Wstęp do użytkowania systemu operacyjnego LINUX

Magdalena Kuich

Kontakt: mkuich@fuw.edu.pl

Mteriały: www.fuw.edu.pl/~mkuich/tik2020/

Organizacja zajęć - informacje ogólne

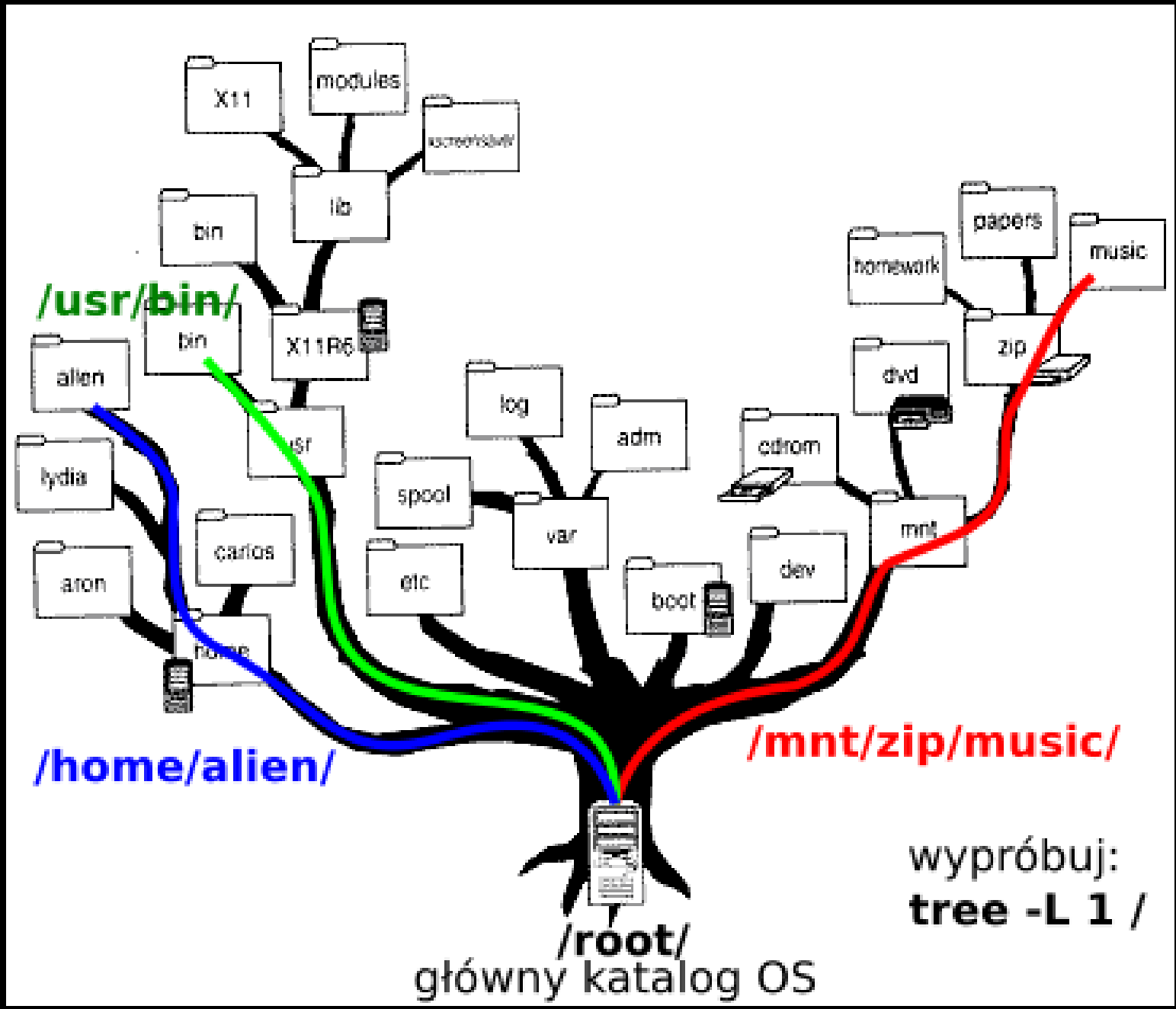
- Zajęcia odbywają się w trybie stacjonarnym w dwóch salach (gr. A i B)
- Obecność na zajęciach nie jest obowiązkowa - obecność na kolokwiach jest!
- Obecność jest sprawdzana na każdych zajęciach informacyjnie
- Materiały z zajęć będą dostępne na stronie prowadzącego
- **RESPEKTUJEMY ZASADY BEZPIECZEŃSTWA SANITARNEGO:**
  - $\rightarrow$ **zakrywaj usta i nos**
  - $\rightarrow$ **często dezynfekuj ręce**
  - $\rightarrow$ **zachowaj bezpieczny dystans od koleżanki/kolegi/prowadzącego**
  - $\rightarrow$ **pilnujmy się nawzajem**
  - $\rightarrow$ **jeśli zaobserwujesz u siebie objawy grypopodobne - poinformuj prowadzącego (najlepiej mailowo) i nie przychodź na zajęcia**
- Rozkład zajęć:

temat	liczba zajęć	zaliczenie
LINUX	2 zajęcia	–
LATEX	2 zajęcia	praca domowa
PYTHON	3 (+1) zajęcia	kolokwium
Mathematica	3 (+1) zajęcia	kolokwium
kolokwium poprawkowe	1 zajęcia	1 wybrane kolokwium

LINUX OS 1

- Unixopodobny system operacyjny oparty na jądrze LINUX
  - → wielozadaniowość, wielowątkowość, wielobieżność
  - → biblioteki współdzielone, dynamiczne, statyczne
  - → hierarchiczny system plików
  - → ...
- Oprogramowanie jest otwarte i wolne - każdy może z niego korzystać za darmo i rozwijać na własną rękę
- Interfejsy:
  - → graficzny - zarządzamy komputerem poprzez odpowiednie menadżery, różne środowiska graficzne (Xfce, KDE, GNOME, Unity, Mate ...) ⇒ Zad.1
  - → wiersza poleceń - zarządzamy komputerem za pomocą koment wysyłanych w terminalu/konsoli ⇒ Zad.2
  - → w trybie graficznym mamy możemy korzystać z emulatora tyrgu tekstowego - terminala ⇒ Zad.3
- Najbardziej znane dystrybucje (rodzaje):
  - → Mint LINUX
  - → Ubuntu
  - → Fedora, CentOS
  - → Debian
  - → Arch Linux

Hierarchiczny system plików



Katalogi i serwery

- Każdy użytkownik w systemie Linux ma przypisany swój **katalog domowy** np.: `/dmj/2020/ab123456/`
  - → dla wygody wprowadzono skrót: `~/` = `/dmj/2020/ab123456/`
  - → to podstawode miejsce pracy na systemie Lixux
  - → miejce przeznaczone na wszystkie nasze dane, indywidualne pliki konfiguracyjne itp.
  - → pojemność katalogu domowego na studenckich serwerach wydziałowych jest ograniczona do 200 MB
- Pracując na serwerach `okwf`, tj. `tempac` lub `primus`, mamy do dyspozycji także **katalog roboczy**:
  - → `/mnt/work/2020/ab123456`
  - → jest on podpięty (podlinkowany) w katalogu domowym: `~/_work_/`
  - → pojemność katalogu roboczego na studenckich serwerach wydziałowych wynosi 1 GB
- Studenckie serwery wydziałowe:
  - → `tempac.fuw.edu.pl` - jest dostępny do logowania (SSH) ze świata, umożliwia studentom dostęp do zawartości własnych (o tej samej zawartości, którą mają na komputerach w pracowni komputerowej) kont spoza Wydziału
  - → `primus.okwf.fuw.edu.pl` - dostępny do logowania z sieci Wydziału, można na nim uruchamiać dłuższe zadania obliczeniowe

Terminal i program

- W terminalu możemy uruchamiać programy, wykonywać polecenia na plikach i katalogach, tworzyć i usuwać pliki i katalogi, edytować pliki tekstowe, etc.
  - → każde poprawne polecenie w wpisane w terminalu i zatwierdzone poprzez wciśnienie przycisku `Enter` uruchamia jakiś program/programik/skrypt, np.: `gedit` - edytor tekstu
  - → uruchomiony program "zamraża" terminal, czyli uniemożliwia wpisywanie kolejnych poleceń
  - → aby "odblokować" terminal można zastosować:
    `Ctrl+C` - zamyka aktualnie uruchomiony program `Ctrl+Z` - zawiesza aktualnie uruchomiony program i przywraca terminal. Można wtedy wpisać w terminalu komendę `bg` - *background*, która wprowadza zawieszony program do pracy w "tle", a aby przywrócić program z pracy w tle można wtedy wpisać w terminalu komendę `fg` - *foreground*.
  - → Można też uruchomić program, wpisując znak & (ampersant) po nazwie programu, co sprawia, że program działa w "tle" i okno terminala jest nadal aktywne:
    `gedit&`

Dokumentacja podstawowych poleceń i procesy

- Polecenie `man` pozwala na przeglądanie dokumentacji wszystkich programów na naszym linuksie.
  - → `man nazwa_polecenia`, np. `man gedit`
  - → z man'a wychodzimy przyciskiem `q`
  - → w man'ie szukamy przyciskiem `/`
  - → następny wynik wyszukiwania jest dostępny po kliknięciu `n`
- Polecenia `ps -e` lub `ps -ef` wyświetlają listę wszystkich procesów działających na danym komputerze
  - → każdy proces ma przypisane numer procesu (PID)
  - → każdy proces jest wywołany przez użytkownika
  - → każdy proces można wyłączyć znając jego PID i korzystając z polecenia `kill` np. `kill 618087`

Katalogi i pliki - podstawowe polecenia 1

- `tree` - wyświetla strukturę drzewa katalogów
- `pwd` - *print working directory*, wyświetla aktualny katalog roboczy
- `cd` - *change directory* pozwala zmienić katalog roboczy
  - → `cd ścieżka_do_katalogu` - wchodzimy do katalogu
  - → `cd ..` - wychodzimy o poziom wyżej
  - → `cd` - wychodzimy do katalogu domowego
- `ls ścieżka_do_katalogu` - *list content*, wyświetla zawartości katalogu
  - → `ls -a` - pozwala na wyświetlenie plików „ukrytych”
  - → `ls -l` - wyświetla szczegółowe dane plików
  - → `ls -R` - listuje katalogi „rekurencyjnie”
- `mkdir nazwa_katalogu` - tworzy katalogu o nazwie nazwa_katalogu
- `touch nazwa_pliku` - tworzy pusty plik o nazwie nazwa_pliku

Zadania 4–11

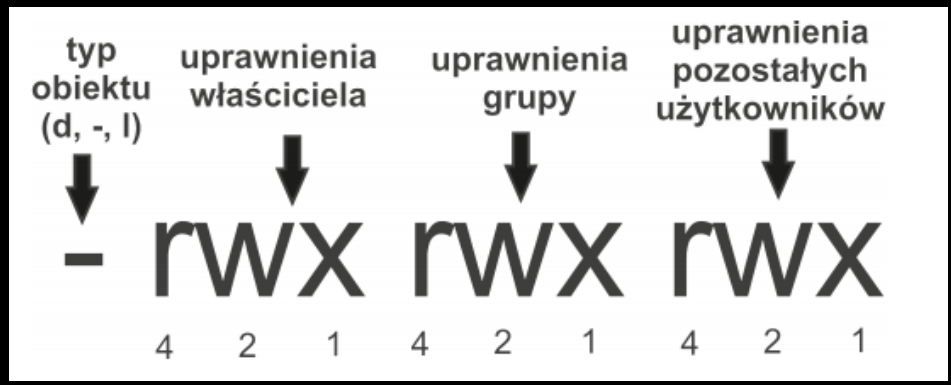
Katalogi i pliki - podstawowe polecenia2

- `rmdir nazwa_katalogu` usuwa pusty katalog o nazwie nazwa_katalogu
- `rm nazwa_pliku` - usuwa plik o nazwie nazwa_pliku
  - → `rm -r nazwa_katalogu` - usuwa z zawatrością
  - → `rm -f` - "forsuje" usuwanie plików/argumentów
  - → `rm -i` - wymusza zapytanie o usunięcie plików/argumentów
- `cp ścieżka1 ścieżka2` - *copy*, pozwala skopiować argument1 w miejsce wskazane przez argument2
  - → `cp -r katalog1 katalog2` - kopiowanie katalogów z zawartością
  - → `cp katalog/plik ./` - kopiowanie pliku, do katalogu w którym się znajdujemy
    `.` lub `./` - oznaczają "tu gdzie jesteś"
- `mv ścieżka1 ścieżka2` - *move*, pozwala przenieść argument1 w miejsce wskazane przez argument2

Zadania 12–14

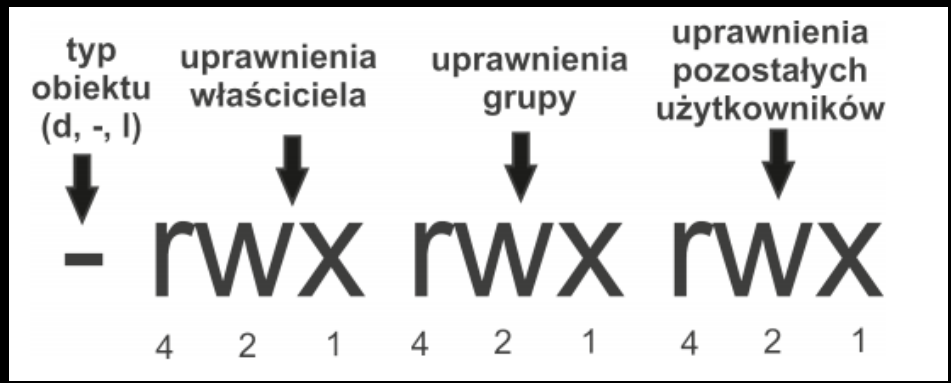
Prawa dostępu

- Każdy plik w systemie linux ma określone prawa dostępu.
- Istnieją trzy podstawowe prawa (poniżej w zapisie symbolicznym):
  - → `r` - *read*, uprawnia do przeczytania pliku
  - → `w` - *write*, uprawnia do zapisu i modyfikowania pliku
  - → `x` - *execute*, uprawnia do wykonania/uruchomienia argumentu (najczęściej skryptu lub programu)
- Każdy z tych atrybutów można ustawić dla właściciela pliku (u-user), innych z grupy (g-group) lub wszystkich innych użytkowników (oothers). Każdy użytkownik może należeć do wielu grup! Aby poznać swoje grupy można skorzystać z polecenia `id`.
- Dla katalogów artybut `x` pozwala na wejście do katalogu lub dowolnego podkatalogu, a `r` na zlistowanie zawartości.



Prawa dostępu

- Typ obiektu:
  - → `d` - *directory*, katalog
  - → `-` - plik
  - → `l` - link
- Prawa dostępu można opisać z pomocą liczb całkowitych z zakresu 0-7. W takim zapisie mamy odpowiednie przyporządkowania:
  - → `x` = 1 - *execute*, uprawnia do wykonania
  - → `w` = 2 - *write*, uprawnia do zapisu i modyfikowania pliku
  - → `r` = 4 - *read*, uprawnia do przeczytania pliku
- Konkretne prawa dostępu uzyskuje się dodając do siebie 1, 2 i 4. Np:
  - → 1+2 = 3 : eXecute + Write
  - → 1+4 = 5 : eXecute + Read
  - → 1+2+4 = 7 : eXecute + Write + Read



Nadawanie uprawnień dostępu - polecenie `chmod`

- Polecenie chmod pozwala na ustawienie praw dostępu dla pliku lub katalogu
- `chmod` używamy w postaci: `chmod <przywileje> nazwa_pliku`, np.:
  - → `chmod go=rx plik.txt` - ustawia uprawnienia do odczytu i wykonywania dla grupy oraz pozostałych użytkowników, odbiera wcześniej istniejące uprawnienia
  - → `chmod u+x,o+r plik.txt` - dodaje prawa już istniejących: do wykonywania dla właściciela i prawa odczytu dla innych
  - → `chmod a+rwx plik.txt` - nadaje wszystkim wszystkie możliwe uprawnienia'plik.txt',
- W zapisie numerycznym przywileje określają 3 cyfry - po kolei dla właściciela, grupy i wszystkich innych, np.:
  - → `chmod 744 nazwa_pliku` - ustawia pełen prawa dla właściciela i prawa odczytu dla innych
  - → `chmod -R 777 /home/user` - wszyscy będą mogli zmieniać zawartość katalogu `/home/user` oraz jego podkatalogów, jak też czytać go i wykonywać zawarte w nim pliki
- Opcja `-R` pozwala (jak zwykle) działać rekurencyjnie na podkatalogach

Zadania 15–18

Wzorce

- Znak $*$ zastępuje dowolną liczbę dowolnych znaków
- Znak ? zastępuje dokładnie jeden dowolny znak.
- Używając [] można określić zakres znaków które mogą się pojawić. Np:
  - → `[abc]` - zastępuje a lub b lub c.
  - → `[a-c]` - zastępuje od a do c
  - → `[0-9]` - zastępuje dowolną cyfrę
  - → `[!a-c]` - zastępuje dowolny znak poza wymienionymi
  - → `{fraza1,fraza2}` - jeden z ciągów znaków oddzielonych przecinkami.

Zadania 19–20

Wyświetlanie plików i zliczanie słów

- Polecenia służące do wyświetlania całej zawartości plików:
  - → `less`, np.: `less plik.txt`
  - → `more`, np.: `more plik.txt`
  - → `cat`, np.: `cat plik.txt`
- Polecenia służące do wyświetlania początków lub końców plików:
  - → `head`, np.: `head plik.txt` - drukuje początek pliku (domyślnie pierwszych 10 wierszy)
  - → `tail`, np.: `tail plik.txt` - drukuje koniec pliku (domyslnie ostatnich 10 wierszy)
- `wc` - *word count*, drukuje liczbę linii, słów i znaków w tekście
  - → można wywłać do bez opcji lub z opcjami `-l`, `-w` lub `c`
  - → np.: `wc plik.txt`

Zadania 21–23

Standardowe wejście/wyjście i operatory przekazania

- Znaki które wpisujemy z klawiatury trafiają w systemie do tzw. „**standardowego wejścia**". Program odpowiada na tzw. „**standardowe wyjście**", które wyświetlane jest na ekranie.
- Operator $<$ podaje na std wejście zawartość pliku.
- Operator |, tzw. *pipe*, pozwala przekierować std wyjście na std wejście
- Dane ze standardowego wyjścia można zapisać do pliku używając znaków $>$ lub $>>$
  - $\rightarrow$ $>$ tworzy nowy plik i zapisuje do niego wynik działania programu. Jeżeli plik już istnieje zostaje zastąpiony!
  - $\rightarrow$ $>>$ działa podobnie, ale jeżeli plik już istnieje, to wynik zostaje dopisany.

Zadania 24–26

Wyszukiwanie i sortowanie

- `find <ścieżka> <warunki>`, gdzie:
  - → `<ścieżka>` - w tym katalogu i jego podkatalogach zostanie dokonane przeszukanie
  - → `<warunki>` - zestaw warunków precyzujących jakie pliki mają być wyszukane.
    - → `-name wzorzec` - pozwala sprecyzować nazwę (działają znaki specjalne `*`,`?` i `[]`). Wzorzec trzeba podać w „”
    - → `-size n[ck]` - rozmiar: `c` - w bajtach, `k` - w kilobajtach. (`+n` -rozmiar większy niż, `-n` - mniejszy niż)
    - → `-type` - wyszukiwanie “po typie” plików.
- `grep <wzorzec> <plik/wejście>` - przeszukuje dane na wejściu w poszukiwaniu linii ze wzorcem
  - → wzorzec trzeba podać w „”, (działają znaki specjalne `*`,`?` i `[]`)
- `du` - oblicza i wypisuje informacje o ilości miejsca na dysku zajmowanego przez pliki w poszczególnych katalogach
- `sort` - sortuje dane wg wybranych kryteriów

Zadania 27–30

Komunikacja zdalna - `ssh`

- `ssh` - *secure shell*, to protokół komunikacyjny stosowany w sieciach TCP/IP. Służy do zdalnego i szyfrowanego łączenia terminalowego z komputerami.
- Poprzez `ssh` głównie pracujemy w trybie tekstowym (tj. w terminalu), ale możemy przesyłać także tryb graficzny, jeśli wywołamy połączenie z opcją `-X` lub `-Y`.
- Łącząc się przez `ssh` musimy podać login użytkownika, serwer (host) oraz opcje (np. transjisja grafiki lub nr portu)
- Domyślnie łączmy się przez port 22, ale można wymusić łączenie przez inne porty.
- Chcąc połączyć się z pracownią komputerową na wydziale z komputera spoza sieci wydziałowej:
  `ssh -X ab123456 (at) tempac.okwf.fuw.edu.pl`
  - → `ab123456` - mój login
  - → `tempac.okwf.fuw.edu.pl` - nazwa host'a