Info 3 - Wykład 1

S. W. Gepner
sgepner@meil.pw.edu.pl

March 6, 2020

1 Plan zajęć

- System operacyjny - wprowadzenie
- Linux, krótka historia
  - Pierwsze logowanie
  - Przydatne komendy
- Praca z powłoką BASH
- Programowanie w BASH
- Relacyjne bazy danych
- LateX - jak tworzyć piękne dokumenty i nie tylko
- Programowanie równoległe
  - MPI - wprowadzenie
  - podstawowe pojęcia
  - wątki a procesy
  - Systemy SMP (Symmetric MultiProcessing)
  - MPP (Massively Parallel Processing)
  - skalowanie
  - jak myśleć "równolegle"
  - komunikacja i przestoje
- Praca z systemami HPC
  - Model systemu HPC
  - Kolejka i dostęp do zasobów
- Interfejsy sieciowe

1.1 Źródła i literatura:

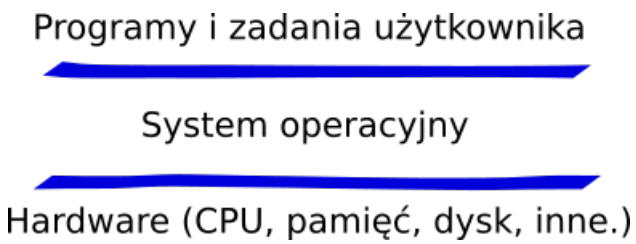
Materiały powstały w oparciu o następujące źródła:

- https://www.doc.ic.ac.uk/~{}wjk/UnixIntro/
- https://www.levenez.com/unix/
- https://pdos.csail.mit.edu/6.828/2014/xv6.html
- https://pdos.csail.mit.edu/6.828/2014/xv6/book-rev8.pdf

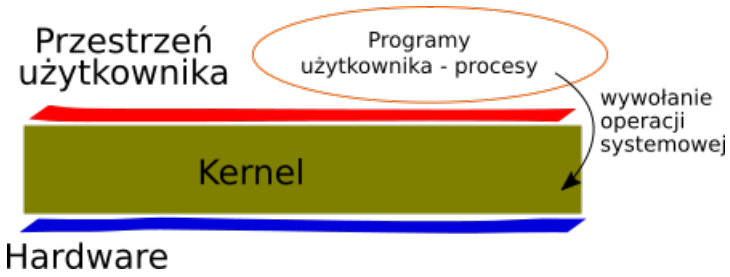
---

2 System operacyjny - wprowadzenie

- Warstwa pośrednia pomiędzy zadaniami użytkownika (programami), a zasobami sprzętowymi (Hardware) komputera
- Zarządza działaniem oraz dostępem do sprzętu (CPU, pamięci, urządzeń I/O itd.)
- Wprowadza poziom abstrakcji w pracy z komputerem, tak aby działające programy "nie przejmowały" się na jakim sprzęcie pracują i jak go między siebie dzielą
- Można patrzeć na OS jak na zwielokrotniacz (multiplexer) sprzętu pozwalający na działanie wielu programów (przynajmniej pozornie) na raz
- Zapewnia kontrolowany i standardowy dostęp do zasobów i komunikacji między programami (procesami)



SO zapewnia zestaw usług poprzez interfejs. Z jednej strony interfejs powinien być prosty i dobrze zdefiniowany, by zapewnić łatwość i jednoznaczność implementacji. Jednocześnie jest pokusa aby interfejs taki zapewniał złożone (nowoczesne?) usługi. Może to prowadzić do problemów z prawidłowym zaprojektowaniem takiego systemu.



Działający programy ma swój obszar pamięci, który zawiera zestaw instrukcji do wykonania, dane i stos (stack). Instrukcje to po prostu zadania jakie ma zrealizować program, dane to zmienne przechowywane w czasie działania. Stos służy zaś do organizacji wywołań procedur.

SO ma zapewnić **bezpieczeństwo** działania, **wydajność** wykorzystania zasobów i **abstrakcyjność** w dostępie.

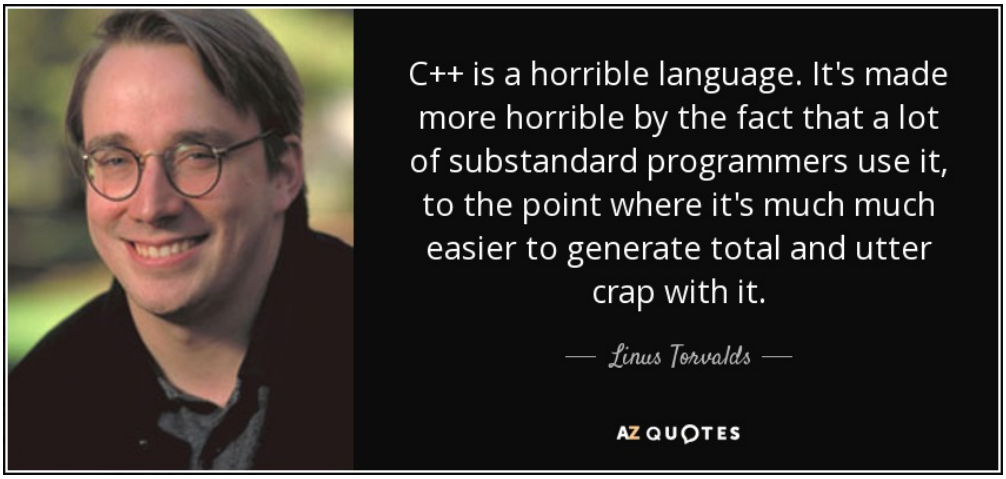
- **Bezpieczeństwo**: Kontroluje dostęp do zasobów, np. dopuści jeden program do korzystania z drukarki.
- **Wydajność**: Poza zapewne bardziej zaawansowanymi mechanizmami, może zawiesić wykonanie procesów oczekujących np. na operacje wejścia/wyjścia i w tym czasie wykonać inne zadania.
- **Abstrakcyjność**: odseparowanie oprogramowania i użytkownika od niuansów sprzętu (pliki zamiast dysku).

3 Bardzo krótka historia Linuxa

Dłuższą wersję można znaleźć tu: https://www.levenez.com/unix/

Mało wyraźne, bo duże drzewo genealogiczne systemów z rodziny UNIX.
https://www.levenez.com/unix/.

- 1960's MULTICS (Multiplexed Information and Computing System) multi-user, multi-tasking
- 1969 UNICS (Uniplexed Information and Computing System) później UNIX by Ken Thompson at Bell Labs
  - zasoby są drogie, wykorzystuje więc krótkie nazwy komend by zaoszczędzić miejsce i czas wykonania, stąd tradycyjne komendy Linuxowe są krótkie (cp, rm, cat itd.)
- 1973/4 Ken Thompson i Dennis Ritchie (także z Bell Labs - twórca języka C) Fifth Edition of UNIX kernel napisana w C.
- 1978 Wraz z 7 edycją następuje podział na SysV i BSD (Berkeley Software Distribution)
- 1991 Linus Torvalds, student z Finlandii wypuszcza wolnego UNIXA - LINUX. System łączy w sobie cechy SysV i BSD. Stara się też by system był zgodny ze standardem POSIX (Portable Operating System Interface).



3.1 Linux

- Open Source - dostępny kod źródłowy, każdy może go poprawić, zmienić i "oddać" światu
- To co zaczęło się jako hobbystyczne zajęcie jednego człowieka, jest obecnie rozwijane przez rzesze programistów, ale też przez komercyjne firmy
- Obecnie dostępne jest wiele dystrybucji (Debian, Slackware, RedHat, Centos ...). Dystrybucja obejmuje wersję kernela, zestaw aplikacji systemowych, GUI i aplikacje użytkownika.

3.2 Kernel

- Zawiera sterowniki dla sprzętu, zarządza pamięcią i procesorem, wspiera rózne systemy plików. implementuje większość wywołań systemowych z SysV, BSD i standardu POSIX. Kernel ładowany jest do pamięci przy starcie systemu a znaleźć go można w pliku /boot/vmlinuz natomiast źródła są w katalogu /usr/src/linux.

3.3 Powłoka (Shell) i GUI

- Dostępne są w zasadzie dwie metody interakcji z systemem. Graficzny dostęp (GUI) przez interfejsy graficzne takie jak KDE, GNOME, XFCE i inne oraz dostęp poprzez powłokę (shell). Powłoka to interpreter poleceń dla systemu. Zawiera zestaw możliwych do wykonania operacji a także możliwość ich łączenia dla bardziej złożonych zadań. Wraz z Unix 7 pojawiła się powłoka **sh** (Bourne shell). Otwarta implementacja powstała w 1989 jako Bourne-Again Shell - **bash** i jest obecnie najpopularniejszą powłoką systemową.

3.4 Komendy systemowe

- *ls, cp, grep, awk, sed, bc, wc, more* i inne. Narzędzia systemowe pozwalające na pracę z systemem. Zaprojektowane by wykonywać jedno zadania bardzo dobrze oraz by możliwe było ich wywołanie "łączone". Pozwala to na rozwiązanie wielu problemów bez konieczności pisania dedykowanych programów.
- Serwisy i usługi systemowe zapewnione są przez *demony* (Disk And Execution MONitor). Programy te powoływane są do życia przy starcie systemu i większość czasu spędzają w uśpieniu czekając na odpowiednie przypisane im zdarzenie.

3.5 Aplikacje

- Poza komendami systemowymi typowa dystrybucja Linuxa posiada wiele użytecznych programów. Takich jak kompilatory (gcc, g++ inne), edytory (vi, vim). I wiele innych.

---

4 Praca z systemem Linux

Interfejs graficzny jest bardzo wygodny do codziennej pracy z komputerem. Pozwolił też na zaprowadzenie komputerów pod strzechy. Nie jest on jednak dostępny zawsze i wszędzie. Są też sytuacje, gdzie użycie GUI było by po prostu kontr-produktywne. Jest tak np. w przypadku dużych, połączonych szybką siecią systemów obliczeniowych (HPC – High Performance Computing), gdzie wiele komputerów współpracuje ze sobą. W takiej sytuacji jedynym dostępnym środowiskiem użytkownika jest "czarny ekran i białe literki", czyli powłoka (konsola - shell). W naszym przypadku będzie to **bash**.

Systemy z rodziny UNIXów są z założenia systemami multi-user, czyli takimi w których na raz pracować może wielu użytkowników. Jest to istotna różnica w porównaniu do systemów wywodzących się z rodziny systemów DOS (Disk Operating System), które z założenia miały działać na komputerach osobistych jednego użytkownika (stare ale też nowe Windows?).

Ze względu na możliwość pracy wielu użytkowników systemy z rodziny UNIX wymagają do pracy nazwy użytkownika - login, oraz hasła. Stosują też system *praw dostępu* do plików (i zasobów) tak aby możliwe było zachowanie prywatności i zapewnienie bezpieczeństwa pracy.

4.1 Logowanie i zmiana hasła

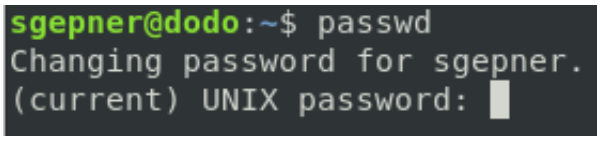
Po uruchomieniu komputera w trybie dostępu tekstowego, bądź przy próbie zalogowania zdalnego użytkownik dysponujący loginem i hasłem może spróbować zalogować się do systemu.

```
In [ ]: login: sgepner
        password:
```

Po zalogowaniu wyświetlona zostanie nazwa użytkownika, nazwa maszyny, katalog w którym się obecnie znajdujemy (~ oznacza katalog domowy użytkownika - o strukturze katalogów powiemy zaraz), oraz znak zachęty (command prompt) $.

Pierwszą operację jaką przeprowadzimy po zalogowaniu będzie zmiana hasła. Użyjemy komendy systemowej *passwd*:

```
In [ ]: passwd
```



Musimy podać swoje dotychczasowe hasło, a następnie zastąpić je nowym (odpowiednio złożonym).

4.2 Struktura polecań w systemie

Najczęstszą praktyką poleceń i programów wykorzystywanych w pracy na systemach unixowych jest następująca struktura polecenia:

```
In [ ]: $ command -options targets
```

gdzie *command* to polecenie, np.: *passwd*, *ls* lub inne, po myślniku (czasem dwóch) następują opcje polecenia następnie elementy na których ma ono zadziałać (jeżeli potrzebne).

4.3 System plików

System plików pozwala na przechowywanie (możliwe, że na nośniku) danych samego systemu, programów, użytkowników, a także plików specjalnych dających dostęp do urządzeń i specjalnych funkcji systemu. W dużym przybliżeniu można powiedzieć, że w zasadzie wszystko jest plikiem.

Elementy przechowywane przez system plików podzielimy na cztery kategorie: * **Zwykłe pliki** zawierające dane użytkownika, systemu czy instrukcje programu. Nie zawierają innych plików w sobie. W odróżnieniu od systemów z rodziny Windows pliki nie muszą posiadać rozszerzenia (np. .exe). Nazwa pliku może być dowolna ale nie powinna zawierać znaków specjalnych (*,? itd.) w szczególności '/'. Odradza się też używania w nazwach spacji.

- **Katalogi** przechowują pliki i inne katalogi.
- **Urządzenia** System udostępnia urządzenia jako pliki.
- **Dowiązania** (linki) działają jak skróty, pozwalając na dostęp do innych plików. Mogą występować w dwóch wersjach. Dowiązania twarde (hard links) są w zasadzie nieodróżnialne od pliku. Dowiązania miękkie (soft links) wskazują na plik do którego wiodą.

4.4 Struktura katalogów

Struktura katalogów ma charakter drzewa i jest bardzo podobna w różnych systemach z rodziny Unix. Wszystko zaczyna się od katalogu głównego '/' (*root*) będącego na najwyższym poziomie drzewa. Zawiera on katalogi systemowe, a w nich zawarte są kolejne, itd.

4.4.1 Najważniejsze katalogi

- / -katalog główny (*root)
- /home - zawiera katalogi użytkowników
- /dev - pliki odpowiadające urządzeniom (dyski, drukarki, inne)
- /bin - aplikacje systemowe
- /usr, /usr/bin, /usr/lib - aplikacje i biblioteki aplikacji do użytku przez użytkowników
- /etc - pliki konfiguracyjne systemu

4.4.2 Określenie położenia

- '/' - katalog główny
- './' - katalog bieżący
- '~' - katalog domowy użytkownika
- '../' - katalog nadrzędny

Przykład drzewa katalogów (wykorzystanie polecenia tree: `tree` - wyświetla drzewo plików, działa rekursywnie, przełącznik -L służy do określenia "głębokości").

```
In [1]: tree -L 1 /
```

Można też dla konkretnego katalogu

```
In [7]: tree -L 1 /home
```

Albo, domyślnie dla katalogu bieżącego

```
In [1]: tree -L 1
```

Albo jeden do góry.

```
In [9]: tree ../ -L 1
```

4.4.3 Nawigacja po katalogach

Ścieżki mogą być zapisywane jako **absolutne** od katalogu głównego '/':
`/home/sgepner/dydaktyka/info3/oceny.xls`
lub odnoszące się do obecnego położenia (**relatywne**), np. z katalogu `dydaktyka` odwołanie do `info3` może wyglądać o tak:
`./info3/oceny.xls`
a do katalogu nadrzędnego tak:
`../info2/oceny.xls`

4.5 Kilka przydatnych poleceń

Note: manual do poszczególnych poleceń może być wywołany przez polecenie `man`

1. `pwd` - print working dir - podaje bieżący katalog
   `man pwd`

```
In [2]: man pwd
```

```
PWD(1)                          User Commands                          PWD(1)

NAME
       pwd - print name of current/working directory

SYNOPSIS
```

```
       pwd [OPTION]...

DESCRIPTION
       Print the full filename of the current working directory.

       -L, --logical
              use PWD from environment, even if it contains symlinks

       -P, --physical
              avoid all symlinks

       --help display this help and exit

       --version
              output version information and exit

       If no option is specified, -P is assumed.

       NOTE:  your shell may have its own version of pwd, which usually super[U+2010]
       sedes the version described here.  Please refer to your  shell's  docu[U+2010]
       mentation for details about the options it supports.

AUTHOR
       Written by Jim Meyering.

REPORTING BUGS
       GNU coreutils online help: <http://www.gnu.org/software/coreutils/>
       Report pwd translation bugs to <http://translationproject.org/team/>

COPYRIGHT
       Copyright  ©  2017  Free Software Foundation, Inc.  License GPLv3+: GNU
       GPL version 3 or later <http://gnu.org/licenses/gpl.html>.
       This is free software: you are free  to  change  and  redistribute  it.
       There is NO WARRANTY, to the extent permitted by law.

SEE ALSO
       getcwd(3)

       Full documentation at: <http://www.gnu.org/software/coreutils/pwd>
       or available locally via: info '(coreutils) pwd invocation'

GNU coreutils 8.28              January 2018                            PWD(1)
```

```
In [3]: pwd
```

```
/home/sgepner/OneDrive/Dydaktyka/Info_3/Wykład
```

2. `ls` - list directory - pokazuje zawartość katalogu. Możliwe są modyfikacje wyświetlanego rezultatu poprzez opcje. Podstawowe przełączniki to `-l` listujący zawartość katalogu wraz z dodatkowymi informacjami i `-a` pokazujący także pliki ukryte. (Spróbuj `ls -l`, `ls -a`, `ls -l`, `ls -la`). Przydatnym przełącznikiem jest też `-h` ponieważ pozwala wyświetlić rozmiary plików w sposób czytelny dla ludzi, zamiast w bajtach.

   Pliki ukryte: plik, którego nazwa zaczyna się od kropki '.' będzie z założenia plikiem ukrytym, tzn. widocznym tylko, jeżeli będziemy tego bardzo chcieli.

```
In [5]: ls -l

total 2908
-rw-rw-r-- 1 sgepner sgepner  110559 mar  5 12:28  burza
-rw-r--r-- 1 sgepner sgepner   13182 mar  3 16:24  fig1.png
-rw-rw-r-- 1 sgepner sgepner   17000 mar  3 16:23  fig1.svg
-rw-rw-r-- 1 sgepner sgepner   16358 mar  3 16:24  fig2.png
-rw-r--r-- 1 sgepner sgepner    2920 mar  3 18:04 'Info 3 - Wykład 1.aux'
-rw-r--r-- 1 sgepner sgepner  241754 mar  6 11:54 'Info 3 - Wykład 1.ipynb'
-rw-r--r-- 1 sgepner sgepner   35451 mar  3 18:04 'Info 3 - Wykład 1.log'
-rw-r--r-- 1 sgepner sgepner     699 mar  3 18:04 'Info 3 - Wykład 1.out'
-rw-r--r-- 1 sgepner sgepner 1227271 mar  3 18:04 'Info 3 - Wykład 1.pdf'
-rw-r--r-- 1 sgepner sgepner   21745 mar  3 18:04 'Info 3 - Wykład 1.synctex.gz'
-rw-rw-r-- 1 sgepner sgepner   24086 mar  3 18:04 'Info 3 - Wykład 1.tex'
-rw-r--r-- 1 sgepner sgepner   65458 mar  3 16:06  quote-c-is-a-horrible-language-it-s-made-more
-rw-r--r-- 1 sgepner sgepner 1166196 mar  3 17:37  unix.png
-rw-r--r-- 1 sgepner sgepner    6150 mar  3 18:03  Zaznaczenie_004.png
-rw-r--r-- 1 sgepner sgepner    1768 mar  3 18:03  Zaznaczenie_005.png
```

Kilka słów o tym co pokazał nam `ls -l`

```
$ ls -l     prawa  dowiązania           rozmiar               nazwa   -rw-rw-r--  1
sgepner sgepner   13182 mar  3 16:24  fig1.png   typ          właściciel i grupa
data i czas modyfikacji
```

- typ: określa typ pliku 'd'-katalog, '-'-zwykły plik, 'l'-dowiązanie, 'b' lub 'c' - urządzenie
- prawa: określa kto i w jaki sposób ma dostęp do pliku. Zawiera 9 symboli opisujących trzy możliwe sposoby interakcji z plikiem dla trzech kategorii użytkowników:
  - Prawa `rwx`:
    - 'r' - prawo do odczytu
    - 'w' - prawo do zapisu
    - 'x' - prawo do wykonania
  - Użytkownicy `ugo`:
    - właściciel (owner) - pierwsze trzy symbole
    - grupa (group) - środkowe trzy
    - inni (others) - inni

3. `cd` - change dir - zmień katalog

```
In [6]: pwd
        cd /home/sgepner/OneDrive/Dydaktyka/Info_3
        pwd
```

```
/home/sgepner/OneDrive/Dydaktyka/Info_3/Wykład
/home/sgepner/OneDrive/Dydaktyka/Info_3
```

4. `mkdir` - tworzenie katalogu

```
In [11]: mkdir katalog
```

```
In [12]: ls
```

```
'Info 3 - Wykład 1.tex~'   listy.odt               'Zasady przedmiotu.pdf'
 katalog                   Wykład
 lista                    'Zasady przedmiotu.odt'
```

```
In [13]: cd katalog
         pwd
```

```
/home/sgepner/OneDrive/Dydaktyka/Info_3/katalog
```

```
In [14]: cd ../
```

5. `rmdir` - kasowanie katalogu. Zadziała tylko jeżeli katalog będzie pusty.

```
In [15]: rmdir katalog
```

```
In [16]: ls
```

```
'Info 3 - Wykład 1.tex~'   listy.odt  'Zasady przedmiotu.odt'
 lista                     Wykład     'Zasady przedmiotu.pdf'
```

6. Operacje na plikach - tworzenie, kopiowanie przenoszenie i kasowanie:
   `touch` - tworzy pusty plik
   `cp` - kopiowanie
   `mv` - przenoszenie / zmiana nazwy
   `rm` - usuwanie + opcje

```
In [17]: mkdir kat
         cd kat
         touch aaa
         pwd
         ls
```

```
/home/sgepner/OneDrive/Dydaktyka/Info_3/kat
aaa


In [18]: cd ../
         cp kat/aaa ./

In [19]: ls

 aaa                       kat     listy.odt  'Zasady przedmiotu.odt'
'Info 3 - Wykład 1.tex~'   lista   Wykład     'Zasady przedmiotu.pdf'


In [20]: rm aaa
         ls

'Info 3 - Wykład 1.tex~'   listy.odt               'Zasady przedmiotu.pdf'
 kat                       Wykład
 lista                    'Zasady przedmiotu.odt'


In [21]: mv kat/aaa ./
         ls

 aaa                       kat     listy.odt  'Zasady przedmiotu.odt'
'Info 3 - Wykład 1.tex~'   lista   Wykład     'Zasady przedmiotu.pdf'


In [23]: ls kat

In [25]: rmdir kat

In [27]: rm aaa

In [3]: ls

 burza
 fig1.png
 fig1.svg
 fig2.png
'Info 3 - Wykład 1.ipynb'
 quote-c-is-a-horrible-language-it-s-made-more-horrible-by-the-fact-that-a-lot-of-substandard-li
 unix.png
 Zaznaczenie_004.png
 Zaznaczenie_005.png
```

7. Wyświetlanie zawartości pliku:

`cat` - Łączenie plików i wyświetlanie zawartości na wyjście standardowe.
`more` or `less` - wyświetla zawartość pliku w bardziej przyjaznej formie - pozwalają na aktywne przeszukiwanie zawartości.
`tail` i `head` - wyświetlają odpowiednio początek i koniec pliku.

```
In [8]: cat burza
```

```
Burza
OSOBY:
ALONSO, król neapolitański.
SEBASTIAN, brat jego.
PROSPERO, prawy książę Mediolanu.
ANTONIO, brat jego, przywłaszczyciel księstwa.
FERDYNAND, syn króla neapolitańskiego.
GONZALO, poczciwy stary radca króla Neapolu.
ADRIAN, FRANCISKO, panowie.
KALIBAN, dziki i potworny niewolnik.
TRYNKULO, trefniś.
STEFANO, piwniczy, pijak.
KAPITAN
BOSMAN
MAJTKOWIE


AKT PIERWSZY
SCENA I

(oraz pozostałe 3000 linii)
```

```
In [6]: more burza_short
```

```
Burza
OSOBY:
ALONSO, król neapolitański.
SEBASTIAN, brat jego.
PROSPERO, prawy książę Mediolanu.
ANTONIO, brat jego, przywłaszczyciel księstwa.
FERDYNAND, syn króla neapolitańskiego.
GONZALO, poczciwy stary radca króla Neapolu.
ADRIAN, FRANCISKO, panowie.
KALIBAN, dziki i potworny niewolnik.
```

```
In [9]: # 10 pierwszych linii pliku brza
        head -n 10 burza
```

```
Burza
OSOBY:
ALONSO, król neapolitański.
```

```
SEBASTIAN, brat jego.
PROSPERO, prawy książę Mediolanu.
ANTONIO, brat jego, przywłaszczyciel księstwa.
FERDYNAND, syn króla neapolitańskiego.
GONZALO, poczciwy stary radca króla Neapolu.
ADRIAN, FRANCISKO, panowie.
KALIBAN, dziki i potworny niewolnik.
```

```
In [11]: #wyświetla koniec pliku
         tail burza
```

```
durzyć - odurzać. [przypis edytorski]

[38]
cobądźkolwiek - dziś: co bądź a. cokolwiek. [przypis edytorski]

[39]
coraggio (wł.) - odwaga; coraggio!: odwagi. [przypis edytorski]

[40]
pobieżyć (daw.) - podążyć. [przypis edytorski]
```

4.5.1 Dzikie karty

Pozwalają na użycie wzorca w celu zastosowania polecenia do więcej niż jednego podmiotu.

- '?' - zastępuje pojedynczy symbol
- '*' - dowolny ciąg znaków
- '[' ']' - odpowiada nazwie zawierającej co najmniej jeden ze znaków
- '{' '}' - iloczyn kartezjański zbiorów

```
In [14]: ls
```

```
 burza
 burza_short
 fig1.png
 fig1.svg
 fig2.png
'Info 3 - Wykład 1.ipynb'
 quote-c-is-a-horrible-language-it-s-made-more-horrible-by-the-fact-that-a-lot-of-substandard-li
 unix.png
 Zaznaczenie_004.png
 Zaznaczenie_005.png
```

Np. poniższe polecenie zostanie zastosowane do wszystkich plików zaczynających się od fig, po którym jest dowolny znak, kropka i dowolny ciąg znaków.

```
In [15]: ls -l fig?.*

-rw-r--r-- 1 sgepner sgepner 13182 mar  3 16:24 fig1.png
-rw-rw-r-- 1 sgepner sgepner 17000 mar  3 16:23 fig1.svg
-rw-rw-r-- 1 sgepner sgepner 16358 mar  3 16:24 fig2.png
```

Albo w ten sposób:

```
In [63]: ls -l fig{1,2}*.[ps]*

-rw-r--r-- 1 sgepner sgepner 13182 mar  3 16:24 fig1.png
-rw-rw-r-- 1 sgepner sgepner 17000 mar  3 16:23 fig1.svg
-rw-rw-r-- 1 sgepner sgepner 16358 mar  3 16:24 fig2.png
```

4.5.2 Polecenia można ze sobą łączyć

Przydatne może być wykorzystanie rezultatu jednego polecenia jako celu dla innego. Służy do tego ukośny apostrof `` <- pod *Esc* nad *Tab*

```
In [65]: cd `pwd`

In [68]: echo Nazwa tego hosta to `hostname`

Nazwa tego hosta to dodo
```