LW-03. Linux/UNIX Shell Environment Variables.

1. Target.

- Learn to use environment variables and assign new values to them;
- Consolidate skills in the use of elementary commands and utilities of UNIX/Linux;
- Get the concept of initialization program files using the shell example;
- Acquire skills in creating and debugging shell configuration files.

The Linux/UNIX commands are used:

```
sudo, adduser, userdel, set, unset, readonly, export, env, more, less, man, ls, echo, pwd, wc, who,
date, ...
```

2. Assignments.

NOTE.

Start Your **UbuntuMini** Virtual Machine on your VirtualBox. You need only Linux Terminal to complete the lab tasks.
Before completing the tasks, make a SnapShot of your Virtual Linux. If there are problems, you can easily go back to working condition!

2.1. Shell Environment Variables Analyze. (Fill in a Table)

2.2. Make Example bash config file. (Make a config-file Screenshot)

2.3. Generate Your Task Variant Nr. (Make a Task Text Copy)

2.4. Change 3 Shell Config Files (according to your variant). (Make 3 Text Copy of config-files)

3. Report.

The report is provided electronic form with Report Blank Form (use a docx).

Report for Lab Work 03: Linux/UNIX Shell Environment Variables

Student Name Surname	**Student ID (nV)**	**Date**

3.1. Insert Completing Shell Environment Variables Analyze Table,

3.2. Insert Screenshot with Your Config-file and working Clock,

3.3. Insert Text Copy of Task for Your Variant,

3.4. Insert Text Copy of Your Modification for 3 Configuration Files
with the necessary comments:

˜/.bash_profile,

˜/.bashrc,

˜/.bash_logout.

4. Guidelines.

4.1. Bash Environment Variables Analyze.

4.1.1. Create new User account for this Lab Work.

- Login as student account (**user with sudo permissions**).
- Create new user account, example stud. Use **adduser** command. (NOTE. You can use the command "userdel –rf stud" to delete stud account from your Linux.)

```
$ sudo adduser stud
```

- Logout from student account (**logout**) and login as stud.

4.1.2. Expansion (substitution) in shell.

Expansion is the process of analyzing the command line or shell-script line in order to find special notation in it and substitute corresponding values in their place.

To view the result of expansion of the typed string, press <Esc> + <Ctrl> + <E>. To return the previous line - <Ctrl> + <->. For example,

```
$ ls -l `echo ${HOME}/..`
```

interpreted as

```
$ ls -l /home/stud/..
```

The order of bash expansion:

1. Checking the syntax of the command, if there is an error, then the shell reports this and stops the execution of the command.
2. Substitution of the result of the work of `` `commands` `` that are taken in accent marks.
3. Substitution of $parameter values and $variables starting with dollar.
4. Arithmetic substitution of $((expression)) or $[expression].
5. Divide the string into lexemes by interpreting <Space>, <Tab>, <Enter>, which are not in "quotation" marks or not \escaped.
6. Generation of file names. Every word containing characters ? * [ ], if possible, expand to alphabetically ordered list of file names.
7. Removing all quotation marks except shielded slashes \".

4.1.3. Cancel interpretation of special characters.

If your string contains some special characters (metacharacters), such as <Space>, <Tab>, <Enter>, |, >, <, &, $ and others, then you may experience problems with their interpretation.

Metacharacters are used not only by the shell, but also by other programs, but the first program will be the shell, so the shell tries to interpret the metacharacters as “their own” and does not pass it on.

Screening methods (cancellation) of the special meaning of metacharacters

\c - backslash, cancels the special character c;

’- an apostrophe, in the line taken in apostrophes the value of all metacharacters is canceled;

”- quotation mark, the string enclosed in quotation marks cancels the value of all metacharacters except: \,`, ”, $.

For example, compare 4 examples:

```
$ echo $HOME; echo ’$HOME’
$ echo \$HOME; echo ”$HOME”
```

4.1.4. The concept of shell variables.

1. In any programming language, variables are used to short store values. All shell variables are stored as text.

2. To get the value of a variable, you need to use the $ sign in front of its name: $variable.

```
$ echo $LOGNAME
```

3a. The command to assign a value to shell variables has the syntax variable = value:

```
$ ux=UNIX; lx=Linux; ws=Windows
$ echo $ux
$ echo $ux$lx; echo $ux $lx; echo $ux\/$lx
```

3b. You can assign a variable the value of another variable variable1 = $variable2:

```
$ syst=$ux; echo $syst
```

3c. You can assign a value to a variable by generating a string from the command output stream, enclosing the command in accent marks: variable = `command`:

```
$ cdir=`pwd`
$ echo $cdir
/home/student/documents
```

3d. When assigning a variable to a value that contains spaces, punctuation, and newlines, enclose the values in quotation marks or apostrophes, or escape the specified characters with a backslash \:

```
$ VP=”Karl Marks”; echo $VP
$ VP=Karl\ Marks; echo $VP
```

4. Variables are deleted with the unset command or by assigning a new empty value:

```
$ unset lx ux; syst=’’; ws=
```

The created variables work until the end of the current shell session, and then are automatic deleted.

5. The readonly command marks a variable as immutable:

```
$ readonly VP cdir
```

Now test setting the new value for VP:

```
$ VP=HACKER
```

The Read Only mode for the variable runs until the end of the current shell session.

6. The newly created variables are local; to inherit the shell child, you must mark the variable as an external with export command. The primary shell itself also inherits some variables from its parent, the login process, and also receives them from ini files.

```
$ export vp cdir
$ export ws=”Windows 10”
```

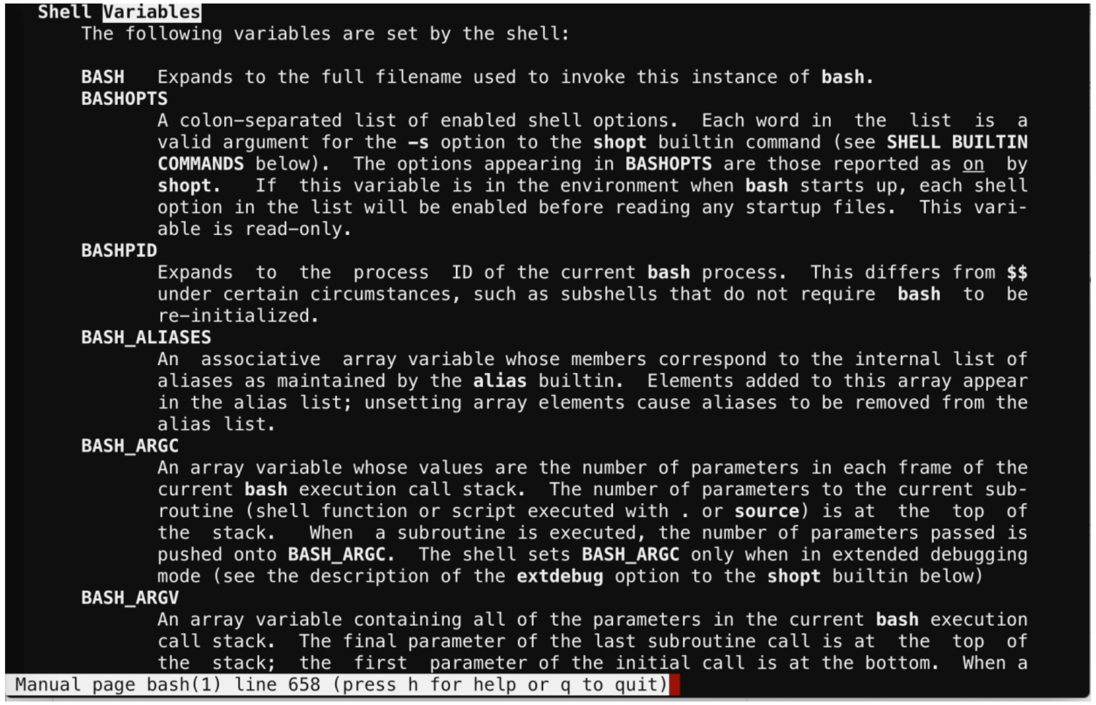
7. You can view all variables with the set command, only environment variables with the env command, a specific variables with echo.

```
$ set | more
```

4.1.5. Task 1 for Report. Fill in Table with Shell Environment Variables.

1. **Explore** the purpose of the environment variables listed in the table.
2. Give them a brief **description**.
3. Define their current **value** and **type** (Read-only, Writable).
4. Learn to **change** them, if possible on your LinuxMini.
5. Use the commands:

```
$ env                  # list env. variables
$ env | less           # <space>, b, q
$ man bash             # /variables$
$ man bash             # /prompting$
$ echo $PS1            # out current value
$ t=$PS1               # save PS1$
$ PS1='abracadabra' # define new value
$ echo $PS1            # test new value
$ PS1=`echo 'Hi, '$LOGNAME`   # change
$ echo $PS1            # test value
$ PS1=$t               # restore old PS1
```



```
   Shell Variables
       The following variables are set by the shell:

       BASH   Expands to the full filename used to invoke this instance of bash.
       BASHOPTS
              A colon-separated list of enabled shell options.  Each word in  the  list  is  a
              valid argument for the -s option to the shopt builtin command (see SHELL BUILTIN
              COMMANDS below).  The options appearing in BASHOPTS are those reported as on  by
              shopt.   If  this variable is in the environment when bash starts up, each shell
              option in the list will be enabled before reading any startup files.  This vari-
              able is read-only.
       BASHPID
              Expands  to  the  process  ID of the current bash process.  This differs from $$
              under certain circumstances, such as subshells that do not require  bash  to  be
              re-initialized.
       BASH_ALIASES
              An  associative  array variable whose members correspond to the internal list of
              aliases as maintained by the alias builtin.  Elements added to this array appear
              in the alias list; unsetting array elements cause aliases to be removed from the
              alias list.
       BASH_ARGC
              An array variable whose values are the number of parameters in each frame of the
              current bash execution call stack.  The number of parameters to the current sub-
              routine (shell function or script executed with . or source) is at  the  top  of
              the  stack.   When  a subroutine is executed, the number of parameters passed is
              pushed onto BASH_ARGC.  The shell sets BASH_ARGC only when in extended debugging
              mode (see the description of the extdebug option to the shopt builtin below)
       BASH_ARGV
              An array variable containing all of the parameters in the current bash execution
              call stack.  The final parameter of the last subroutine call is at  the  top  of
              the  stack;  the  first  parameter of the initial call is at the bottom.  When a
 Manual page bash(1) line 658 (press h for help or q to quit)
```

TABLE. SHELL ENVIRONMENT VARIABLES ANALYZE.

Variable	Short Description	Type:ro-wr	Your Linux Value
PATH			
HOME			
TERM			
PS1			
PS2			
LOGNAME	Login User Name (account)	WR	stud
MAIL			
MAILCHEK			
HOSTNAME			
HOSTTYPE			
HISTSIZE			
HISTFILE			
TMOUT			
IGNOREEOF			
IFS			
SECONDS			
OSTYPE			
PWD			
OLDPWD			
EDITOR			
RANDOM			

SHLVL			
SHELL			
PROMPT_COMMAND	Define a command that is executed every time before showing PS1. For PS1, PS2, PS3, and PROMPT_COMMAND, dynamically calculated escape sequences exist, such as: \w - current directory, full path; \W - current directory, without path; \u - username; \h - hostname; \$ - $ for user and # for root; \t is the time of day; \d - current date; \s - the current shell; and others.	WR	“Is -l”
EUID			
PPID		R**O**	
GROUPS			

4.2. Make Example of bash config file.

4.2.1. Bash config files:

• /bin/bash	The bash executable
• /etc/profile	The systemwide initialization file, executed for login shells (example bash)
• /etc/bash.bashrc	The systemwide per-interactive-shell startup file
• /etc/bash.bash.logout	The systemwide login shell cleanup file, executed when a login shell exits
• ~/.bash_profile	The personal initialization file, executed for login shells
• ~/.bashrc	The individual per-interactive-shell startup file
• ~/.bash_logout	The individual login shell cleanup file, executed when a login shell exits
• ~/.inputrc	Individual readline initialization file

You need modify only 3 files: ˜/.bash_profile, ˜/.bashrc, ˜/.bash_logout.

4.2.2. Linux Console Control Basic (ESC sequence).

Read before:

```
$ man 4 console_codes
```

The -e and -n options for the echo command are needed to enable the interpretation of ESC sequences and to cancel the transition to a new line.

ESC sequence: `\033ParamArgum`, where `\033` is the <ESC> character, followed by the Parameters and Arguments.

Combining of such control sequences is allowed. For example, 3 color setting commands `\033[1m\033[5m\033[36m` can be replaced by the equivalent sequence `\033[1;5;36m`

Color Control:

\033[Pm or \e[Pm – templates

P-param	Result
0	reset all attributes to their defaults
1	set bold
2	set half-bright
4	set full bright
30	set black foreground
31	set red foreground
32	set green foreground
33	set brown foreground
34	set blue foreground
35	set magenta foreground
36	set cyan foreground
37	set white foreground
39	set underscore off, set default foreground color

Color Control:

\033[Pm or \e[Pm – templates

P-param	Result
22	set normal bright
…	
25	blink off
40	set black background
41	set red background
42	set green background
43	set brown background
44	set blue background
45	set magenta background
46	set cyan background
47	set white background
…	
49	set default background color

Cursor Position Control:

\033[#P or \e[#P - templates

P-param	Result
A	Move cursor up the indicated # of rows.
B	Move cursor down the indicated # of rows.
C	Move cursor right the indicated # of columns.
D	Move cursor left the indicated # of columns.
E	Move cursor down the indicated # of rows, to column 1.
F	Move cursor up the indicated # of rows, to column 1.
G	Move cursor to indicated column in current row.
H	Move cursor to the indicated row, column (origin at 1,1).

Sound Control:

\033[P;#] \e[P;#] – templates

P-param	Result
\007	make a sound;
\033[10;#]	set the frequency of the sound signal in hertz;
\033[11;#]	set the signal duration in milliseconds.

Combining the last two sequences is allowed.
For example, \033[10;55]\033[11;30] can be replaced by the equivalent sequence \033[10;55;11;30] or \e[10;55;11;30]

Other Param

\e7	Save cursor position (or \e7)
\e8	Restore cursor position (or \e8)
\e[K	Clear a string from cursor position to right end
\2;999r	Scroll region

Exercises 1. Change the background color to red (easy for root). Test it::

```
$ echo -en "\033[41m"
```

Exercises 2. Display on screen the message in bold white fonts on a red background and return the terminal to its normal state.

```
$ echo -en "\033[37;1;41m ATTENTION \033[0m"
```

Exercises 3. Set color prompt PS1: green clock; then a solid red on magenta current directory, dollar and space; then return to normal colors:

```
$ PS1="\e[1;32m\t\e[25;31;45m\w\$ \e[0m"
```

Exercises 4. Set the frequency and duration of the sound signal to 5000Hz and 200ms and emit a sound signal:

```
$ echo -en "\e[10;5000;11;200]\007"
```

Exercises 5. Test escape-commands for terminal: J, K, L, M, P, X, a, c, d, e, f, g, h, l, m, n, q, r, s, u.

4.2.3. ~/.PROFILE CONFIG FILE EXAMPLE.

Test it.

```
# .profile Example:
PATH=$PATH:$HOME:~/bin         # add bin-directory to your commands finding path
TERM=xterm-256color            # set new Terminal, look infocmp vt100 vt220 linux xterm-256color
PS1="\t \h\w`echo "\007"`\$"   # set prompt with beep signal
export TERM PATH PS1           # export to child
alias lh="ls -hal"             # set new short name for command ls
echo "Hi, $LOGNAME. `who|wc -l` users are in system"  # Hello with statistic
# Цветные часы в верхней строке экрана
PROMPT_COMMAND='echo -en "\e7\e[2;999r\e[1;1H\e[00;44m\e[K"`date`"\e[00m\e8"'
```

TASK 2 FOR REPORT. (MAKE A SCREENSHOT WITH WORKING CLOCK AND YOUR SCRIPT CODE).

Make a new Clock on **middle line** (use tput cols command) on first row with format HH:MM.SS red color on green background.

```
$ cat ~/.profile
```

4.3. Modify Bash Config Files for your variant Nr of Task.

Task 3 for Report. (Make Text Copy for Your Variant Nr of Task).

4.3.1. Generate your Variant Number and Select Task Variants v00 and vN, where N –generated from Your Name.

a) Write your LastName in the letters of the English alphabet. Must be at least 7 letters, if not enough, then add the required number of letters from the FirstName (if not enough, then repeat LastName and FirstName).

For example, for Yurijs Li there will be LIYURIJS.

b) Replace the first 7 letters with their ordinal numbers in the alphabet.

For example, (12 + 09 + 25 + 21 +18 + 09 + 10).

c) Consistently add modulo 10 these 7 numbers and add 1.

For example, (12 + 09 + 25 + 21 +18 + 09 + 10)mod10 + 1 = 104mod10 + 1 = 4 +1 = 5 (it's Your Variant Nr).

4.3.2. Hints for You.

You need modify 3 files (˜/.bash_profile, ˜/.bash_logout, ˜/.bashrc), if necessary, create these files using the **touch** command.

Hint for a). You will need use: echo, cal -3, $LOGNAME to modify the ~/.bach_profile file.

Hint for b). You will need use: echo, $LOGNAME, $SECONDS, sleep to modify the ~/.bash_logout.

Hint for c). You will need use: echo, $PROMPT_COMMAD, $PS1 to modify the ~/.bashrc file.

Hint for d). You will need use: different variables, commands and constructions to make a special value of PS1 on ~/.bashrc file.

Hint for e), You will need use: ESC-codes to control the foregroung console colors – man 4 console _codes.

Hint for f). You will need use: ESC-codes to control the background console colors – man 4 console _codes.

Hint for g). You will need: export command to modify the ~/.bashrc file.

TASKS VARIANTS.

v00. For All. Change the configuration files of your shell (example, bash) as follows:

a) At the beginning of the work session (after login procedure), a greeting should be displayed in the following format:

Hello UserName!

Next is the Calendar for the previous, current and next months in one line (cal -3).

b) At the end of the work session (after the exit command), a farewell should be displayed for 5 seconds in the following format:

Goodbye, UserName! You worked s seconds ($SECONDS).

where UserName is the user login name.

v01. Change the configuration files of your shell (example, bash) as follows:

c) A user's primary prompt (PS1) should be dynamically changing (recalculated after each press of the Enter key);

d) PS1 format:

HH:MM W \

where NN and MM, respectively, hours and minutes with a leading zero, and W day of week without a leading zero (date +%);

e) The color of PS1 should be red on black;

f) The color of the rest of the command line should be white on black;

g) PS1 must be inherited in child shells.

v02. Change the configuration files of your shell (example, bash) as follows:

c) A user's primary prompt (PS1) should be dynamically changing (recalculated after each press of the Enter key);

d) PS1 format:

K |

where K is the number, reflecting the number of users who are currently interactively working in the system (who | wc);

e) The color of PS1 should be green on black;

f) The color of the rest of the command line should be white on black;

g) PS1 must be inherited in child shells.

v03. Change the configuration files of your shell (example, bash) as follows:

c) A user’s primary prompt (PS1) should be dynamically changing (recalculated after each press of the Enter key);

d) PS1 format:

X >>>

where X is the number, reflecting the total number (recursively) of the user’s files and subdirectories in his home directory (ls | wc);

e) The color of PS1 should be red on black;

f) The color of the rest of the command line should be green on black;

g) PS1 must be inherited in child shells.

v04. Change the configuration files of your shell (example, bash) as follows:

c) A user’s primary prompt (PS1) should be dynamically changing (recalculated after each press of the Enter key);

d) PS1 format:

M :\\

where M is the number, reflecting the operating time in the current session, measured in minutes ($SECONDS and $[expression]);

e) The color of PS1 should be blue on black;

f) The color of the rest of the command line should be red on black;

g) PS1 must be inherited in child shells.

v05. Change the configuration files of your shell (example, bash) as follows:

c) A user’s primary prompt (PS1) should be dynamically changing (recalculated after each press of the Enter key);

d) PS1 format:

\P\>

where P is the number, reflecting the numbers of current processes (ps aux | wc) ;

e) The color of PS1 should be white on black;

f) The color of the rest of the command line should be green on black;

g) PS1 must be inherited in child shells.

v06. Change the configuration files of your shell (example, bash) as follows:

c) A user's primary prompt (PS1) should be dynamically changing (recalculated after each press of the Enter key);

d) PS1 format:

R !

where R is a random digits number, ($RANDOM);

e) The color of PS1 should be white on black;

f) The color of the rest of the command line should be red on black;

g) PS1 must be inherited in child shells.

v07. Change the configuration files of your shell (example, bash) as follows:

c) A user's primary prompt (PS1) should be dynamically changing (recalculated after each press of the Enter key);

d) PS1 format:

|N|

where N is a number, representing the number of characters in the path of the current directory. For example, for the directory /home/student/bin N=17 (pwd | wc);

e) The color of PS1 should be red on black;

f) The color of the rest of the command line should be green on black;

g) PS1 must be inherited in child shells.

v08. Change the configuration files of your shell (example, bash) as follows:

c) A user's primary prompt (PS1) should be dynamically changing (recalculated after each press of the Enter key);

d) PS1 format:

<S>

where S is a number that reflects the system's working and idle time in seconds since the last time it was turned on (cat /proc/uptime);

e) The color of PS1 should be red on black;

f) The color of the rest of the command line should be green on black;

g) PS1 must be inherited in child shells.

v09. Change the configuration files of your shell (example, bash) as follows:

c) A user’s primary prompt (PS1) should be dynamically changing (recalculated after each press of the Enter key);

d) PS1 format:

.V.

where V is a number, reflecting the numbers files and directory of your home directory and all subdirectories recursively (ls -R | wc);

e) The color of PS1 should be green on black;

f) The color of the rest of the command line should be white on brown;

g) PS1 must be inherited in child shells.

v10. Change the configuration files of your shell (example, bash) as follows:

c) A user’s primary prompt (PS1) should be dynamically changing (recalculated after each press of the Enter key);

d) PS1 format:

"H"

where H is a number, reflecting the size of your command history (history | wc);

e) The color of PS1 should be green on brown;

f) The color the rest of the command line should be white on brown;

g) PS1 must be inherited in child shells.

ПОМОЩЬ НА РУССКОМ. Выберете вариант задания v00 и vN, где N – ваш номер варианта = вашему номеру в списке группы.

Подсказка 1. Для bash вам нужно модифицировать файлы ~/.bach_profile, ~/.bashrc и ~/.bash_logout.

Подсказка 2. Вам потребуются: echo, cal -3, $LOGNAME, $SECONDS, sleep.

Подсказка 3. Вам потребуются переопределить $PROMPT_COMMAD и $PS1.

Подсказка 4. Вам потребуются использовать ESC-коды для управления цветами на терминале – man 4 console _codes.

v00-RU. Для всех. Измените конфиг-файлы вашего командного интерпретатора (например, bash) следующим образом:

a) В начале сеанса работы (после процедуры login) должно выводиться приветствие, имеющее следующий формат:

Здравствуйте, UserName!

Далее идет Календарь на предыдущий, текущий и следующий месяцы в одну строку.

b) В конце сеанса работы (после команды exit) на 5 секунд должно выводиться прощание, имеющая следующий формат:

До свидания, UserName! Вы работали S секунд.

где UserName это логин пользователя.

v01-RU. Измените конфигурационные файлы вашего командного интерпретатора (например, bash) следующим образом:

c) Должна формироваться динамически изменяющаяся (после каждого нажатия клавиши Enter) первичная подсказка пользователя (PS1).

d) Формат PS1:

HH:MM.S \

где HH и MM соответственно часы и минуты с ведущим нулем, а S секунды без ведущего нуля.

e) Цвет PS1 должен быть красный на черном.

f) Цвет остальной командной строки должен быть белый на черном.

g) PS1 должна наследоваться в дочерние shell.