**Przedmiot:** Systemy operacyjne - laboratorium

**Tematyka:** Podstawy automatyzacji skryptowej

**Informacje wstępne**

W ramach przygotowań do laboratorium:

1. Przygotować system operacyjny Linux, np.: Linux Mint Cinnamon: https://www.linuxmint.com/
   1. Wykonać instalację natywnie (bezpośrednio na dysku twardym), lub
   2. Wykonać instalację w środowisku wirtualnym, np. VirtualBox (https://www.virtualbox.org/).
   3. Instrukcja dla Mac M1: https://www.youtube.com/watch?v=1WWj6qoWhJw
2. Przygotować konto ssh na serwerze uczelnianym z systemem linux/unix:
   1. Adres Panelu: https://panel.agh.edu.pl/
   2. Dostęp VPN: https://pomoc-it.agh.edu.pl/vpn-zdalny-dostep-do-sieci/
   3. Dostęp SSH:

https://pomoc-it.agh.edu.pl/konta-unix/instrukcje/zakladanie-konta-unix/

Materiał do samodzielnego przepracowania:

1. Dokumentację w systemie linux możemy odczytać programem ‘man’:

$ man <nazwa\_polecenia\_lub\_programu>

Z dokumentacji wychodzimy klawiszem ‘q’.

1. Zapoznać się z poleceniami: **pwd, cd, ls, cat, cp, mv, mkdir, rm, touch, locate, find, grep, df, du, head, tail, diff, tar, chmod, chown, id, jobs, kill, ping, wget, history, man, echo, zip, unzip, hostname, useradd, userdel, curl, df, diff, echo, exit, finger, free, grep, groups, less, passwd, ping, shutdown, ssh, reboot, sudo, top, uname, w, whoami**
2. Zapoznać się z instrukcjami skryptowymi: **if, for, while, until,** etc.
3. Zapoznać się z powłokami: **bash**, zsh, tcsh, itp.
4. Zapoznać się z edytorami: **vim**, pico, etc.

Pomoc w nauce pisania skryptów:

* https://www.tldp.org/LDP/abs/html/

**Sprawozdania**

Sprawozdania z wykonywanych ćwiczeń laboratoryjnych nie są wymagane.

**Rozwiązania zadań**

Rozwiązania zadań laboratoryjnych proszę umieszczać na stronie kursu UPEL.

**Zestaw zadań**

Zadania bez gwiazdki są obowiązkowe dla wszystkich (zad. 1, 2, 3, 5, 6 i 9).

Zadania z jedną gwiazdką są obowiązkowe tylko dla kierunku Cyberbezpieczeństwo (zad. 7).

Zadania z podwójną gwiazdką są nieobowiązkowe dla obu kierunków (zad. 4 i 8).

**Zadanie 1.**

Napisz skrypt, którego wywołanie jest następujące:

$ ./skrypt01.sh <katalog> <wielkość>

Skrypt w podanym katalogu (i podkatalogach) wyszykuje najnowszy plik, jednocześnie większy niż podana wielkość w bajtach.

**Zadanie 2.**

Napisz skrypt, którego wywołanie jest następujące:

$ ./skrypt02.sh <min> <max>

Skrypt losuje liczbę całkowitą z podanego zakresu i umożliwia wykonanie tradycyjnej rozgrywki “zgadnij liczbę”. Proszę sprawdzić, czy min>max. Proszę ograniczyć liczbę zgadywań.

**Zadanie 3.**

Napisz skrypt, którego wywołanie jest następujące:

$ ./skrypt03.sh <podstawa> <potega>

Skrypt dla zadanych wartości oblicza potęgę liczby.

(jedno rozwiązanie operatorem, jedno rozwiązanie pętlą)

**Zadanie 4. \*\***

Napisz skrypt, którego wywołanie jest następujące:

$ ./skrypt04.sh <dokument.pdf> <podpis.png> <x> <y>

Skrypt wykorzysta oprogramowanie ImageMagick i w pliku <dokument.pdf> nałoży zawartość pliku <podpis.png> w pozycji <x> i <y>.

**(brak możliwości zrealizowania w laboratorium uczelnianym)**

**(uwaga do zadania: podpis rozumiec nalezy jako dowolny napis kolorem np. Niebieski, czarnym, lub innym ale z przezroczystym tlem - dlatego plik jest png)**

**Zadanie 9.**

Napisz skrypt, którego wywołanie jest następujące:

$ ./skrypt09.sh <n>

Skrypt wypisuje <n> elementów ciągu Fibonacciego.

**Zadanie 5.**

Napisać skrypt, który wczytuje dowolną liczbę parametrów i wypisuje je w odwrotnej kolejności.

Przykład:

$ ./skrypt05.sh a b 1 2

2 1 b a

***Wskazówki do rozwiązania***

#!/bin/bash

# $\* - lista parametrów rozdzielonych separatorem

# $@ - lista parametrów bez separatorów

# $# - liczba parametrów

# wzrokowo między $\* a $@ nie widać różnicy

echo $\*

echo $@

echo $#

# wybierzmy jeden format zapisu i zapiszemy do zmiennej 'params'

# nawiasami robimy tablicę

params=($\*)

# zapamiętajmy też liczbę parametrów:

n=$#

# wypisanie wszystkich parametrów:

echo ${params[@]}

echo ${params[\*]}

# wypisanie wybranego parametru:

echo ${params[0]}

echo ${params[1]}

echo ${params[2]}

# wypisanie po kolei parametrów:

for (( i=$n-1; i>=0; i-- ))

do

echo -n "${params[$i]} "

done

**Zadanie 6.**

Napisz skrypty grające w “papier-nożyce-kamień”:

* Jeden skrypt (serwer.sh) będzie “serwerem”, który:
  + Dla zadanego parametru wywołania, na przykład:
    - $ ./serwer.sh 10
    - Wykona zadaną liczbę gier “papier-nożyce-kamień” (w podanym przykłądzie to będzie 10 razy) z sktyptami opisanymi dalej (gracz1.sh i gracz2.sh) poprzez opisany poniżej algorytm:
      * Utwórz plik ‘komenda.txt’, a w nim zapisz komendę: ‘start’
      * Odczekaj, aż pojawią się oba pliki: los1.txt i los2.txt, a następnie:
        + Odczekaj 0.1 [s]
        + Skasuj plik ‘komenda.txt’
        + Pobierz z obu plików los1.txt i los2.txt zawartość
        + Rozstrzygnij wynik i zapisz go do gra.log
        + Skasuj pliki los1.txt i los2.txt
    - Po wykonaniu zadanej liczby gier podaj sumaryczny wynik
    - Do pliku ‘komenda.txt’ wpisz komendę: ‘stop’
    - Odczekaj 1 [s]
    - Skasuj plik ‘komenda.txt’
    - Zakończ działanie.
* Dwa takie same skrypty (gracz1.sh i gracz2.sh), które będą graczami:
  + Sprawdzają istnienie pliku ‘komenda.txt’ oraz jego zawartość:
    - Jeśli jest w nim ‘start’ i jednocześnie nie ma pliku odpowiednio los1.txt lub los2.txt, to skrypt losuje ‘papier-nożyce-kamień’ i wylosowany wynik zapisuje odpowiednio w los1.txt lub los2.txt,
    - Jeśli jest w nim ‘stop’, zakończ działanie.

Jaki będzie wynik gry z parametrem 1000 ?

Czy potrafisz zmodyfikować plik serwer.sh tak, aby sam uruchamiał graczy ?

Czy spróbujesz rozbudować serwer.sh tak, aby także mógł obsługiwać dowolną liczbę graczy też podawaną jako parametr, np.: $ ./serwer.sh <liczba graczy> <liczba gier>

(nowe) Proszę uruchomić test dla 10 graczy i kolejno 10, 100 i 1000 rozgrywek, a następnie policzyć średnią i odchylenie standardowe wyników. Proszę o interpretację otrzymanych wartości.

**Zadanie 7. \***

Zadanie polega na usprawnieniu obsługi połączenia VPN do AGH:

https://pomoc-it.agh.edu.pl/vpn-zdalny-dostep-do-sieci/

Najprostrzym rozwiązaniem jest konfiguracja ustawień sieciowych do połączeń VPN z wykorzystaniem GUI dostępnego w systemach okienkowych.

W przypadku systemu bez interfejsu graficznego można posłużyć się przykładem:

sudo openvpn ~/path/to/ovpn/VPN-AGH.2022.ovpn; echo -n "Opening VPN: "; while [ `ifconfig | grep tun0 | wc -l` -eq 0 ]; do echo -en "."; sleep 1; done; ssh user@student.agh.edu.pl; echo "Closing VPN..."; sudo kill `cat /tmp/vpn.agh.pid`

Powyższe polecenie ustanowi VPN do AGH i utrzyma tak długo, aż nie zostanie przerwane ctrl+c.

Monitorowanie stanu połączenia VPN:

$ journalctl -u NetworkManager -ef

Proszę napisać skrypt, lub rozbudować powyższe polecenie o następujące mechanizmy:

* Obsługę zerwania i zawieszenia połączenia
* Rozpoznanie istniejącego połączenia
* Eleganckie zamknięcie połączenia.
* Log połączeń (nawiązanie, status, zakończenie, czas trwania, sposób zamknięcia, itp.)

**Zadanie 8. \*\***

Napisz skrypt odnajdujący w sposób numeryczny (nie algebraiczny, a optymalizacyjny) miejsca zerowe zadanej fukcji wielomianowej (czyli szukamy takiego x, dla którego f(x0 == 0). Skrypt przyjmuje parametry w sposób następujący:

$ ./skrypt08.sh n a b c d…

* n - stopień wielomianu
* a, b, c, … - współczynniki wielomianu

Przykład:

$ ./skrypt08.sh 5 2 6 -5 0 8 -3

To wielomian 5-tego stopnia:

f(x) = 2 \* x ^ 5 + 6 \* x ^ 4 - 5 \* x ^ 3 + 0 \* x ^ 2 + 8 \* x - 3