

Презентация по лабораторной работе №11

Текстовый редактор emacs

Мутале Ч.

21 Апреля 2025

Российский университет дружбы народов, Москва, Россия

Информация



- Мутале Чали
- студент НКА 05-24
- факультет физико-математических и естественных наук
- Российский университет дружбы народов
- 1032239667@rudn.ru
- <https://cmutale-skept.github.io/ru/>

.....
.....

Цель работы

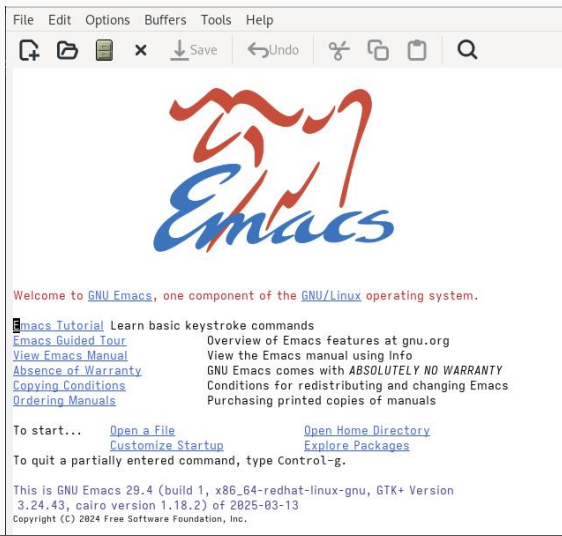
Получить практические навыки работы с редактором Emacs.

Для данной работы, мне надо был установить Emacs:

```
cmutale@cmutale:~$ sudo dnf install emacs  
Updating and loading repositories:  
Copr repo for gitflow owned by elegos
```

Рис. 1: Установка Emacs

Выполнив Emacs в командной строке, я открыл текстовый редактор:



С помощью комбинации Ctrl-x Ctrl-f, создал файл lab07.sh:



Рис. 3: Созданный файл

Основные команды с редактором

Я сохранил файл с помощью комбинации Ctrl-x Ctrl-s:

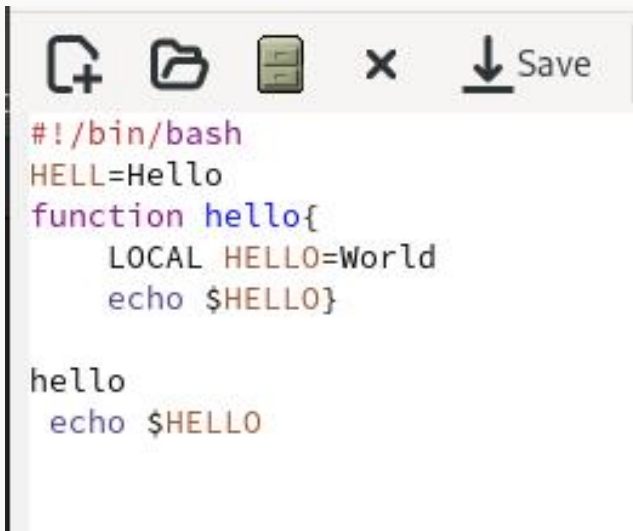


The image shows a code editor window with a toolbar at the top. The toolbar contains icons for opening a file, saving a file, and closing a file, followed by a close button (X). To the right of these icons are two buttons: 'Save' with a downward arrow icon and 'Undo' with a curved arrow icon. Below the toolbar, the editor displays a shell script with syntax highlighting. The script defines a function named 'hello' that prints 'World' and then prints the value of the 'HELLO' variable. The script ends with a call to the 'hello' function. A black cursor is visible at the end of the last line of the script.

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

Основные команды с редактором

Вырезать целую строку (C-k):

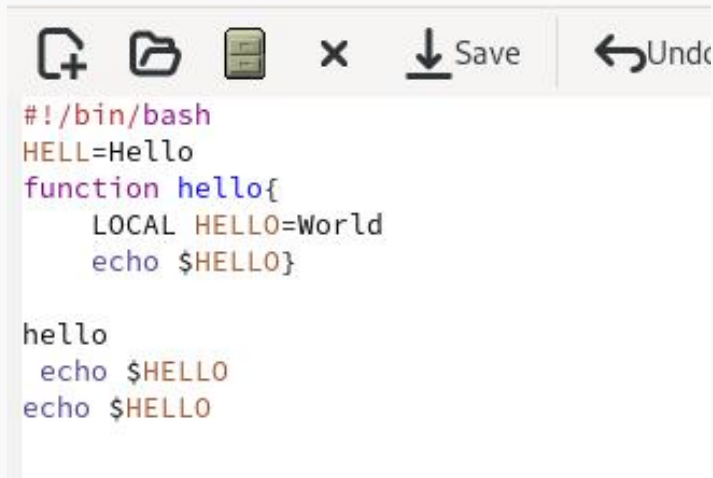


```
#!/bin/bash
HELL=Hello
function hello{
    LOCAL HELLO=World
    echo $HELLO}

hello
echo $HELLO
```

Основные команды с редактором

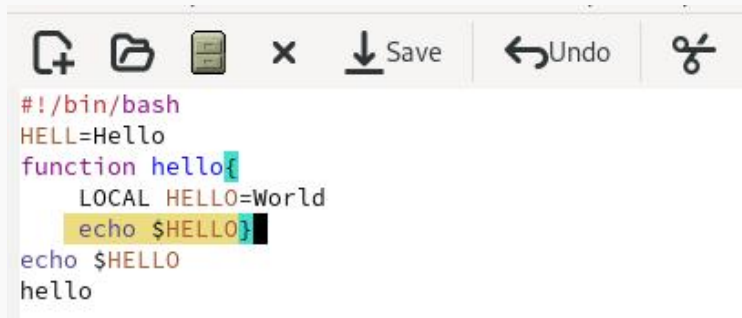
С помощью C-у вставил эту строку в конец файла:



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

hello
    echo $HELLO
echo $HELLO
```

Выделил область текста (C-space):



The screenshot shows a code editor window with a toolbar at the top containing icons for file operations (new, open, save, close) and editing (undo, redo). The code is a shell script with the following content:

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

A text selection is visible on the line `echo $HELLO}` within the `function hello{` block. The selection is highlighted in yellow, and the cursor is positioned at the end of the line.

Рис. 7: Выделенный текст


Скопировал область в буфер обмена (M-w) и вставила ее в конец файла:



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

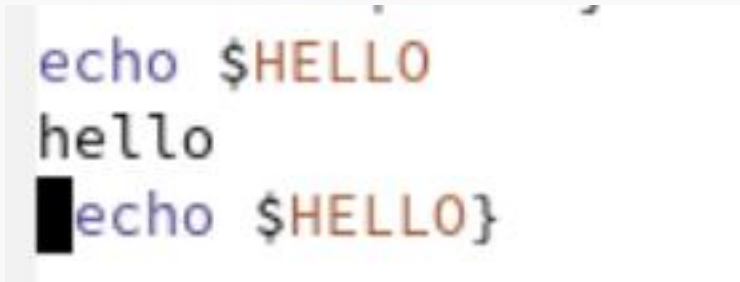
Рис. 8: копирование и вставка

Выделил эту же область и на этот раз вырезал её (C-w):



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

С помощью `C/` отменил последнее действие:



```
echo $HELLO  
hello  
[undo] echo $HELLO}
```

The screenshot shows a terminal window with a light gray background. The first line is the command 'echo \$HELLO' where 'echo' is in blue and '\$HELLO' is in red. The second line is the output 'hello'. The third line shows the undo operation: a black square icon followed by 'echo \$HELLO}' in the same color scheme as the first line. This indicates that the previous command has been restored to the input buffer.

Рис. 10: отмена действие

Основные команды с редактором

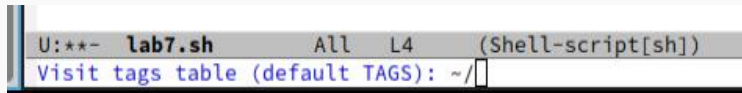
С помощью `C-a` можно переместить курсор в начало строки. С помощью `C-e` переместил курсор в конец строки:



The image shows a terminal editor window with a toolbar at the top. The toolbar contains icons for opening a file, saving a file, closing a file, a close button (X), a save button (floppy disk), an undo button (curved arrow), and a redo button (curved arrow with a plus sign). Below the toolbar, a shell script is displayed with syntax highlighting. The script defines a function named 'hello' that prints 'Hello World' and 'Hello'.

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

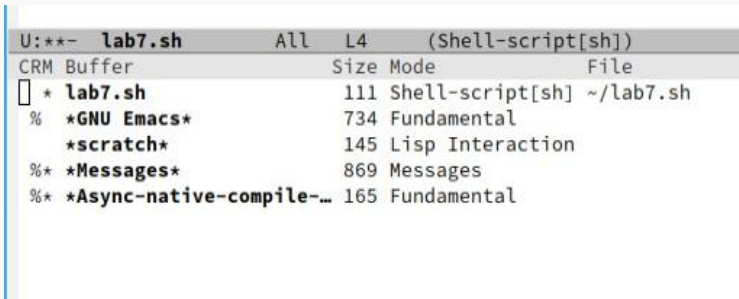

Переместил курсор в начало и конец буфера с помощью M-< и M-> соответственно:



```
U:***- lab7.sh All L4 (Shell-script[sh])
Visit tags table (default TAGS): ~/
```

Рис. 12: Перемещение курсор в буфере

Выводил список активных буферов на экран с помощью C-x C-b:

A screenshot of the Emacs buffer list window. The title bar at the top reads 'U: *~ lab7.sh All L4 (Shell-script[sh])'. The table below lists active buffers with columns for 'CRM Buffer', 'Size', 'Mode', and 'File'. The first buffer is '* lab7.sh' with size 111, mode 'Shell-script[sh]', and file '~/lab7.sh'. It is selected with a cursor. Other buffers include '*GNU Emacs*' (size 734, mode 'Fundamental'), '*scratch*' (size 145, mode 'Lisp Interaction'), '%* *Messages*' (size 869, mode 'Messages'), and '%* *Async-native-compile-...' (size 165, mode 'Fundamental').

U: *~ lab7.sh All L4 (Shell-script[sh])			
CRM Buffer	Size	Mode	File
<input checked="" type="checkbox"/> * lab7.sh	111	Shell-script[sh]	~/lab7.sh
% *GNU Emacs*	734	Fundamental	
scratch	145	Lisp Interaction	
%* *Messages*	869	Messages	
%* *Async-native-compile-...	165	Fundamental	

Рис. 13: Активные буферы

С помощью C-x o переместился во вновь открытое окно со списком открытых буферов и переключился на другой буфер:

```
-:--- lab7.sh All L9 (Shell-script[bash])
CRM Buffer Size Mode File
[.] lab7.sh 111 Shell-script[ba... ~/lab7.sh
% *GNU Emacs* 734 Fundamental
*scratch* 145 Lisp Interaction
%* *Messages* 552 Messages
%* *Async-native-compile-... 165 Fundamental
```

С помощью C-x 0 закрыл окно со списком открытых буферов:

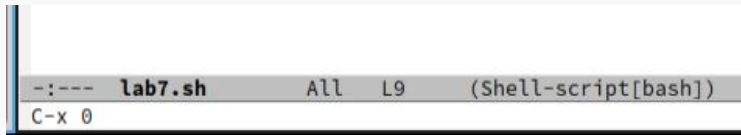
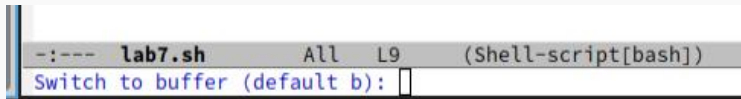


Рис. 15: Закрытие окно

Без вывода списка буферов, я переключился между буферами:

A screenshot of a terminal window. The top bar shows the prompt `-:--- lab7.sh`, the user `All`, the host `L9`, and the shell `(Shell-script[bash])`. Below the prompt, the command `Switch to buffer (default b):` is entered, followed by a cursor pointing to an empty input field.

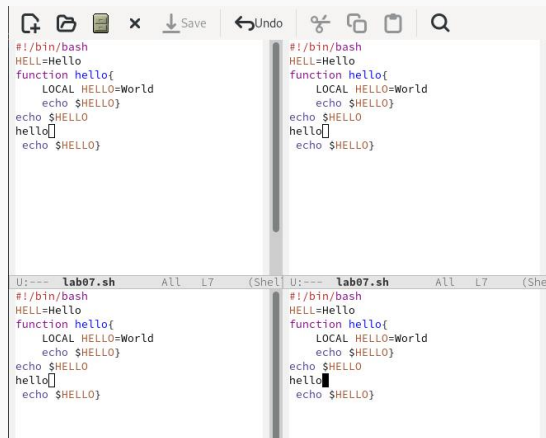
```
-:--- lab7.sh      All  L9      (Shell-script[bash])  
Switch to buffer (default b):
```

Рис. 16: Переключение между буферами



Рис. 17: Новый буфер

Поделил фрейм на 4 части. Сначала я разделил фрейм на два окна по вертикали (С-х 3), а затем каждое из этих окон на две части по горизонтали (С-х 2):



В каждом из четырёх созданных окон открыл новый буфер:

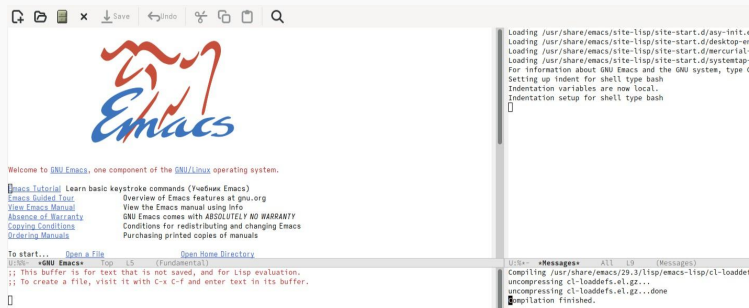


Рис. 19: Новые буферы

Переключился в режим поиска (C-s) и искал Indent:

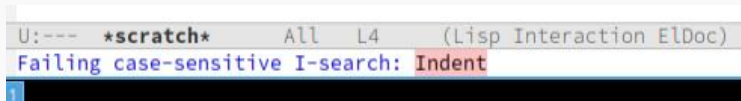


Рис. 20: Режим поиска

Переключался между результатами поиска, нажимая C-s и вышел из режима поиска, нажав C-g:

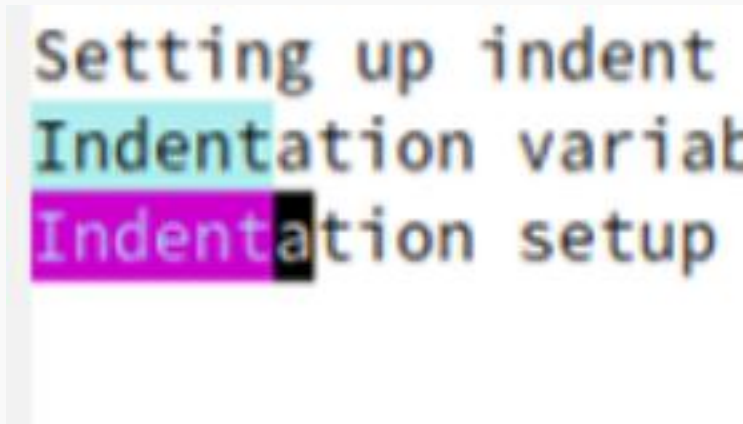


Рис. 21: Переключение между результатами

Перешёл в режим поиска и замены (M-% или M-x query replace):

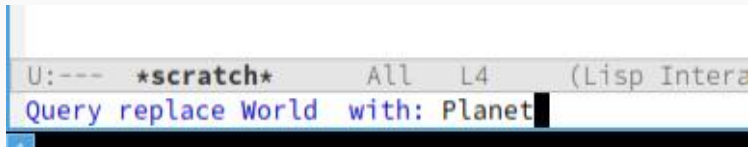


Рис. 22: Режим поиска

Нажав M-s o, я использовал другой режим поиска. Он отличается от предыдущего тем, что выводит результаты поиска в новом окне:

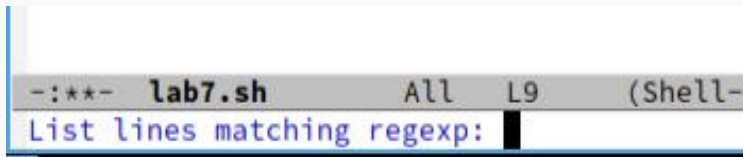



Рис. 23: другой режим поиска



```
6 matches for "hello" in buffer: lab7.sh
2:HELL=Hello
3:function hello{
4:    LOCAL HELLO=Planet
5:    echo $HELLO}
7:hello
8: echo $HELLO
```

Рис. 24: Результаты поиска

Выводы

При выполнении данной работы я получил практические навыки работы с Emacs.

СПАСИБО ЗА ВНИМАНИЕ
