Київський національний університет імені Тараса Шевченка

Факультет комп’ютерних наук та кібернетики

АНАЛІТИЧНИЙ ЗВІТ

з курсу

“Актуальні проблеми обробки інформації в комп'ютерних мережах”

на тему

“Приклади сценаріїв мовою Perl для операційної системи Unix”

Виконав

студент 1-го курсу магістратури

Факультету Комп’ютерних наук та кібернетики

Сакевич Руслан Дмитрович

Київ - 2019

Вступ

Perl (неофіційна розшифровка англ. Practical Extraction and Reporting Language — практична мова витягів та звітів) — високорівнева, інтерпретована, динамічна мова програмування загального призначення. Perl було розроблено у 1987 році Ларі Воллом, лінгвістом і програмістом за освітою, який у той час працював системним адміністратором у NASA, як скриптова мова для Unix, метою якої було полегшити процес обробки текстів файлів логів. З того часу до мови було внесено багато змін, і здійснено перегляд її концепції та архітектури, в результаті чого вона стала дуже популярною серед програмістів. Ларі Волл продовжує працювати над ядром мови, і наразі очікується вихід нової версії, Perl 6.

Perl запозичує можливості багатьох інших мов програмування, як то C, shell scripting, AWK та sed. Мова надає потужні можливості для обробки тексту без довільних обмежень на довжину даних багатьох сучасних інструментів Unix, полегшує маніпуляції з текстовими файлами. Застосовується для програмування графіки, системного адміністрування, у мережному програмуванні, для програмного забезпечення, яке взаємодіє з базами даних, у програмуванні CGI для веб. Perl за свою гнучкість і потужність отримав прізвисько «швейцарського армійського ножа мов програмування».

В даній роботі розглядаються сценарії написані мовою Perl, що тим чи іншим чином використовують можливості операційних систем сімейства Unix. Багато з наведених сценаріїв є спрощеними реалізаціями стандартних утиліт Unix. Вони наведені для демонстрації того наскільки легко і просто можна звертатися до операційної системи з самої мови програмування.

Список сценаріїв

1. Спрощена реалізація утиліти **date**
2. Спрощена реалізація утиліти **ls**
3. Спрощена реалізація утиліти **pwent**
4. Спрощена реалізація утиліти **time**
5. Спрощена реалізація утиліти **env**
6. Спрощена реалізація утиліти **grep**
7. Спрощена реалізація утиліти **ping**
8. Спрощена реалізація утиліти **kill**
9. Спрощена реалізація утиліти **xargs**
10. Спрощена реалізація утиліти **nslookup**
11. Спрощена реалізація утиліти **hexdump**
12. Спрощена реалізація утиліти **sort**
13. Спрощена реалізація утиліти **md5sum**
14. Спрощена реалізація утиліти **curl**
15. Спрощена реалізація утиліти **factor (multithreading)**
16. Спрощена реалізація утиліти **mail**
17. Спрощена реалізація утиліти **ps**
18. Форматування повідомлення в форматі **JSON**
19. Знаходження **PID**, **UID** та **GID** процесу
20. Обробка сигналів операційної системи **Unix**
21. Рекурсивний обхід директорії
22. Підрахунок кількості файлів (**multiprocess**)
23. Комунікація між процесами за допомогою **Named Pipes**
24. **TCP** сервер з підтримкою декількох одночасних з'єднань
25. **TCP** клієнт (спрощена версія утиліти **netcat**)
26. Широкомовлення **UDP** повідомлення
27. Комунікація між процесами за допомогою **Unix Socket**
28. Перелік файлів на віддаленому **FTP** сервері
29. Архівація файлів в **.tar.gz**
30. **HTTP POST** запит на сервер

|  |
| --- |
| Спрощена реалізація утиліти **date** операційної системи Unix |
| *#!/usr/bin/env perl*    *# date util*  *#*  *# Usage:*  *# $ ./date.pl*  *# Wed Apr 10 00:51:58 EEST 2019*  *#*  *# $ date*  *# Wed Apr 10 00:51:58 EEST 2019*    **use** strict;  **use** warnings;  **use** POSIX qw(strftime);  **use** feature "say";    say strftime "%a %b %e %H:%M:%S %Z %Y", localtime; |

|  |
| --- |
| Спрощена реалізація утиліти **ls** операційної системи Unix |
| *#!/usr/bin/env perl*    *# ls util*  *#*  *# Usage:*  *# $ ./ls.pl*  *# rwxr-xr-x lionell users 768 Apr 10 00:08 pretty\_print.pl*  *# rwxr-xr-x lionell users 255 Apr 10 01:30 date.pl*  *# rwxr-xr-x lionell users 1400 Apr 10 02:38 hello.pl*  *# rw-r--r-- lionell users 290 Apr 10 00:10 out.txt*  *# rwxr-xr-x lionell users 655 Apr 08 21:16 currency.pl*  *# rw-r--r-- lionell users 145 Apr 09 23:46 test.json*  *# rwxr-xr-x lionell users 1680 Apr 10 02:39 ls.pl*  *# rwxr-xr-x lionell users 976 Apr 10 00:52 time.pl*  *#*  *# $ ./ls.pl /root*  *# Permission denied at ./ls.pl line 22.*  *#*  *# $ sudo ./ls.pl /root*  *# rwxr-xr-x root root 4096 Feb 16 00:17 .local*  *# rw-r--r-- root root 45 Oct 18 09:40 .nix-channels*  *# rwx------ root root 4096 Apr 09 20:00 .nix-defexpr*  *# rw------- root root 20661 Mar 23 21:59 .viminfo*  *# rwx------ root root 4096 Feb 16 00:17 .emacs.d*  *# rw------- root root 4825 Apr 01 22:58 .bash\_history*  *# rw------- root root 50 Feb 16 00:17 .Xauthority*  *# rwxr-xr-x root root 4096 Feb 16 00:17 .compose-cache*  *# rwx------ root root 4096 Feb 16 00:17 .dbus*  *# rwxr-xr-x root root 4096 Oct 30 06:45 .cache*    **use** strict;  **use** warnings;  **use** feature "say";  **use** experimental qw(switch);  **use** Fcntl qw(:mode);  **use** POSIX qw(strftime);  **use** File::stat;  **use** File::Spec;    **sub** perms {  given (shift @\_) {  when (7) { 'rwx' }  when (6) { 'rw-' }  when (5) { 'r-x' }  when (4) { 'r--' }  when (3) { '-wx' }  when (2) { '-w-' }  when (1) { '--x' }  when (0) { '---' }  default { die "Invalid file permissions"; }  }  }    **sub** fmt {  my $path = shift @\_;  my $stats = stat $path;  my $str = '';  $str .= perms(($stats->mode & S\_IRWXU) >> 6); *# user perms*  $str .= perms(($stats->mode & S\_IRWXG) >> 3); *# group perms*  $str .= perms($stats->mode & S\_IRWXO); *# other perms*  $str .= " " . getpwuid $stats->uid; *# user name*  $str .= " " . getgrgid $stats->gid; *# group name*  $str .= " " . -s $path; *# file size*  $str .= "**\t**" . strftime "%b %d %H:%M", localtime $stats->mtime;  $str .= " " . (split '/', $path)[-1]; *# file name*  }    my $dir = shift || '.';  opendir DIR, $dir or die $!;  for (sort readdir DIR) {  my $path = File::Spec->join($dir, $\_);  next if $\_ eq '.' or $\_ eq '..' or -l $path; *# skip '.', '..' and symbolic links*  say fmt $path;  }  closedir(DIR); |

|  |
| --- |
| Спрощена реалізація утиліти **pwent** операційної системи Unix |
| *#!/usr/bin/env perl*    *# List all users and their shells*  *#*  *# $ ./pwent.pl*  *# /run/current-system/sw/bin/bash root*  *# /run/current-system/sw/bin/nologin messagebus*  *# /run/current-system/sw/bin/nologin polkituser*  *# /run/current-system/sw/bin/nologin cups*  *# /run/current-system/sw/bin/nologin rtkit*  *# /run/current-system/sw/bin/nologin transmission*  *# /run/current-system/sw/bin/nologin systemd-journal-gateway*  *# /run/current-system/sw/bin/nologin systemd-network*  *# /run/current-system/sw/bin/nologin systemd-resolve*  *# /run/current-system/sw/bin/nologin systemd-timesync*  *# /run/current-system/sw/bin/nologin nscd*  *# /run/current-system/sw/bin/fish lionell*  *# /run/current-system/sw/bin/nologin nobody*    **use** strict;  **use** warnings;  **use** User::pwent;  **use** feature "say";    while (my $e = getpwent()) {  say $e->shell . " **\t**" . $e->name;  } |

|  |
| --- |
| Спрощена реалізація утиліти **time** операційної системи Unix |
| *#!/usr/bin/env perl*    *# time util*  *#*  *# Usage:*  *# $ ./time.pl sleep 5*  *# Execution took 5 seconds.*  *#*  *# $ ./time.pl ping www.google.com*  *# PING www.google.com (172.217.20.196) 56(84) bytes of data.*  *# 64 bytes from 172.217.20.196: icmp\_seq=1 ttl=54 time=29.5 ms*  *# 64 bytes from 172.217.20.196: icmp\_seq=2 ttl=54 time=27.5 ms*  *# 64 bytes from 172.217.20.196: icmp\_seq=3 ttl=54 time=29.7 ms*  *# 64 bytes from 172.217.20.196: icmp\_seq=4 ttl=54 time=26.1 ms*  *# 64 bytes from 172.217.20.196: icmp\_seq=5 ttl=54 time=38.8 ms*  *# ^C*  *# --- www.google.com ping statistics ---*  *# 5 packets transmitted, 5 received, 0% packet loss, time 10ms*  *# rtt min/avg/max/mdev = 26.140/30.340/38.824/4.443 ms*  *# Execution took 5 seconds.*    **use** strict;  **use** warnings;  **use** feature "say";    my $start = time;  system join " ", @**ARGV**;  say "Execution took ", (time - $start), " seconds."; |

|  |
| --- |
| Спрощена реалізація утиліти **env** операційної системи Unix |
| *#!/usr/bin/env perl*    *# Print environment variables*  *#*  *# Usage:*  *# $ ./env.pl*  *# COLORTERM=truecolor*  *# DISPLAY=:0.0*  *# EDITOR=vim*  *# GOPATH=/home/lionell/dev/go*  *# HOME=/home/lionell*  *# LANG=en\_US.UTF-8*  *# MAIL=/var/mail/lionell*  *# PAGER=less -R*  *# SHELL=/run/current-system/sw/bin/fish*  *# SSH\_AUTH\_SOCK=/run/user/1000/ssh-agent*  *# TERM=screen-256color*  *# USER=lionell*  *# ...*    **use** strict;  **use** warnings;  **use** feature "say";    say "$\_=$ENV{$\_}" for (sort keys %ENV) |

|  |
| --- |
| Спрощена реалізація утиліти **grep** операційної системи Unix |
| *#!/usr/bin/env perl*    *# grep util*  *#*  *# Usage:*  *# $ ifconfig | ./grep.pl broadcast*  *# inet 172.18.0.1 netmask 255.255.0.0 broadcast 172.18.255.255*  *# inet 172.19.0.1 netmask 255.255.0.0 broadcast 172.19.255.255*  *# inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255*  *# inet 192.168.0.107 netmask 255.255.255.0 broadcast 192.168.0.255*    **use** strict;  **use** warnings;    my $word = shift;  for (<STDIN>) {  print if $\_ =~ */$word/*;  } |

|  |
| --- |
| Спрощена реалізація утиліти **ping** операційної системи Unix |
| *#!/usr/bin/env perl*    *# ping util*  *#*  *# Usage:*  *# sudo ./ping.pl lun.ua*  *# ICMP ip=46.101.136.215 time=64.3ms*  *# ICMP ip=46.101.136.215 time=46.9ms*  *# ICMP ip=46.101.136.215 time=43.8ms*  *# ICMP ip=46.101.136.215 time=39.8ms*  *# ^C*    **use** strict;  **use** warnings;  **use** feature "say";  **use** Net::Ping;    my $p = Net::Ping->**new**("icmp");  while (1) {  my ($status, $time, $ip) = $p->ping(shift);  if ($status) {  printf "ICMP $ip time=%0.1fms**\n**", 1000 \* $time;  } else {  say "Server is not reachable";  }  sleep(1);  }  $p->close(); |

|  |
| --- |
| Спрощена реалізація утиліти **kill** операційної системи Unix |
| *#!/usr/bin/env perl*    *# kill util*  *#*  *# $ ./signal.pl &*  *# Going into sleep for 20 seconds.*  *# Press Ctrl-C to interrupt.*  *# Try killing the process(pid: 18277).*  *#*  *# $ ./kill.pl 18277*  *# Caught a SIGTERM at ./signal.pl line 23.*    **use** strict;  **use** warnings;    kill SIGTERM => shift; |

|  |
| --- |
| Спрощена реалізація утиліти **xargs** операційної системи Unix |
| *#!/usr/bin/env perl*    *# xargs util*  *#*  *# Usage:*  *# $ cat file*  *# Lorem ipsum.*  *# Sit dolor.*  *# Amet*  *#*  *# $ cat file | ./xargs util -n*  *# Lorem ipsum.Sit dolor.Amet*    **use** strict;  **use** warnings;    for (<STDIN>) {  chomp;  my @cmd = @**ARGV**;  push @cmd, $\_;  system @cmd;  } |

|  |
| --- |
| Спрощена реалізація утиліти **nslookup** операційної системи Unix |
| *#!/usr/bin/env perl*    *# nslookup util*  *#*  *# Usage:*  *# $ ./nslookup.pl www.google.com*  *# www.google.com => 216.58.215.100*    **use** strict;  **use** warnings;  **use** feature "say";  **use** Socket;  **use** Net::hostent;    my $host = gethostbyname shift;  my $name = $host->name;  my $addresses = $host->addr\_list;  my @lst = map { inet\_ntoa($\_) } @$addresses;  say "$name => @lst"; |

|  |
| --- |
| Спрощена реалізація утиліти **hexdump** операційної системи Unix |
| *#!/usr/bin/env perl*    *# hexdump util*  *#*  *# Usage:*  *# $ ./hexdump.pl 64 $(which bash)*  *# 00000000 7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00*  *# 00000010 02 00 3e 00 01 00 00 00 b0 0a 42 00 00 00 00 00*  *# 00000020 40 00 00 00 00 00 00 00 d0 0b 0f 00 00 00 00 00*  *# 00000030 00 00 00 00 40 00 38 00 0a 00 40 00 1e 00 1d 00*  *#*  *# $ hexdump -C -n 64 $(which bash)*  *# 00000000 7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00 |.ELF............|*  *# 00000010 02 00 3e 00 01 00 00 00 b0 0a 42 00 00 00 00 00 |..>.......B.....|*  *# 00000020 40 00 00 00 00 00 00 00 d0 0b 0f 00 00 00 00 00 |@...............|*  *# 00000030 00 00 00 00 40 00 38 00 0a 00 40 00 1e 00 1d 00 |....@.8...@.....|*    **use** strict;  **use** warnings;    my ($count, $binary) = @**ARGV**;  open my $file, "<:raw", $binary;  for (0 .. $count - 1) {  printf "%08x ", $\_ unless $\_ % 16; # show offset every 16 bytes  read($file, my $byte, 1); # read one byte  printf " " unless $\_ % 8; # additional whitespace every 8 bytes  printf "%02x ", ord $byte; # print byte  printf "**\n**" unless ($\_ + 1) % 16; # move to new line after 16 bytes  }  close $file; |

|  |
| --- |
| Спрощена реалізація утиліти **sort** операційної системи Unix |
| *#!/usr/bin/env perl*    *# sort util*  *#*  *# Usage:*  *# $ cat in.txt*  *# Lorem ipsum.*  *# Sit dolor.*  *# Amet*  *#*  *# $ ./sort.pl in.txt*  *# Amet*  *# Lorem ipsum.*  *# Sit dolor.*  *#*  *# $ cat in.txt | ./sort.pl*  *# Amet*  *# Lorem ipsum.*  *# Sit dolor.*    **use** strict;  **use** warnings;    print sort <>; |

|  |
| --- |
| Спрощена реалізація утиліти **md5sum** операційної системи Unix |
| *#!/usr/bin/env perl*    *# md5sum util*  *#*  *# Usage:*  *# $ ./md5sum.pl test.json*  *# 2bd2e3973c48e87576172ef59584f627*  *#*  *# $ cat test.json | ./md5sum.pl*  *# 2bd2e3973c48e87576172ef59584f627*  *#*  *# $ md5sum test.json*  *# 2bd2e3973c48e87576172ef59584f627 test.json*    **use** strict;  **use** warnings;  **use** Digest::MD5 qw(md5\_hex);    print md5\_hex join '', <>; |

|  |
| --- |
| Спрощена реалізація утиліти **curl** операційної системи Unix |
| *#!/usr/bin/env perl*    *# curl util*  *#*  *# Usage:*  *# $ ./curl.pl 'http://wttr.in?format=3'*  *# Kyiv, Ukraine: +9°C*    **use** strict;  **use** warnings;  **use** HTTP::Tiny;    my $response = HTTP::Tiny->new->get(shift);  if ($response->{success}) {  print $response->{content};  } else {  die "Failed $response->{status} $response->{reason}";  } |

|  |
| --- |
| Спрощена реалізація утиліти **factor** операційної системи Unix |
| *#!/usr/bin/env perl*    *# factor util (multithreaded)*  *#*  *# Usage:*  *# $ time -p ./factor.pl 1125899839733759 1125899839733759 1125899839733759*  *# real 4.47*  *# user 12.04*  *# sys 0.00*    **use** strict;  **use** warnings;  **use** feature "say";  **use** threads;    **sub** factor {  my $num = shift;  my $id = threads->tid();  print "[$id] $num = ";    my @factors = ();  for (my $i = 2; $i \* $i <= $num; $i++) {  until ($num % $i) {  $num /= $i;  push @factors, $i;  }  }  push @factors, $num if $num > 1;  say join " \* ", @factors;  }    my @threads = ();  for (@**ARGV**) {  push @threads, threads->create(**sub** { factor $\_; threads->exit(); });  }  for (@threads) {  $\_->join();  } |

|  |
| --- |
| Спрощена реалізація утиліти **mail** операційної системи Unix |
| *#!/usr/bin/env perl*    *# mail util (using sendmail)*  *#*  *# Usage:*  *# ./email.pl -t my\_friend@gmail.com -f me@gmail.com \*  *# -s 'Top secret' -m 'Check this link out'*  *# Sent successfully!*    **use** strict;  **use** warnings;  **use** feature "say";  **use** Getopt::Long qw(GetOptions);    my $to = "";  my $from = "";  my $subject = "";  my $message = "";  GetOptions(  "to=s" => \$to,  "from=s" => \$from,  "subject=s" => \$subject,  "message=s" => \$message,  ) or die $!;    open my $mail, "|sendmail -t";  print $mail "To: $to**\n**";  print $mail "From: $from**\n**";  print $mail "Subject: $subject**\n\n**";  print $mail $message;  close $mail;    say "Sent successfully!"; |

|  |
| --- |
| Спрощена реалізація утиліти **ps** операційної системи Unix |
| *#!/usr/bin/env perl*    *# Show process list via external call*  *#*  *# Usage:*  *# $ ./ps.pl*  *# USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND*  *# lionell 2483 0.0 0.0 146924 3616 pts/1 S+ Apr08 0:00 tmux*  *# lionell 7231 0.0 0.0 300816 4872 pts/4 Ss+ Apr07 0:00 -fish*  *# lionell 7361 0.0 0.0 300816 4976 pts/5 Ss+ Apr07 0:00 -fish*  *# lionell 8568 0.0 0.0 115336 11624 ? S 00:31 0:16 i3bar*  *# lionell 8569 0.0 0.0 605260 6084 ? Sl 00:31 0:32 i3status*  *# lionell 9773 0.0 0.0 133504 2304 pts/6 R+ 23:17 0:00 ps ux*  *# lionell 13356 0.0 0.0 311972 9944 pts/6 Ss 01:44 0:08 -fish*  *# lionell 30060 0.0 0.0 299200 7120 pts/3 Ss Apr09 0:00 -fish*  *# lionell 30158 0.0 1.9 509316 324544 pts/3 S+ Apr09 0:13 perl*  *# lionell 31479 0.0 0.2 180748 36160 ? Ss Apr02 2:09 tmux*  *# lionell 31704 0.0 0.0 377932 8332 pts/2 Ss+ Apr02 0:27 -fish*  *# ...*    **use** strict;  **use** warnings;    system ("ps", "ux"); |

|  |
| --- |
| Форматування повідомлення в форматі **JSON** |
| *#!/usr/bin/env perl*    *# JSON pretty printer*  *#*  *# Usage:*  *# $ cat test.json*  *# {"classes":["Chemistry","Math","Litreture"],"gender":null,"name":"Foo Bar","email":"foo@bar.com","address":{"city":"Fooville","planet":"Earth"}}*  *# $ ./pretty\_print.pl test.json*  *# {*  *# "address" : {*  *# "city" : "Fooville",*  *# "planet" : "Earth"*  *# },*  *# "classes" : [*  *# "Chemistry",*  *# "Math",*  *# "Litreture"*  *# ],*  *# "email" : "foo@bar.com",*  *# "gender" : null,*  *# "name" : "Foo Bar"*  *# }*  *#*  *# Alternative usage:*  *# $ cat test.json | ./pretty\_print.pl*    **use** strict;  **use** warnings;  **use** JSON::MaybeXS;    print JSON::MaybeXS  ->**new**(pretty => 1, sort\_by => 1)  ->encode(JSON::MaybeXS::decode\_json join "", <>); |

|  |
| --- |
| Знаходження **PID**, **UID** та **GID** процесу |
| *#!/usr/bin/env perl*    *# Show pid, uid, gid of the current process*  *#*  *# Usage:*  *# $ ./id.pl*  *# PID: 17649*  *# Real UID: 1000*  *# Effective UID: 1000*  *# Real GIDs: 100 1 17 70 100 131 499*  *# Effective GIDs: 100 1 17 70 100 131 499*  *#*  *# $ sudo ./id.pl*  *# [sudo] password for lionell:*  *# PID: 17600*  *# Real UID: 0*  *# Effective UID: 0*  *# Real GIDs: 0 0*  *# Effective GIDs: 0 0*    **use** strict;  **use** warnings;  **use** feature "say";    say "PID: ", $$;  say "Real UID: ", $<;  say "Effective UID: ", $>;  say "Real GIDs: ", $(;  say "Effective GIDs: ", $); |

|  |
| --- |
| Обробка сигналів операційної системи **Unix** |
| *#!/usr/bin/env perl*    *# Handle system signals*  *#*  *# Usage:*  *# $ ./signal.pl*  *# Going into sleep for 20 seconds.*  *# Press Ctrl-C to interrupt.*  *# Try killing the process(pid: 5212).*  *# ^CCaught a SIGINT at ./signal.pl line 25.*  *#*  *# $ ./signal.pl & kill 5213*  *# Going into sleep for 20 seconds.*  *# Press Ctrl-C to interrupt.*  *# Try killing the process(pid: 5213).*  *# Caught a SIGTERM at ./signal.pl line 26.*    **use** strict;  **use** warnings;  **use** feature "say";    $SIG{INT} = **sub** { die "Caught a SIGINT"; };  $SIG{TERM} = **sub** { die "Caught a SIGTERM"; };    say "Going into sleep for 20 seconds.";  say "Press Ctrl-C to interrupt.";  say "Try killing the process(pid: $$).";  sleep(20);  say "Finished successfully!"  say "Real UID: ", $<;  say "Effective UID: ", $>;  say "Real GIDs: ", $(;  say "Effective GIDs: ", $); |

|  |
| --- |
| Рекурсивний обхід директорії |
| *#!/usr/bin/env perl*    *# Recursively traverse directory*  *#*  *# Usage:*  *# $ ./traverse.pl*  *# ./README.md*  *# ./currency.pl*  *# ./date.pl*  *# ./env.pl*  *# ./hello.pl*  *# ./id.pl*  *# ./ls.pl*  *# ./out.txt*  *# ./pretty\_print.pl*  *# ./pwent.pl*  *# ./signal.pl*  *# ./test.json*  *# ./time.pl*  *# ./traverse.pl*  *#*  *# $ ./traverse.pl ~/dev/labs/os*  *# /home/lionell/dev/labs/os/lab1*  *# /home/lionell/dev/labs/os/lab1/Makefile*  *# /home/lionell/dev/labs/os/lab1/in.txt*  *# /home/lionell/dev/labs/os/lab1/main.cc*  *# /home/lionell/dev/labs/os/lab2*  *# /home/lionell/dev/labs/os/lab2/Makefile*  *# /home/lionell/dev/labs/os/lab2/child.cc*  *# /home/lionell/dev/labs/os/lab2/main.cc*  *# /home/lionell/dev/labs/os/lab2/main.o*  *# /home/lionell/dev/labs/os/lab2/pipes.cc*    **use** strict;  **use** warnings;  **use** feature "say";  **use** File::Spec;    **sub** traverse {  my $path = shift;  return unless -d $path;  opendir my $dir, $path or die $!;  for (sort readdir $dir) {  next if $\_ eq '.' or $\_ eq '..';  my $new = File::Spec->join($path, $\_);  say $new;  traverse($new);  }  closedir $dir;  }    traverse (shift || '.'); |

|  |
| --- |
| Підрахунок кількості файлів (**multiprocess**) |
| *#!/usr/bin/env perl*    *# Count files in parallel (multiprocess)*  *#*  *# Usage:*  *# $ ./count.pl /home/lionell/dev*  *# PID Count Path*  *# 25064 2 /home/lionell/dev/bluck*  *# 25056 160 /home/lionell/dev/site*  *# 25063 64 /home/lionell/dev/pagerank*  *# 25060 67 /home/lionell/dev/smart-pacmans*  *# 25061 46 /home/lionell/dev/resume*  *# 25055 596 /home/lionell/dev/algorithmix*  *# 25058 510 /home/lionell/dev/slides*  *# 25057 2142 /home/lionell/dev/labs*    **use** strict;  **use** warnings;  **use** feature "say";  **use** File::Spec;    **sub** cnt {  my $path = shift;  return 1 unless -d $path;  my $res = 0;  for (list\_dir($path)) {  $res += cnt(File::Spec->join($path, $\_));  }  return $res;  }    **sub** list\_dir {  opendir my $dir, shift or die $!;  my @lst = grep { !/^\.{1,2}$/ } readdir $dir;  closedir $dir;  return @lst;  }    say "PID**\t**Count**\t**Path";  my $path = shift || '.';  my @lst = list\_dir($path);  for (@lst) {  if (not fork) {  my $new = File::Spec->join($path, $\_);  my $count = cnt($new);  say "$$**\t**$count**\t**$new";  exit;  }  }  wait for (0 .. $#lst); |

|  |
| --- |
| Комунікація між процесами за допомогою **Named Pipes** |
| *#!/usr/bin/env perl*    *# Process communication via named pipe*  *#*  *# Usage:*  *# $ ./pipe.pl*  *# [27555] Named pipe 'pipe.p' created.*  *# [27555] 'Ping' sent.*  *# [27555] 'Ping' sent.*  *# [27555] 'Ping' sent.*  *# [27555] 'Ping' sent.*  *# [27555] 'Ping' sent.*  *# [27556] 'Ping' received.*  *# [27556] 'Ping' received.*  *# [27556] 'Ping' received.*  *# [27556] 'Ping' received.*  *# [27556] 'Ping' received.*  *# [27555] Named pipe 'pipe.p' removed.*    **use** strict;  **use** warnings;  **use** feature "say";  **use** POSIX qw(mkfifo);    my $filename = "named.pipe";  mkfifo($filename, 0700) or die $!;  say "[$$] Named pipe '$filename' created.";  if (fork) {  # Parent  open my $out, ">", $filename or die $!;  for (1 .. 5) {  say "[$$] 'Ping' sent.";  say $out "Ping";  }  close $out;  wait;  unlink($filename) or die $!;  say "[$$] Named pipe '$filename' removed.";  } else {  # Child  open my $in, "<", $filename or die $!;  while (<$in>) {  chomp;  say "[$$] '$\_' received.";  }  close $in;  } |

|  |
| --- |
| **TCP** сервер з підтримкою декількох одночасних з'єднань |
| *#!/usr/bin/env perl*    *# TCP server that can handle multiple connections*  *#*  *# Usage:*  *# (SERVER)*  *# $ ./tcp\_server.pl 1234*  *# [1836] 127.0.0.1:38156 connected.*  *# [1858] 127.0.0.1:38158 connected.*  *# [1836] 127.0.0.1:38156 sent 'hello from client 1'*  *# [1858] 127.0.0.1:38158 sent 'hello from client 2'*  *# [1858] 127.0.0.1:38158 sent 'bye'*  *# [1858] 127.0.0.1:38158 disconnected.*  *# [1836] 127.0.0.1:38156 sent 'bye'*  *# [1836] 127.0.0.1:38156 disconnected.*  *# ^C[1832] Shutting down gracefully...*  *#*  *# (CLIENT 1)*  *# $ nc 127.0.0.1 1234*  *# hello from client 1*  *# Echo: hello from client 1*  *# bye*  *#*  *# (CLIENT 2)*  *# $ nc 127.0.0.1 1234*  *# hello from client 2*  *# Echo: hello from client 2*  *# bye*    **use** strict;  **use** warnings;  **use** feature "say";  **use** IO::Socket::INET;    my $sock = IO::Socket::INET->**new**(LocalPort => shift, Listen => 5, Reuse => 1) or die $!;    **sub** handle {  my $client = shift;  $SIG{INT} = **sub** {  $client->close;  $sock->close;  exit;  };  my $ip = $client->peerhost;  my $port = $client->peerport;  my $peer = "$ip:$port";  say "[$$] $peer connected.";    my $msg;  while (1) {  $client->recv($msg, 1024);  last unless length $msg;  chomp $msg;  say "[$$] $peer sent '$msg'";  last if $msg eq "bye";  $client->send("Echo: $msg**\n**");  }  say "[$$] $peer disconnected.";  $client->close;  exit;  }    my @children = ();    $SIG{INT} = **sub** {  say "[$$] Shutting down gracefully...";  $sock->close;  kill "INT", @children;  exit;  };    while (my $client = $sock->accept) {  if (my $pid = fork) {  push @children, $pid;  } else {  handle $client;  }  } |

|  |
| --- |
| **TCP** клієнт (спрощена версія утиліти **netcat**) |
| *#!/usr/bin/env perl*    *# TCP client*  *#*  *# Usage:*  *# (SERVER)*  *# $ ./tcp\_server.pl 1234*  *# [2463] 127.0.0.1:38288 connected.*  *# [2463] 127.0.0.1:38288 sent 'here'*  *# [2463] 127.0.0.1:38288 sent 'we'*  *# [2463] 127.0.0.1:38288 sent 'go'*  *# [2463] 127.0.0.1:38288 sent 'bye'*  *# [2463] 127.0.0.1:38288 disconnected.*  *# ^C[2459] Shutting down gracefully...*  *#*  *# (CLIENT)*  *# $ ./tcp\_client.pl 127.0.0.1 1234*  *# > here*  *# < Echo: here*  *# > we*  *# < Echo: we*  *# > go*  *# < Echo: go*  *# > bye*    **use** strict;  **use** warnings;  **use** IO::Socket::INET;    my $sock = IO::Socket::INET->**new**(PeerHost => shift, PeerPort => shift) or die $!;  my $msg = "";  do {  print "> ";  chomp($msg = <STDIN>);  $sock->send($msg);  $sock->recv($msg, 1024);  print "< $msg" if $msg;  } while (length $msg);  $sock->close; |

|  |
| --- |
| Широкомовлення **UDP** повідомлення |
| *#!/usr/bin/env perl*    *# UDP broadcasting*  *#*  *# Usage:*  *# (SERVER)*  *# $ ./broadcast.pl 4321*  *# Broadcasting 'Hello, world!' message...*  *#*  *# (CLIENT)*  *# $ nc -l -u 4321*  *# Hello, world!*    **use** strict;  **use** warnings;  **use** feature "say";  **use** IO::Socket::INET;    my $msg = "Hello, world!";  my $sock = IO::Socket::INET->**new**(  PeerAddr => "255.255.255.255",  PeerPort => shift,  Proto => "udp",  Broadcast => 1  ) or die $!;    $SIG{INT} = **sub** {  $sock->close;  exit;  };    say "Broadcasting '$msg' message...";  while (1) {  $sock->send($msg);  sleep(1);  } |

|  |
| --- |
| Комунікація між процесами за допомогою **Unix Socket** |
| *#!/usr/bin/env perl*    *# UNIX domain socket listener*  *#*  *# Usage:*  *# (SERVER)*  *# $ ./unix\_socket.pl unix.socket*  *# Socket file 'unix.socket' created.*  *# Client connected.*  *# Received: 'Hello from there'*  *# Client disconnected.*  *# Client connected.*  *# Received: 'I'm here again'*  *# Client disconnected.*  *# ^CSocket file removed.*  *#*  *# (CLIENT)*  *# $ nc -U unix.socket*  *# Hello from there*  *# Echo: Hello from there*  *# ^C*  *#*  *# $ nc -U unix.socket*  *# I'm here again*  *# Echo: I'm here again*  *# ^C*    **use** strict;  **use** warnings;  **use** feature "say";  **use** IO::Socket::UNIX;    my $file = shift;  my $server = IO::Socket::UNIX->**new**(  Type => SOCK\_STREAM(),  Local => $file,  Listen => 1,  ) or die $!;    $SIG{INT} = **sub** {  $server->close;  unlink $file;  say "Socket file removed.";  exit;  };    say "Socket file '$file' created.";  while (my $conn = $server->accept) {  say "Client connected.";  while (<$conn>) {  chomp;  say "Received: '$\_'";  $conn->print("Echo: $\_**\n**");  }  $conn->close;  say "Client disconnected.";  } |

|  |
| --- |
| Перелік файлів на віддаленому **FTP** сервері |
| *#!/usr/bin/env perl*    *# List files on FTP server*  *#*  *# Usage:*  *# ./ftp.pl speedtest.tele2.net anonymous super\_hard\_password*  *# Listing files on server...*  *# 1000GB.zip*  *# 100GB.zip*  *# 100KB.zip*  *# 100MB.zip*  *# 10GB.zip*  *# 10MB.zip*  *# 1GB.zip*  *# 1KB.zip*  *# 1MB.zip*  *# 200MB.zip*  *# 20MB.zip*  *# 2MB.zip*  *# 3MB.zip*  *# 500MB.zip*  *# 50MB.zip*  *# 512KB.zip*  *# 5MB.zip*  *# upload*    **use** strict;  **use** warnings;  **use** feature "say";  **use** Net::FTP;    my ($host, $login, $password) = @**ARGV**;  my $ftp = Net::FTP->**new**($host) or die $!;  $ftp->login($login, $password) or die $ftp->message;  say "Listing files on server...";  say join "**\n**", $ftp->ls or die $ftp->message;  $ftp->quit; |

|  |
| --- |
| Архівація файлів в **.tar.gz** |
| *#!/usr/bin/env perl*    *# Archive into .tar.gz*  *#*  *# Usage:*  *# $ ./tar.pl test.tar.gz ls.pl ps.pl*  *# Files successfully archived into 'test.tar.gz'*  *#*  *# $ tar tf test.tar.gz*  *# ping.pl*  *# pipe.pl*    **use** strict;  **use** warnings;  **use** feature "say";  **use** Archive::Tar;    my $file = shift;  my $tar = Archive::Tar->**new**;  $tar->add\_files(@**ARGV**);  $tar->write($file, COMPRESS\_GZIP);  say "Files successfully archived into '$file'"; |

|  |
| --- |
| **HTTP POST** запит на сервер |
| *#!/usr/bin/env perl*    *# Make HTTP POST request*  *#*  *# Usage:*  *# $ ./post.pl http://jsonplaceholder.typicode.com/posts \*  *# '{"title": "my\_title", "body": "my\_body", "userId": 10}'*  *# {*  *# "body": "my\_body",*  *# "title": "my\_title",*  *# "userId": "10",*  *# "id": 101*  *# }*    **use** strict;  **use** warnings;  **use** HTTP::Tiny;  **use** JSON::MaybeXS qw(decode\_json);    my $response = HTTP::Tiny->new->post\_form(shift, decode\_json shift);  if ($response->{success}) {  print $response->{content};  } else {  die "Failed with reason: $response->{reason}";  } |

Використана література

1. Automating System Administration with Perl, 2nd Edition by David N. Blank-Edelman
2. Perl for system administrators by David N. Blank-Edelman, First edition, published July 2000.
3. Wikipedia стаття про Perl - <https://uk.wikipedia.org/wiki/Perl>
4. Офіційний сайт Perl - [https://www.perl.com](https://www.perl.com/)
5. Онлайн документація - [https://perldoc.perl.org](https://perldoc.perl.org/)
6. Статті на тематику Perl - [https://www.perlmonks.org](https://www.perlmonks.org/)
7. Онлайн туторіал - [https://learn.perl.org](https://learn.perl.org/)
8. Альтренативне джерело статей про Perl - [https://perlmaven.com](https://perlmaven.com/)