Ćwiczenie 1 → Moc zaklęta w terminalu[1]

0. Połączenie z linuxem

- Pobierz klucz .pem z adres http://enwo.pl/cm/uam.pem i zapisz w znanej lokalizacji
- Uruchom terminal (np. Cygwin , Putty, cmd.exe, Powershell, git-bash) I przejdź do lokalizacji z pobranym plikiem .pem
- Połącz się z serwerem:

```
ssh -i uam.pem userNR@3.67.83.170 (lub inne IP)
```

gdzie NR = numer twojego stanowiska. Hasło: Almukantarata12345

1. Poruszanie się po terminalu

'cd' – zmiana katalog (cd = *change directory)*

'ls' – wyświetla zawartość katalogu

'pwd' – wyświetla pełną ścieżkę do obecnego katalogu

- *Wskazówka → Bardzo przydatnym klawiszem podczas używania konsoli jest <Tab>.*
- *<Tab> uzupełnia brakujący tekst polecenia/nazwy pliku/nazwy katalogu. Innym ważnym zastosowaniem tego klawisza jest dopełnienie nazwy polecenia, którego nie pamiętamy.*

a) Wejdź w terminalu do katalogu /home/userXXX/dane/ i wyświetl jego zawartość

b) Przejdź następnie jeden katalog wyżej i wyświetl jego pełną ścieżkę oraz zawartość katalogu

c) Przejdź następnie do katalogu /

d) Sprawdź działanie polecenia cd ~ Do jakiego katalogu zostaliśmy teraz bezpośrednio przeniesieni?

2. Tworzenie, usuwanie, kopiowanie i przenoszenie plików i katalogów

'touch *nazwapliku'* – tworzy pusty plik *nazwapliku*

'rm nazwapliku' – usuwa plik *nazwapliku*

'mkdir *nazwakatalogu*' – tworzy katalog *nazwakatalogu* (mkdir = *make directory)*

'rmdir *nazwakatalogu*' – usuwa katalog *nazwakatalogu;* osobiście wolę nieco inną formę tego polecenia, które w bash'u można zastąpić 'rm -r'

'mv' – przenoszenie lub nowa nazwa pliku

'cp' – kopiowanie pliku

---

1 Materiały opracowano na podstawie: [1]http://www.ubuntu-pomoc.org/moc-zakleta-w-terminalu-cz-7/ [2] http://czytelnia.ubuntu.pl/index.php/2010/02/18/konsola_nie_gryzie_czesc1/ [3] http://linuxcommand.org/lc3_lts0020.php [4] https://www.linux.org.pl/PLUG/susebook/ [5] http://www.i-szkola.cba.pl/index.php/technik-informatyk/soisk/32-strumienie

'cp -r' – kopiowanie katalogu

a) Utwórz w katalogu domowym folder „test”
b) Bez wchodzenia do stworzonego katalogu utwórz w nim 2 pliki o nazwach: „pusty_plik” i „pusty plik”. Do stworzenia obu plików użyj jednego polecenia (ten drugi ma spację w nazwie pliku)
c) Wykorzystując dowolny edytor tekstowy (np. vim lub nano) otwórz „pusty_plik” i wpisz w nim 2 dowolne linie tekstu. Zapisz zmiany wychodząc z terminala.
d) Wyświetl jego zawartość w terminalu z wykorzystaniem polecenia 'cat'.
e) Wyświetl statystyki edytowanego pliku z wykorzystaniem programu 'wc' (word count)
f) Zamień nazwę pliku „pusty plik” na inną dowolną; następnie usuń ten plik
g) Skopiuj pozostały plik do katalogu domowego
h) Skopiuj cały katalog testowy (~/test) do katalogu domowego; Następnie wyświetl zawartość katalogu domowego aby sprawdzić czy wszystko skopiowało się poprawnie
i) Usuń skopiowany plik i katalog z katalogu domowego

3. Zarządzanie procesami *(+wstęp do potokowania)*
'ps' – wyświetla listę procesów
a) *Wyświetl wyniki poleceń:* 'ps -A', 'ps -u', 'ps -aux'
b) Uruchom „python” lub „R” z poziomu terminala. Poeksperymentuj ze składnią. Następnie uruchom jeszcze raz „python” lub „R” i zakończ komendę znakiem „&”. W tym momencie proces jest uruchomiony w tle i możemy dalej pracować w terminalu
c) Jeśli chcemy ponownie przywołać proces w terminalu możemy go ponownie „przywołać na wierzch” komendą fg 1 (1 – oznacza kolejność procesu przeniesionego w tło)
d) W tym momencie możemy zabić proces kombinacją klawiszy ctrl+c
e) Ponownie uruchom przeglądarkę z terminala
f) Wyświetl wszystkie procesy korzystając z komendy ps -A. Nasz wynik w następnym poleceniu nieco sobie rozszerzymy
g) Przefiltrujmy wszystkie procesy, które zawierają słowo „python”. Do tego celu wykorzystamy polecenie „grep”. Sprawdźmy zatem wynik polecenia: ps -A | grep python
h) skillujmy proces (python lub R): kill numerprocesu
i) Killowanie można też wykonać w inny sposób, ale dotyczy to tylko i wyłącznie sytuacji w której mamy otwarty tylko 1 program o tej samej nazwie. Urachamiamy ponownie R lub python. Następnie testujemy polecenie: killall python

4. Potokowanie strumieni

Każdy uruchomiony w Linuksie proces pobiera skądś dane, gdzieś wysyła wyniki swojego działania i komunikaty o błędach. Dane przesyłane są między urządzeniami w postaci strumieni. Standardowe wejście i wyjście (strumienie danych) możemy przekierować. Do tego celu przygotowano trzy operatory:

znak „<" umożliwia przekierowanie zawartości pliku do standardowego wyjścia, np. more < plik,

znak „>" umożliwia przekierowanie strumienia danych ze standardowego wyjścia do pliku; jeżeli plik istnieje, to jego poprzednia zawartość zostaje usunięta, np. ls > plik,

znaki „>>" umożliwiają przekierowanie strumienia danych ze standardowego wyjścia do pliku; jeżeli plik istnieje, to nowe dane zostają dopisane na końcu pliku.

4.1. Przełączanie standardowego wejścia

Jako standardowe wejście można zamiast klawiatury użyć pliku:

< plik

Najlepiej można to zademonstrować na przykładzie praktycznym:

a) utwórz plik lista o zawartości:

Ubuntu
Mint
Debian
Lubuntu

Użyjemy polecenia sort, dla którego standardowym wejściem będzie plik lista:

b) sort < lista

W wyniku zastosowania polecenia zawartość pliku zostanie posortowana.

4.2. Przełączanie standardowego wyjścia

Wynik jakiegoś polecenia można wysłać do pliku, a nie na ekran. Do tego celu używa się operatora:

> plik

a) Przykład: ls -la /usr/bin > ~/test/wynik

Rezultat działania polecenia ls -la /usr/bin trafi do pliku o nazwie wynik. Jeśli wcześniej nie istniał plik o takiej samej nazwie, to zostanie utworzony, jeśli istniał, cała jego poprzednia zawartość zostanie nadpisana.

Jeśli chcemy, aby dane wyjściowe dopisywane były na końcu pliku, bez wymazywania jego wcześniejszej zawartości, stosujemy operator:

>> plik

b) Przykład: free -m >> ~/test/wynik

Wynik polecenia free -m (pokazuje wykorzystanie pamięci RAM i swap) zostanie dopisany na końcu pliku wynik, nie naruszając jego wcześniejszej zawartości.

c) date >> ~/test/wynik

d) Wyświetl zawartość stworzonego pliku

5. Programy do szybkiej obróbki strumieni tekstowych + wykorzystanie strumieni danych + pobieranie i rozpakowywanie plików w terminalu oraz podstawy wyrażeń regularnych

Zastosowanie znaku | pozwala na łączenie wyjścia jednego polecenia z wejściem innego. Dane wygenerowane za pomocą pierwszego polecenia, przekazane zostaną na wejście następnego polecenia i po przetworzeniu przekazane na wejście kolejnego lub na ekran.
Tego typu przetwarzanie danych nazywane jest potokiem. Polecenia często wykorzystywane w potokach:

more - służy do przeglądania tekstu strona po stronie, jeden ekran na raz, przewijanie stron możliwe tylko „do przodu", np. ls - la | more,
less - podobnie jak more, ale przewijanie stron możliwe w obu kierunkach, np. ls ~/Pobrane | less,
cat - polecenie wyświetla na ekranie zawartość pliku tekstowego (już poznaliśmy) /lub wielu plików testowych/
grep - przeszukuje wskazany strumień danych, szukając linii zawierających ciąg znaków pasujących do podanego wzorca.

head - wyświetla pierwsze 10 linii wskazanego pliku. Możemy za pomocą opcji -n zadeklarować także inną liczbę wierszy

tail - wyświetla ostatnie 10 linii wskazanego pliku. Możemy za pomocą opcji -n zadeklarować także inną liczbę wierszy

cut -c x1 x2 - wyświetla znaki zawarte pomiędzy kolumną x1 do x2

paste plik1 plik2 - łączy kolumny zawarte w plikach: plik1 i plik2

Przejdźmy do sprawdzenia jak można wykorzystać część z tych poleceń w praktyce.

a) Pobieramy przygotowane dane zawierające surowe pomiary 1 roku danych radiosondażowych w Łebie (http://weather.uwyo.edu/cgi-bin/sounding?region=europe&TYPE=TEXT%3ALIST&YEAR=2004&MONTH=08&FROM=1412&TO=1412&STNM=12120). Dane można pobrać bezpośrednio z internetu: http://enwo.pl/cm/sounding.tar.gz . Nie trzeba uruchamiać przeglądarki. Można skorzystać z programu wget działającego w terminalu:

wget -c http://enwo.pl/cm/sounding.tar.gz

b) Rozpakowujemy dane:

tar -xzvf sounding.tar.gz -C /home/userXXX/dane/

c) Wchodzimy do katalogu 'sounding' i wyświetlamy jego zawartość. Wyświetlmy także zawartość jednego z plików

Załóżmy, że chcemy uzyskać z wszystkich plików z godziny 12, tylko w miesiącach letnich (VI-VIII) wartość parametru CAPE, który znajduje się zwykle w końcowym fragmencie każdego z plików. Wykorzystamy do tego celu wyrażenia regularne, które w znacznym stopniu usprawniają nam pracę w konsoli:

Symbol	**Zastępuje**
`.`	dowolny znak
`^`	dopasuj występujące po operatorze wyrażenie do początku wiersza
`$`	dopasuj poprzedzające wyrażenie do końca wiersza
`\x`	znaki specjalne, gdzie `x` to znak specjalny np. `\$` zastąpi znak dolara
[lista]	zastępuje dowolny znak spośród tych wymienionych na liście, mogą to być przedziały np. [0-9] lub [a-d]
`?`	dokładnie jeden element
`a|b`	dopasuje wyrażenie `a` lub wyrażenie `b`
*	dopasuj zero lub więcej wyrażeń znaku poprzedzający operator

d) Spróbujmy w pierwszej kolejności zdefiniować listę plików która będzie nam potrzebna. Do wyświetlania plików wykorzystujemy polecenie „ls". A zatem korzystając z powyższej tabelki można spróbować wyniku polecenia:

ls 2004.0[6-8]*godz12

e) Jeśli chcemy sprawdzić ile plików otrzymaliśmy, możemy przepotokować wynik polecenia do programu „wc"

f) Jeśli chcemy przekazać wynik operacji jednego programu jako listę argumentów dla drugiego programu wykorzystujemy do tego celu komendę podaną w `` (zwykle pod klawiszem ESC), np.:

cat `ls 2004.0[6-8]*godz12`

Wyświetla zawartość wszystkich plików będących wynikiem polecenia: ls 2004.0[6-8]*godz12

g) Teraz chcemy pobrać linie, które zawierają słowo CAPE. Do tego celu wykorzystamy komendę grep:

cat `ls 2004.0[6-8]*godz12` | grep CAPE

h) Widzimy, że wartość CAPE jest podana zawsze w tych samych kolumnach. Możemy wyświetlić wartości bez poprzedzającego tekstu na kilka sposobów. Jedną z możliwości jest funkcja cut, która wymaga wiedzy na temat tego, z którego miejsca mają zostać wybrane wartości do wyświetlenia. Zapiszmy sobie zatem wynik wcześniejszej operacji do jakiegoś pliku (przekierowanie strumienia operacji > nazwapliku) i sprawdźmy, które kolumny będą do tego celu najlepsze

i) Ostatecznym wynikiem będzie zatem komenda:

cat `ls 2004.0[6-8]*godz12` | grep CAPE | cut -c X1-X2 > plik.csv

j) *Możemy sobie teraz plik np. posortować* (sort plik.csv)

6. Wskazówki i skróty do stosowania w terminalu:

ctrl* + *shift* + *c → *kopiowanie zaznaczonego fragmentu tekstu*

ctrl* + *shift* + *v → *wklejanie fragmentu tekstu do terminala*

ctrl* + *shift* + *t → *otworzenie nowej zakładki terminala*

ctrl* + *d → *równoznacznik polecenia exit*

ctrl* + *z → *zatrzymanie polecenia wykonywanego w terminalu*