Introduction to the UNIX shell the C language

Week 01 – Lab

Outline

- Configuration
- The UNIX Shell
- The C Language

Configuration

- Login to your VM via SSH
- Ubuntu, Mac:

```
$ssh -X [username]@osc-[N].edu.innopolis.ru
```

- Windows: Putty client

What is the UNIX shell?

- **Shell** is a text user interface (TUI) for access to an operating system's services. Has many implementations: bash shell, original Unix shell, Bourne shell, ksh, csh, etc.

Introduction to the UNIX shell

`whoami` - Print userid.

`hostname` - Show the system's host name.

`man <item>` - Display **manual** for the **`<item>`**. Use arrows to navigate and **`q`** to exit. Example:

`man whoami` - Display manual on command **`whoami`**.

`man man` - Display man on man.

`man --help` - The other way to get help on command is to write an option **`--help`** or often **`-h`**.

Shell - Display

`less` - Display the contents one screen at a time with navigation.

`head` - Print the first lines of file to standard output.

`tail` - Print the last lines of file to standard output.

```
man -h | head
man --help | tail
```

`grep PATTERN <file>` - Search for PATTERN in file or stdin.

Shell - Streams

Standard streams are preconnected communication channels of programs. They are:

- **`stdin`** - standard input that going into program,
- **`stdout`** - standard out where program writes output,
- **`stderr`** - to display error messages.

It is possible to redirect streams to or from files with **>** and **<**.

Shell - Pipelines

`ls > list.txt` - Save list of files in current directory to file.txt.

`head -n 3 < file.txt` - Display the first 3 entries.

It is possible to redirect output of one program to input of another by **|** (pipe symbol).

```
ls | sort -r | tail -n 3
```

Get list of files, reverse sort and display the 3 last.

Shell - File system commands

`pwd` - Print name of current/working directory.

`mkdir <dirname>` - Make directory.

`cd <path>` - Change directory.

`rm <filenames>` - Remove a file.

`rm -r <dirname>` - Remove (recursive) a directory.

`ls` - List content of a directory.

`mv <old_path> <new_path>` - Move file.

`cat <filenames>` - Concatenate files to stdout.

`gedit <filename>` - Run text editor for GNOME.

Shell - File System - Special Characters

~ - home directory

. - represent current directory

.. - represent parent directory of current directory

Examples:

```
cd ..
 ls .
 cd ~
```

Shell - File System FAQ

Q: How to create a new file?

```
touch <filename>
cat > <filename>
echo > <filename>
gedit <filename>
```

Q: How to rename file?

```
mv <oldname> <newname>
```

Foreground and Background

Foreground processes block shell during execution and **background** do not. Appending **&** will run process in background.

```
gedit &
```

Foreground process can be suspend by **ctrl+z** and run in background with **bg** or foreground with **fg**.

jobs - display list of jobs.

A job can be chosen by it's number in list with %, %+ for current job and %- fr previous:

`fg %1` - run job 1 in foreground

Exercise 1

Create directory “**week1**” in home directory.

```
mkdir ~/week1
cd ~/week1
```

List entries in /usr/bin that contain “**`gcc`**” in reverse alphabetical order. Save results in “**`~/week1/ex1.txt`**”.

Exercise 2

Try some commands and save history to “**~/week1/ex2.txt**”.

```
history > ex2.txt
```

Exercise 3

Write a shell script “**`ex3.sh`**” that prints time (use **date** command), then sleep for 3 seconds (use **sleep 3**) and prints time again. Run script with:

```
sh ex3.sh
```

Exercise 4 - Hello World

Write “Hello world” in the C language. Create source file: **`gedit ~/week1/main.c`**

Write program:

```
#include <stdio.h>
 int main()
 {
   printf(“Hello World”);
   return 0;
 }
```

Exercise 4 - Compilation

Compile the program, where **ex4** is name of executable file:

```
gcc main.c -o ex4
```

Run the program with:

```
./ex4
```

Useful Links

About foreground and background processes

The End.

Be strong!

Week 01 – Lab 01