм на 4 части

8.2. В каждом из четырёх созданных окон откроем новый буфер (файл) и введём несколько строк текста (Рисунок 3.20).

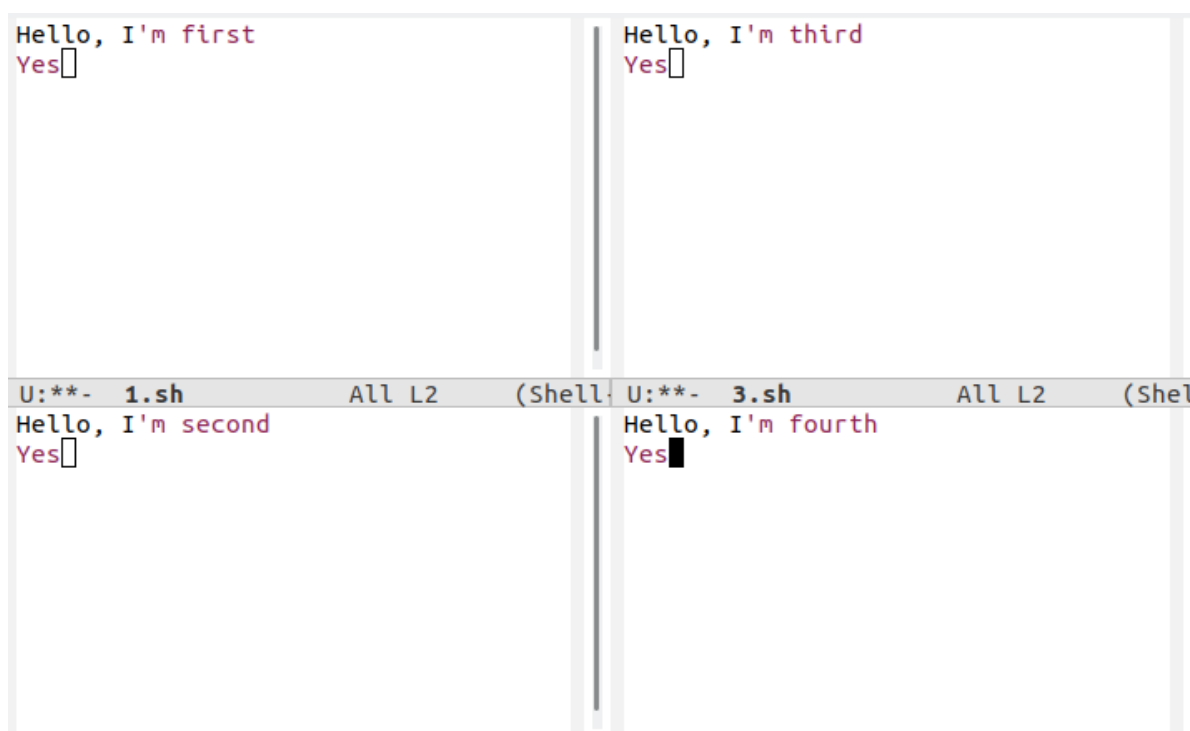


Рис. 3.20: Новые буферы

9.Учимся режиму поиска.

9.1.Переключаемся в режим поиска (C-s) и находим несколько слов, присутствующих в тексте(Рисунок 3.21).

```
Hello, I'm third
Yes
My name is Timur
I'm from Russia
```

```
U:** - 3.sh All L4 (Shell-script[sh] Isearch)
```

```
user-error: No resizable window below this one [3 times]
```

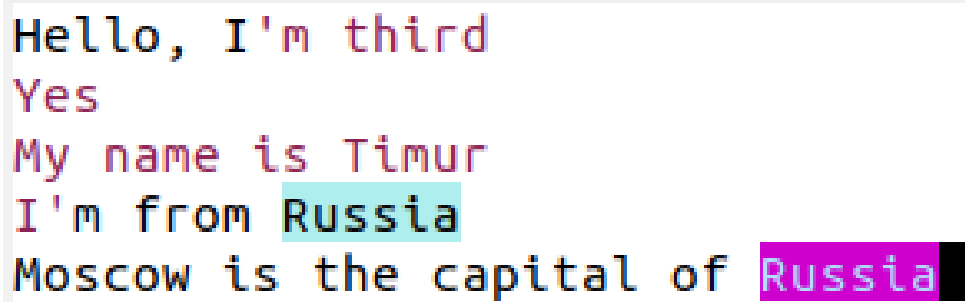
```
user-error: No resizable window below this one
```

```
U:%* - *Messages* Bot L45 (Messages)
```

```
Failing I-search: Russia
```

Рис. 3.21: Режим поиска

9.2.Переключаемся между результатами поиска, нажимая C-s(Рисунок 3.22).



```
Hello, I'm third
Yes
My name is Timur
I'm from Russia
Moscow is the capital of Russia
```

Рис. 3.22: Переключение

9.3. Выходим из режима поиска, нажав C-g (Рисунок 3.23).



```
Hello, I'm third  
Yes  
My name is Timur  
I'm from Russia  
Moscow is the capital of Russia
```



```
U:** - 3.sh All L7 (Shell-script[sh])  
Quit
```

Рис. 3.23: Выход из режима поиска

9.4. Переходим в режим поиска и замены (A-%), вводим текст, который следует найти и заменить, нажимаем Enter, затем вводим текст для замены. После того как будут подсвечены результаты поиска, нажимаем ! для подтверждения замены (Рисунок 3.24).

```
Hello, I'm third
No
My name is Timur
I'm from Russia
Moscow is the capital of Russia
```

```
U:**- 3.sh Bot L555429 (Shell-script[sh])
Query replace (default Yes → No): third
```

Рис. 3.24: Режим поиска и замены

9.5. Испробуем другой режим поиска, нажав A-s o. 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 (Рисунок 3.25).

```
Hello, I'm third  
No  
My name is Timur  
I'm from Russia  
Moscow is the capital of Russia
```

```
U:**- 3.sh Bot L555438 (Shell-script[sh])
```

```
1 match for "Moscow" in buffer: 3.sh  
555432: Moscow is the capital of Russia
```

Рис. 3.25: Другой режим поиска

## 4 Выводы

Я познакомился с операционной системой Linux. Я получил 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); «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 «» (стрелочка) удаляет символ перед курсором, но в редакторе её можно переназначить. Для этого необходимо изменить конфигурацию файла .emacs.

10. Более удобным я считаю редактор emacs, нежели vi, 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.