Linux
Wstęp

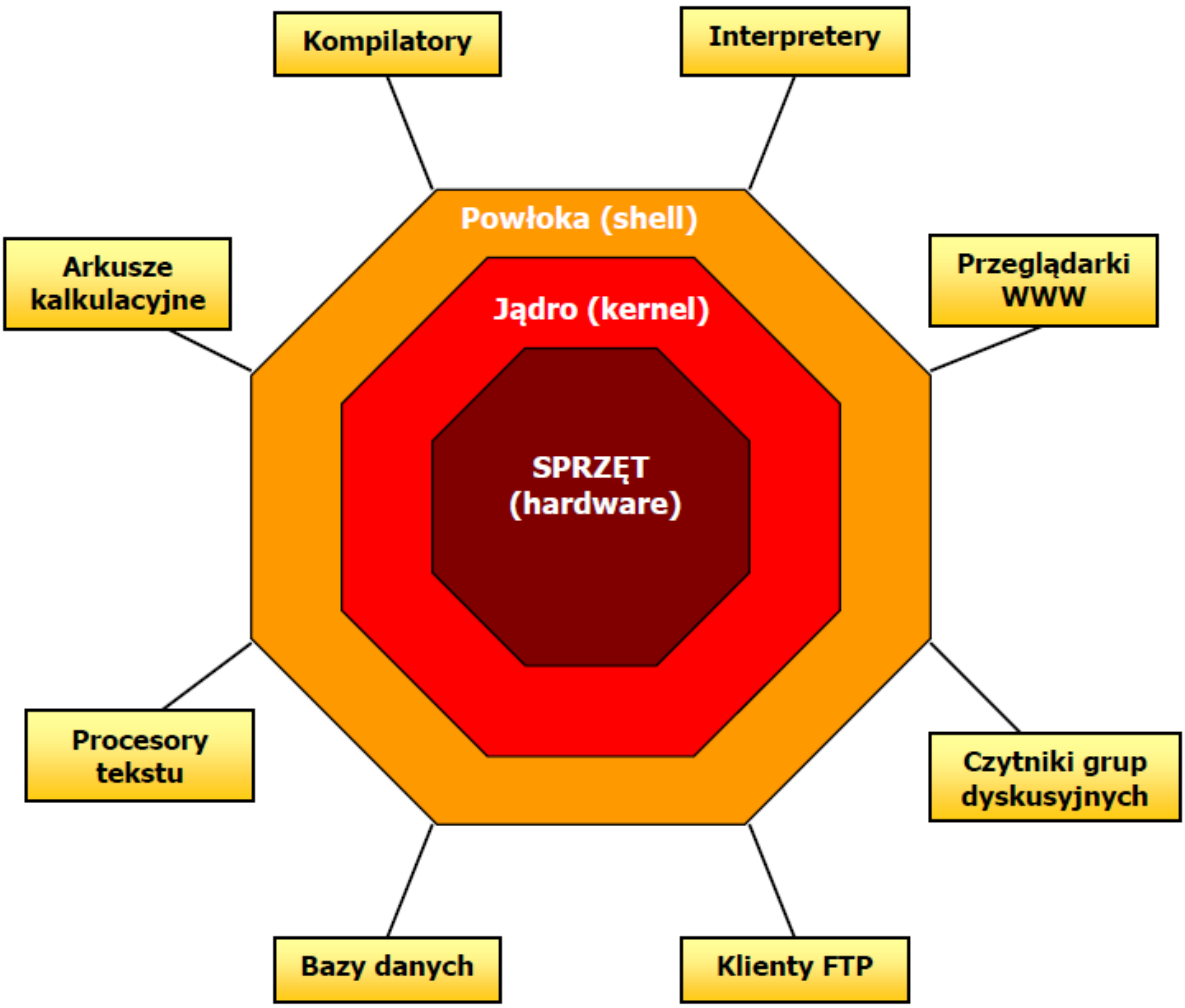
Opracował: Andrzej Nowak

System operacyjny (ang. **OS** - **O**perating **S**ystem) - jest to zbiór programów pośredniczących pomiędzy aplikacjami użytkownika a sprzętem. Każdy system operacyjny oferuje zestaw usług umożliwiających:

- sterowanie wykonaniem procesów poprzez umożliwienie ich tworzenia, kończenia, zatrzymywania i komunikowania się pomiędzy nimi.
- sprawiedliwe szeregowanie procesów ubiegających się o czas procesora. Procesy korzystają z CPU na zasadzie podziału czasu - CPU wykonuje proces, po upływie kwantu czasu jądro zatrzymuje go i wybiera do wykonania inny, później wznawia wykonanie zawieszonego procesu.
- przydzielenie wykonywanemu procesowi pamięci operacyjnej. Jądro pod pewnymi warunkami umożliwia procesom współdzielenie części ich przestrzeni adresowej, lecz chroni prywatną cześć przestrzeni adresowej procesu przed niepowołanym dostępem z zewnątrz. Kiedy systemowi operacyjnemu zaczyna brakować wolnej pamięci RAM, jądro zwalnia pamięć przepisując proces czasowo do pamięci pomocniczej, zwanej pamięcią wymiany (ang. swap).
- Przydzielanie pamięci pomocniczej na efektywne przechowywanie i odczytywanie danych użytkowych. Ta usługa obejmuje system plików. Jądro przydziela pamięć pomocniczą na pliki użytkowe, odzyskuje nieużywaną pamięć, nadaje systemowi plików czytelną strukturę i chroni pliki użytkowe przed niepowołanym dostępem.
- Umożliwianie procesom kontrolowanego dostępu do urządzeń peryferyjnych, takich jak terminale, stacje taśm, stacje dysków i urządzenia sieciowe.

Wprowadzenie do interpretatorów poleceń (ang. command shell) w systemach typu UNIX.

- model warstwowy,
- jądro systemu (ang. kernel),
- ładowany do pamięci podczas startu systemu,
- działa aż do wyjścia z systemu,
- tworzy i kontroluje procesy, zarządza pamięcią, systemem plików, komunikacją między procesami, itp.
- wszystkie pozostałe programy w tym programy powłoki rezydują na dysku.



Shell (interpretator poleceń) jest programem, który jest wywoływany podczas startu systemu operacyjnego (konkretnie po zalogowaniu się), udostępnia interfejs do wydawania poleceń, interpretuje polecenia wydawane na bieżąco (klawiatura), lub zawarte w pliku tzw. skrypcie (programie zawierającym zbiór poleceń).

Trzy główne shell-e w UNIX-ie:

1. Bourne Shell (AT&T), 1979

- oparty na ALGOL-u,
- szybki i kompaktowy,
- standardowy shell używany do administrowania UNIX-a.

2. C Shell (Barkley, koniec lat 70-tych)

- oparty na języku C,
- dodano nowe własności: historia poleceń, aliasy, wbudowana arytmetyka, itp.
- faworyzowany przez użytkowników, lecz powolny

3. Korn Shell (1988)

- jego podzbiorem jest Bourne shell, który został znacznie rozbudowany,

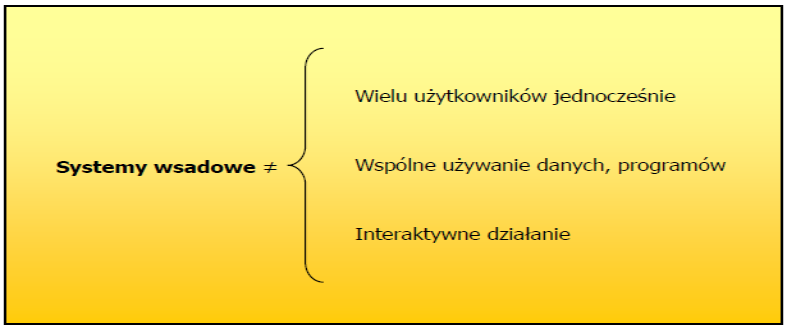
- oba prawie całkowicie kompatybilne,
- graficzny interfejs (nakładka) użytkownika: X-Windows, OSF/Motif, OPEN LOOK.

Powłoki te identyfikowane są przez nazwy:

sh – powłoka Bourne'a
rsh – ograniczona powłoka Bourne'a
ksh – powłoka Korna
rksh – ograniczona powłoka Korna
bash – ponowiona powłoka Bourne'a
csh – powłoka C
tcsh – powłoka ta stanowi rozszerzenie powłoki C.
zsh – jest to najnowsza powłoka, kompatybilna z powłoką Bourne'a

Historia UNIX-a / Shell-a / Linuksa

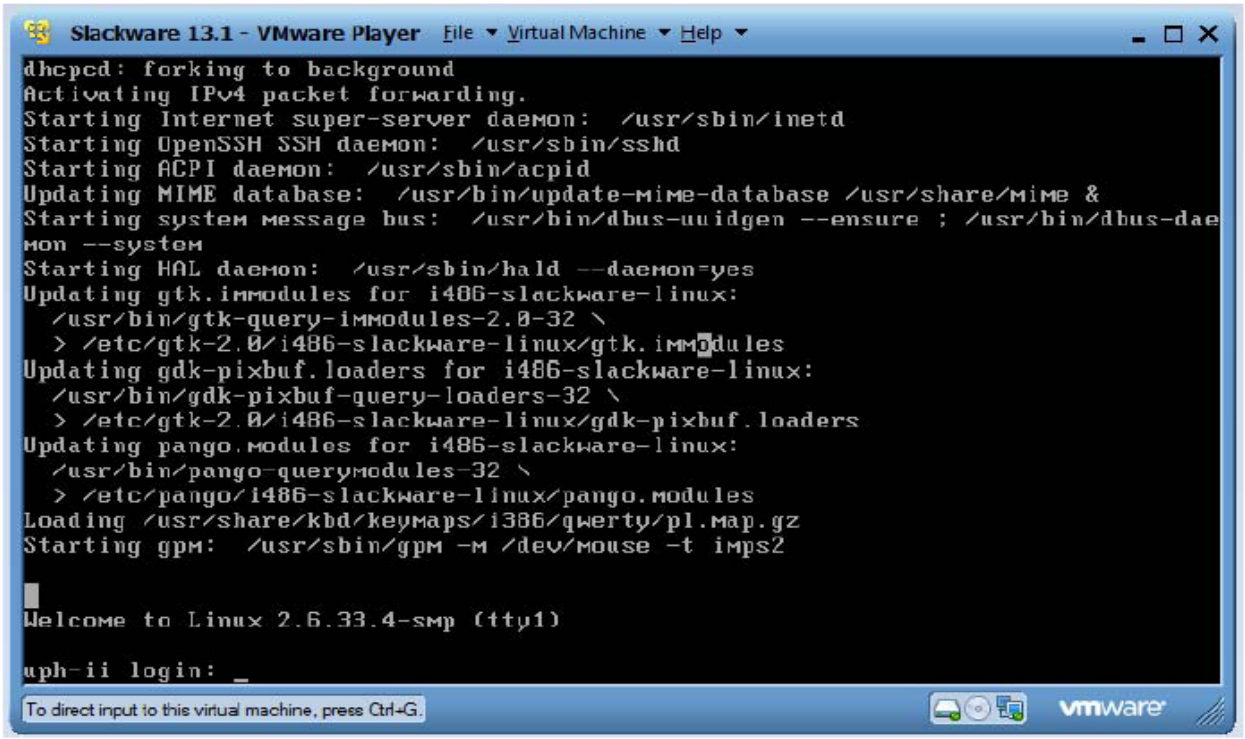
- Motywacja powstania



- Pierwsze wersje UNIX-a: AT&T Bell Labs, 1969 - 1975, Denis Ritchii, Ken Thompson.
- Po pojawieniu się w 1973 kompilatora C 95% kodu przepisano w tym języku (**portabilność** = niezależność od typu komputera).
- AT&T przekazuje źródła UNIX-a uniwersytetom.
- Uniwersytet Kalifornijski w Berkley opracowuje własny wariant BSD (Berkley Software Distribution).
- W 1983 AT&T ogłosiło sprzedaż ulepszonego SYSTEMU V.
- SUN Microsystems, Microsoft i DEC opracowują własne wersje UNIX-a: SunOS, Xenix, Ultrix.
- IEEE opracował standard uniksopodobnych systemów operacyjnych POSIX.
- Ambitny fiński student Linus Torvalds opracował na początku lat 80-tych pierwszą wersję zgodnego ze standardem UNIX-a systemu LINUX dla PC-tów, dostępnego bezpłatnie.

Podstawowe polecenia powłoki:
ls, pwd, date, echo, operator >, cat, more, who, w, finger, man, info, passwd, vi, history, !!.

- Uruchom maszynę wirtualną VMware Player i wczytaj obraz systemu Slackware 13.1.
- Po uruchomieniu powinien pojawić się ekran podobny do poniższego:



- Zaloguj się (login: **root**, hasło: **root**).
- Zainicjuj sesję X-Windows poprzez wydanie polecenia:

```
$ startx
```

- Uruchom terminal.

Przydatne skróty w powłoce:

- **SHIFT + PageUp** - przewijanie ekranu w górę
- **SHIFT + PageDown** - przewijanie ekranu oraz w dół
- **CTRL + a** - przenosi kursor na początek wiersza (e przenosi na koniec)
- **CTRL + d** - kasowanie (uwaga: ten skrót także wylogowuje)
- **CTRL + u** - kasowanie całego wiersza poleceń
- **CTRL + k** - "wycięcie" tekstu do schowka
- **CTRL + y** - "wklejenie" tekstu ze schowka
- **CTRL + l** - jak polecenie clear
- **CTRL + r** - szukamy w historii poleceń polecenia na r
- **strzałki kierunkowe dół, góra** - historia poleceń
- **polecenie + TAB** - dopisze treść polecenia.

Pierwsze polecenia:

```
$ ls
$ pwd
```

katalog domowy / katalog bieżący

```
$ date
$ echo date
$ echo „Jaki dzis dzien”
$ ls
$ echo ls
$ ls > moj_kat
$ cat moj_kat
$ more moj_kat
```

Informacja o poleceniach:

```
$ man ls
$ man man
$ man -k disk
$ man 1 intro
```

Inne sposoby uzyskiwania pomocy:

```
$ ls --help
$ info ls
```

Czy jestem sam w systemie?

```
$ who
$ man who
$ whoami
$ w
$ finger root
$ finger student
```

Zmiana hasła użytkownika:

```
$ man passwd
$ passwd
$ cat /etc/passwd
```

Historia poleceń wydanych przez użytkownika:

```
$ !!
$ history
$ !10
$ !-5
```

Tworzenie i edycja plików:

```
$ vi plik_1
```

- nacisnąć klawisz Insert
- wprowadzić do pliku następujący tekst:

```
To jest nasze pierwsze laboratorium z systemów operacyjnych.
Dzisiaj jest [wstaw nazwę dnia], [wstaw dzień i miesiąc].
```

- nacisnąć **Esc**
- nacisnąć kombinację klawiszy: **Shift + :**
- gdy w lewym dolnym rogu okienka pojawi się znak **:** wtedy kolejno nacisnąć klawisze **wq**

```
$ ls
$ cat plik_1
```

Polecenia edytora vi

1. Wywołanie edytora

$ vi nazwa_pliku

2. Zakończenie sesji z edytorem vi

escape - przełączanie do trybu poleceń
:q - wyjście z edytora, jeśli plik nie został zmodyfikowany
:q! - porzucenie modyfikacji i wyjście bez zapamiętania
:w - zapisanie pliku na dysku
:w nazwa_pliku - zapisanie do pliku o podanej nazwie
:wq - zapisanie pliku na dysk i wyjście z edytora

3. Inne polecenia „dwukropkowe”

:e nazwa_pliku - przerwanie edycji bieżącego pliku i rozpoczęcie edycji innego
:n - przerwanie edycji bieżącego pliku i rozpoczęcie edycji następnego, którego nazwa została podana w linii poleceń
:r nazwa_pliku - wstawienie zawartości podanego pliku do tego, który jest aktualnie edytowany
:! polecenie - uruchomienie określonego polecenia systemowego
!! polecenie - bieżący wiersz zostaje zastąpiony przez wynik działania polecenia
!} polecenie - bieżący akapit zostaje zastąpiony przez tekst będący wynikiem działania określonego polecenia
:s/stary/nowy - zastąpienie pierwszego wystąpienia słowa stary w bieżącej linii słowem nowy
:s/stary/nowy/g - zastąpienie wszystkich wystąpień słowa stary w bieżącej linii słowem nowy
:set nonumber - wyłączenie numeracji wierszy
:set number - włączenie numeracji wierszy

4. Ustawienie kursora na ekranie

H - przesunięcie kursora do górnego wiersza ekranu
M - przesunięcie kursora do środkowego wiersza ekranu
L - przesunięcie kursora do dolnego wiersza ekranu
G - przesunięcie kursora do ostatniego wiersza pliku
nG - przesunięcie kursora do wiersza o n numerze
Ctrl + f - przesunięcie kursora o stronę do przodu
Ctrl + d - przesunięcie kursora o połowę strony do przodu
Ctrl + u - przesunięcie kursora o połowę strony do tyłu

5. Lokalne ruchy kursorem

h - przesunięcie kursora o jeden znak w lewo
l - przesunięcie kursora o jeden znak w prawo
j - przesunięcie kursora o jeden wiersz w dół
k - przesunięcie kursora o jeden wiersz do góry
w - przesunięcie kursora o jedno słowo do przodu
b - przesunięcie kursora o jedno słowo do tyłu
e - przesunięcie kursora do końca aktualnego słowa

6. Szukanie wzorca (w trybie wprowadzania poleceń)

/wzorzec - szukanie wzorca w przód
?wzorzec - szukanie wzorca w tył

Wzorzec może obejmować następujące symbole specjalne:

. - dowolny znak
[…] - każdy ze znaków zawartych w klamrach
[^…] - żaden ze znaków zawartych w klamrach
[e-r] - każdy ze znaków z zakresu od e do r

7. Polecenie przejścia w tryb wprowadzania tekstu

a - przełączenie do trybu wstawiania tekstu z prawej strony kursora
A - przełączenie do trybu wstawiania tekstu za końcem wiersza
i - przełączenie do trybu wstawiania tekstu z lewej strony kursora
I - przełączenie do trybu wstawiania tekstu na początku wiersza
o - wstawienie nowego, pustego wiersza poniżej bieżącego
O - wstawienie nowego, pustego wiersza powyżej bieżącego

8. Usuwanie tekst

D - skasowanie tekstu od pozycji kursora do końca wiersza
x - usunięcie znaku za kursorem
X - usunięcie znaku przed kursorem
dw - kasowanie słowa
dd - kasowanie wiersza
ndd - usunięcie n wierszy

9. Cofanie zmian

u - cofnięcie ostatniej wykonanej operacji

Kończenie pracy w systemie Linux.

- Wyloguj się z powłoki poprzez:

```
$ logout
```

- Zakończ sesję X-Windows.
- Możemy zamknąć bezpiecznie system operacyjny poprzez wydanie jednego z poleceń:

```
$ shutdown –r now
```

Parametr –r oznacza, że komputer zostanie uruchomiony ponownie.

```
$ halt
$ reboot
```

System plików

Główny system plików jest specyficzny dla każdej z maszyn (generalnie znajduje się na poszczególnych maszynach, choć może być montowany z sieci, itd.), zawiera pliki niezbędne do uruchomienia systemu i zamontowania innych systemów plików. Zawartość powinna być wystarczająca do uruchomienia systemu w trybie jednego użytkownika, naprawy pozostałych systemów plików oraz odtwarzania kopii zapasowych. Z systemem plików związany jest podział dysku na partycje.

Rodzaje partycji:

- **podstawowa** (ang. primary partition) – zazwyczaj pierwsza partycja w systemie plików.
- **rozszerzona** (ang. extended partition) – specjalny typ partycji podstawowej, która umożliwia umieszczenie w niej wielu partycji logicznych.
- **logiczna** (ang. logical partition) – partycja, która rezyduje na partycji rozszerzonej.

Na jednym fizycznym dysku można umieścić maksymalnie do czterech partycji podstawowych lub trzy partycje podstawowe i jedną rozszerzoną. Wiele systemów operacyjnych takich jak DOS, Windows, FreeBSD musi być bootowanych z partycji podstawowej i dlatego też posiadają one jedną partycję podstawową. Informacje o partycji podstawowej są przechowywane w pierwszym sektorze dysku zwanym też często MBR.
Linux wpiera wiele systemów plików. Standardowym systemem plików w latach 90 dla linuksa był **second extended filesystem** (**ext2** lub **ext2fs**). W chwili obecnej używa się systemów wpierających journaling i są to:

- **Third extended filesystem** (**ext3**), jest następcą ext2fs i używanym systemem plików obsługującym tzw. journaling,
- **Fourth Extended File System** (**ext4**) - czwarta wersja rozszerzonego systemu plików, następca ext3. Umożliwia obsługę woluminów do 1024 petabajtów. Dzięki możliwości rezerwacji obszaru przeznaczonego na nowe pliki może zmniejszyć fragmentację danych oraz wydajność operacji odczytu oraz zapisu. Działa od wersji jądra 2.6.28.
- **ReiserFS**, został dodany jako standardowy składnik jądra 2.4.1,
- **Extent Filesystem** (XFS), który oryginalnie został zaprojektowany dla Silicon Graphics IRIX OS,
- **Journaled Filesystem** (JFS), którego IBM zaprojektował dla swoich systemów AIX i OS/2.

Linux obsługuje także inne systemy plików takie jak:

- **File Allocation Table** (FAT) system plików używany przez DOS i Windows,
- **New Technology Filesystem** (NTFS) używany przez Windows NT/200x/XP,
- High-Performance Filesystem (HPFS) używany przez OS/2,
- Unix Filesystem (UFS; znany również jako Fast Filesystem lub FFS) używany przez różne wersje Unix-ów,
- The Hierarchical Filesystem (HFS) używany przez Mac OS,
- ISO-9660 i Joliet Filesystem używany przez napędy CD-ROM,
- The Universal Disk Format (UDF), następca systemu ISO-9660 przeznaczony dla nośników optycznych.

Struktura katalogowa systemu Linux.

- hierarchiczna struktura,
- katalog główny (korzeń – root) to „/”,
- wśród katalogów istnieje katalog domowy (home),
- każdy plik lub katalog określony jest jednoznacznie ścieżką dostępu; znak „/” wykorzystywany jest jako separator katalogów w ścieżce.

Root

Home Przechowuje pliki użytkowników. Każdy z użytkowników posiada wewnątrz swój odrębny katalog. Odseparowanie danych użytkownika ułatwia ich archiwizację; najczęściej nie trzeba archiwizować pozostałych danych, lub można to robić rzadziej (nie są tak często zmieniane).

Boot Przechowuje krytyczne pliki niezbędne do bootowania systemu.

Usr Zawiera wszelkie polecenia, biblioteki, strony podręcznika i inne niezmieniające się dane potrzebne podczas normalnej pracy. Pliki w /usr nie powinny być zależne od architektury komputera, nie powinny być zmieniane podczas pracy systemu. Umożliwia to dzielenie plików poprzez sieć, co może być okazać się bardzo ekonomiczne - oszczędza przestrzeń dyskową (katalog ten może zajmować kilkaset megabajtów) i łatwe do administracji (wystarczy uaktualnić/zainstalować program na jednym dysku) Nawet, jeżeli system plików znajduje się na lokalnym dysku może być montowany wyłącznie do odczytu - zmniejsza to szansę wystąpienia błędów po padzie systemu.

Opt Przechowuje programy i pliki z danymi, które nie są związane z systemem operacyjnym.

Var Przechowuje różne pliki związane z codziennym użytkowaniem systemu operacyjnego. Te pliki często są tymczasowe i usuwane są po jakimś czasie np. obsługa kolejkowania poczty e-mail,
drukarek.

Tmp Przechowuje pliki tymczasowe.

Mnt Zawiera punkty montowania dla urządzeń z wymiennymi nośnikami takie jak stacja dyskietek lub CD/DVD.

Dev Zawiera pliki odwzorowujące dyski, terminale i inne urządzenia, które można montować jako urządzenia SO. Programista może się odwoływać do tych urządzeń jak do zwyczajnych plików.

Lib Zawiera biblioteki run-time różnych języków programowania, min.: C/C++

Bin Przechowuje wykonywalne narzędzia systemowe dostępne dla wszystkich użytkowników systemu, stanowiące jego integralną część.

Etc Zawiera niezbędne pliki konfiguracyjne i administracyjne systemu, katalog może być podzielony na podkatalogi, może też obok plików tekstowych zawierać programy wykonywalne, np. skrypty bootujące.

Przejdź do katalogu „/” i wyświetl jego zawartość:

```
$ cd /
$ ls -l
$ ls -l | more
    znak „|” oznacza potok
$ ls /dev
```

Typowe urządzenia w Linuksie:

/dev/console – konsola systemu
/dev/mouse – mysz szeregowa
/dev/hda – pierwszy dysk IDE
/dev/hda1 – pierwsza partycja pierwszego dysku IDE
/dev/hdb – drugi dysk IDE

/dev/hdb1 – pierwsza partycja drugiego dysku IDE
/dev/sda – pierwszy dysk SCSI
/dev/sdb – drugi dysk SCSI
/dev/lp0 – pierwszy port drukarki LPT1
/dev/null – urządzenie puste
/dev/random – urządzenie generujące liczby pseudolosowe
/dev/ttyN – konsola wirtualna
/dev/ptyN – pseudoterminal do logowania się przez sieć
/dev/ttySN – port szeregowy

```
$ ls /var
$ ls /bin
$ ls -p /usr /bin/ls
```

Parametr –p dodaje wskaźnik na końcu każdej nazwy określając typ każdego z plików. Poniżej są opisane te wskaźniki:
/ – katalog
@ – dowiązanie symboliczne
= – gniazdo (ang. socket)
| – potok (ang. pipe)

Informacje o ilości zajętego/wolnego miejsca:

```
$ du -sh *
$ df /dev/hda
$ free
```

Montowanie urządzeń

```
$ mount
$ fdisk -l
```

Montowanie dyskietki

```
$ mount /dev/floppy /mnt/floppy
$ mount /dev/fd0 /mnt/fd0
```

Montowanie napędu CD

```
$ mount /dev/cdrom /mnt/cdrom
$ mount -t iso9660 /dev/cdrom /mnt/cdrom
```

Montowanie partycji z Windows

```
$ mount -t vfat /dev/hdc1 /mnt/dyskc
```

Odmontowywanie urządzeń

```
$ umount /mnt/cdrom
```

Listowanie katalogu

Uwaga do polecenia ls. Polecenie to listuje zawartość bieżącego katalogu. Lista zawartości katalogu jest kolorowana (ls jest aliasem do tego samego polecenia, ale z parametrem kolorowania), a poszczególne kolory oznaczają (nie zawsze):

- **ciemnoniebieski** katalogi,
- **szary** zwykłe pliki,
- **zielony** pliki wykonywalne,
- **magenta (karmazynowy)** pliki graficzne,
- **czerwony** skompresowane,
- **jasnoniebieski** dowiązania symboliczne,
- **żółty** pliki urządzeń,

- **brązowy** FIFO (potoki).

```
$ alias ls
$ ls
$ date
$ date > data
$ ls
$ cat data
```

Tworzenie katalogów

```
$ pwd
$ mkdir lab2
$ ls
$ ls -d
$ cd lab2
$ pwd
$ ls -a
$ vi week
```

Utwórz plik **week** o następującej strukturze:

```
poniedziałek
wtorek
środa
czwartek
piątek
sobota
niedziela
```

```
$ ls -l
$ mkdir lab2_1
$ mkdir lab2_2
$ ls
```

Kopiowanie plików

```
$ cp week week1
$ cat week1
$ cp week week2
$ ls
$ cp -i week1 week2
```

Polecenie z parametrem –i pyta czy ma nadpisać istniejący plik; jest również użyteczny przy poleceniu mv

Zmiana nazwy pliku

```
$ mv week2 wee
$ ls
```

Usuwanie plików

```
$ rm wee
$ ls
$ rm *
```

BŁĄD?D?

```
$ date; ls > info
$ ls -l
```

```
$ cat info
$ (date; ls) > info1
```

Przechodzenie do podkatalogów

```
$ pwd
$ cd lab2_1
$ pwd
$ ls -a
$ cd ..
$ ls; pwd
$ cd lab2_1
$ pwd
$ cd
$ pwd; ls
```

Usunięcie katalogów (pustych)

```
$ cd lab2
$ ls
$ rmdir lab2_2
$ ls
```

Usuwanie katalogów niepustych

```
$ ls
$ cd ..
$ ls
$ rmdir lab2
$ rm -r lab2
```

Kilka poleceń można wykonać z pojedynczej linii poleceń

```
$ date; ls
$ date;\
ls
$ cd lab3
```

Polecenie `cat` można wykorzystać do połączenia kilku plików w jeden plik

```
$ cat > p1
    Co by tutaj napisać.
```

Naciskamy Ctrl + C

```
$ ls
$ cat p1
```

Utwórzmy kilka plików:

```
$ date > data
$ w > kto
$ ls > lista
$ ls
```

Łączenie plików

```
$ cat p1 data kto > razem
$ cat razem
```

Dołączanie do pliku

```
$ cat data lista >> razem
$ cat razem
```

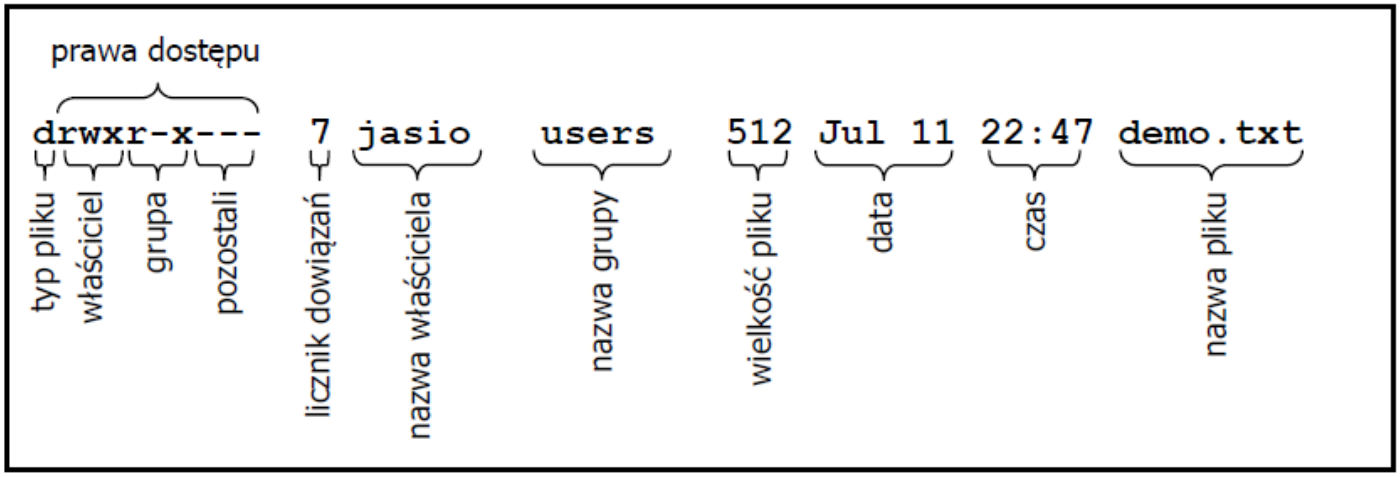
Ale!

```
$ cat data lista > razem
$ cat razem
```

Jeszcze o touch

```
$ ls -l
$ touch pusty
```

Prawa dostępu do plików/katalogów



Prawa dostępu wyświetlane przy pomocy polecenia ls

- typ pliku:
  - **-** plik zwykły
  - **d** katalog
  - **b** plik specjalny (blokowy)
  - **c** plik specjalny (znakowy)
  - **`** gniazdo
  - **p** łącze nazwane (FIFO)
  - **l** dowiązanie miękkie (symboliczne)
- prawa dostępu:
  - **r** (read) – czytanie
  - **w** (write) - pisanie
  - **x** (execute) – wykonywanie
  - **-** brak prawa (czytania, pisania, wykonywania)

Uwaga: prawa dostępu do podkatalogów

- prawo czytania wystarcza jedynie do obejrzenia zawartości podkatalogu
- aby do niego wejść, użytkownik musi mieć dodatkowo prawo wykonywania

Prawa dostępu nie są nadane raz na zawsze. Możemy je zmieniać. Zmiany praw dostępu do pliku (poleceniem **chmod**) może dokonać jedynie jego właściciel bądź administrator (root). Inną metodą wpływu na dostępność pliku jest zmiana jego właściciela bądź grupy (polecenia **chown** i **chgrp**). Każdy pakiet praw rwx ma swój odpowiednik w postaci zapisu ósemkowego:

```
rwx   rwx   rwx
421   421   421
```

Żeby ustawić jakieś prawa dostępu należy dodać do siebie te liczby, które stoją przy prawach, które chcemy ustawić, ale dla każdej trójki osobno. Trochę przykładów dla lepszego zrozumienia:

```
- r-x --- --- nazwa_pliku
  4 1
```

To będzie 4+0+1, 0+0+0, 0+0+0 czyli 500. `chmod 500 nazwa_pliku` zmieni prawa dostępu na odczyt i wykonywanie dla właściciela i zabierze wszystkie prawa dla grupy i innych użytkowników.

```
- rwx r-x r-x nazwa_pliku
  421 4 1 4 1
```

To będzie 4+2+1, 4+0+1, 4+0+1 czyli 755.

Teraz łatwiejszy sposób na zmianę praw dostępu. Polecenie chmod może też przyjmować takie parametry:

```
chmod ugoa+-=rwx nazwa_pliku
```

I tak: u - user (właściciel pliku), g - group (grupa do jakiej plik należy), o - others (pozostała część użytkowników). + ustawia podane prawa, - zabiera. = czyni podane prawa jakie podaliśmy po znaku „=”.

Przykłady:

chmod `u+x nazwa_pliku` - nadaje właścicielowi pliku prawo do wykonywania.
chmod `ug+rx nazwa_pliku` - nadaje właścicielowi i grupie, do której należy plik prawa do odczytu i wykonywania

Dowiązania miękkie i twarde

```
$ date > plik_1
$ ls > plik_2
$ ls
$ ln plik_1 plik_dt
$ ls -p
$ ln -s plik_2 plik_dm
```

parametr –s tworzy dowiązanie miękkie

```
$ ls -p
$ ls -l
$ cat plik_dt
$ cat plik_dm
$ rm plik_2
$ ls
$ cat plik_dm
```

BŁĄD? `cat: plik_dm: Nie ma takiego pliku ani katalogu`

```
$ rm plik_1
$ ls
$ cat plik_dt
```