Polecenie man – jak *manual*

Wyświetla dokumentację systemu.

Użycie:

man polecenie

np.

man ls

- wyświetli informacje o poleceniu ls

Enter – kolejna linia tekstu
Spacja – kolejna strona tekstu
q – wyjście
W systemach GNU/Linux często zaimplementowane są również kursory.

Użyteczne parametry:
man -s x polecenie
- Wyświetla pomoc z sekcji o numerze x.

Polecenie ls – jak *list*

Wyświetla zawartość bieżącego katalogu.

Polecenie ls sciezka/do/katalogu wypisze na terminal zawartość katalogu podanego jako parametr.

Użyteczne parametry:
ls -l – Wyświetla listę zawierającą dodatkowe informacje
ls -a – Pokazuje również obiekty ukryte (których nazwa rozpoczyna się od znaku .)

Polecenie cp – jak *copy*
Kopiuje plik z pierwszego parametru do miejsca wskazanego przez parametr ostatni.
Przykład:

cp dokument.txt dokument.bak

wykona kopię pliku dokument.txt do pliku dokument.bak.

Użyteczne parametry:
cp -r katalog1/ katalog2/ – kopiuje rekursywnie zawartość katalogu. W przykładzie tworzymy kopię katalogu katalog1 jako katalog2.

Polecenie mv – jak *move*
jak cp, lecz dokonuje przeniesienia obiektu lub zmiany nazwy.

Polecenie mkdir – jak *make directory*
Tworzy nowy katalog o podanej w parametrze nazwie.
Przykład:

mkdir zadanie_1

Użyteczne parametry:
mkdir -p sciezka/do/katalogu
Tworzy wszystkie nieistniejące katalogi „po drodze”. W tym przypadku, jeżeli nie istnieje żaden katalog, tworzy katalog **sciezka/**, w nim katalog **do/**, w nim zaś **katalogu/**.

Polecenie rm – jak *remove*
Usuwa elementy systemu plików. Używać ostrożnie – usuwanie jest **nieodwracalne**.
Użycie:

```
rm plik.txt
```

kasuje plik.txt z bieżącego katalogu.'

Użyteczne parametry
rm -r katalog/ - usuwa rekurencyjnie zawartość katalogu i sam katalog.
-f – nie pyta użytkownika przed usunięciem.

W naszym serwerze użytkownik nie może wyłączyć pytania o usunięcie plików.

Polecenie rmdir – jak *remove directory*
Usuwa katalog pod warunkiem, że jest on pusty.
Przykład:

```
rmdir katalog/
```

Wybieranie wielu obiektów:
W systemie UNIX używane są przynajmniej 3 typy wieloznaczników służących do specyfikacji wielu plików dla poleceń:

- * - zastępuje 0 lub więcej dowolnych znaków.
Np.

```
cp plik*.txt kopia/
```

skopiuje z bieżącego katalogu do podfolderu pliki (o ile takie istnieją) plik.txt, plik10.txt, plik_aaa.txt, ale nie skopiuje już pliku plik.bak.
Tym znakiem nie można zastąpić kropki na początku pliku (oznaczenie pliku ukrytego).

- ? - zastępuje jeden znak.
Np.

```
cp plik?0.txt kopia/
```

skopiuje z bieżącego katalogu do podfolderu pliki (o ile takie istnieją) plik10.txt, plik20.txt, ale nie skopiuje już plików plik1.txt, plik12.txt czy plik0.txt

- [] - definicja zestawu znaków:
Np.

```
ls plik[1-6].txt
```

Pokaże pliki plik1.txt, plik2.txt, plik3.txt, plik4.txt, plik5.txt, plik6.txt.
Można używać również zakresów: a-z, A-Z, 0-9. Przykładowo:

```
ls plik[0-9][a-z].txt
```

wylistuje pliki o nazwie plik, następnie dowolna cyfra, później dowolna małą litera .txt, np. plik1c.txt.

Należy zauważyć, że **nie jest to sekwencja**, tylko zestaw dopuszczalnych znaków. Tak więc wybór plik[10-15].txt nie jest prawidłowy i nie wybierze z przedziału 10-15. Natomiast aby uzyskać taki efekt, możemy użyć wyboru: plik1[0-5].txt.

W większości dystrybucji można zanegować zestaw umieszczając na początku wykrzyknik (np. plik[!0-5].txt).

Uwaga: rekurencyjne usuwanie (kopiowanie, przenoszenie, zmiana uprawnień itd.) plików ukrytych korzystając z wieloznacznika .* nie jest prawidłowe, a skutki (szczególnie na uprawnieniach administratora) są często katastrofalne. Korzystając z wieloznacznika .* zostanie uwzględniony bowiem katalog wyżej – o nazwie .. - więc polecenie "przetworzy" również elementy w katalogu wyżej. Należy używać wieloznacznika .[^.]* - nazwa zaczyna się od . Ale drugim znakiem nie może być kropka.

Polecenie cat – jak *concatenate*
Wypisuje na terminal lub do pliku zawartość podanych w parametrach plików.
Użycie:

```
cat plik.txt
```

Wypisze zawartość pliku na konsolę.

```
cat plik1.txt plik2.txt
```

Wypisze na konsolę zawartości obu plików jeden po drugim.

Przekierowanie
Wyjście dowolnego polecenia można zamiast do konsoli przekierować do pliku. W tym calu korzystamy z operatorów > i >>.
- Operator > zapisuje wyjście polecenia do nowego pliku. Gdy plik istnieje, zostanie nadpisany lub zostanie wyświetlony błąd.
- Operator >> dopisuje wyjście polecenia na koniec istniejącego pliku.

Np.

```
ls -l >lista.txt
```

Utworzy w bieżącym katalogu plik tekstowy zawierający listę obiektów w bieżącym katalogu, jeszcze bez tego pliku.
A później:

```
ls -l ./praca/ >>lista.txt
```

dopisze na koniec tego pliku listę obiektów w podkatalogu praca.

Zadania:
Utwórz pliki do ćwiczeń pobierając i uruchamiając skrypt:

```
wget http://149.156.111.10/~mwilkus/cw1-gen.sh
chmod u+x cw1-gen.sh
./cw1-gen.sh
```

W katalogu zadanie_1 zostaną utworzone pliki.
1. Skasuj wszystkie pliki .tmp
2. Utwórz katalog pierwszy/ i przenieś tam pliki plik1.txt … plik8.txt
3. Połącz pliki plik20.txt .. plik23.txt do osobnego pliku.