



Politechnika
Wroclawska

Systemy operacyjne 2

Laboratorium nr 5
Przetwarzanie potokowe

Szymon Datko
szymon.datko@pwr.edu.pl

Wydział Informatyki i Telekomunikacji,
Politechnika Wroclawska

semestr letni 2022/2023



W skrócie

- Wykorzystanie łącz nienazwanych (potoków – czyli | takich | kresek).
- Przekierowanie wyjścia z jednego procesu na wejście drugiego procesu.
- Zapoznanie się z możliwościami kilku programów do przetwarzania tekstu.

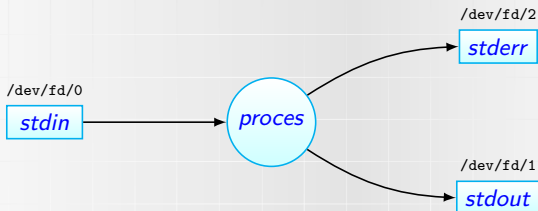


(punkt widzenia: młody administrator odkrywa prawdę o swoim systemie)

Część I

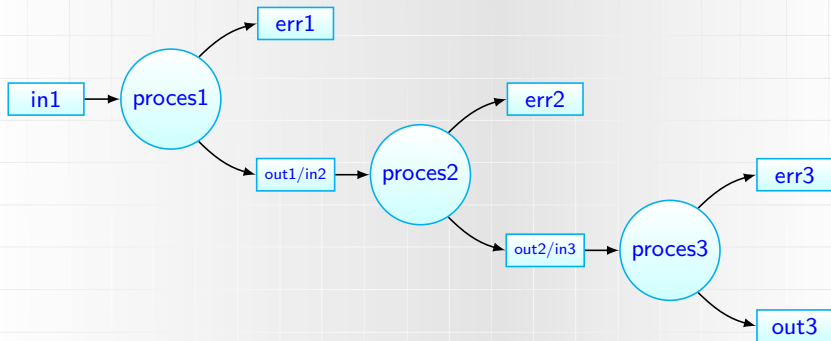
Omówienie zagadnień

Proces w systemie Linux



- Proces stanowi pojedynczą, działającą instancję jakiegoś programu.
- Domyślnie każdy proces ma otwarte przynajmniej 3 podstawowe pliki.
- Otwarcie dodatkowego pliku to pojawienie się nowego bloku na diagramie.
- Operatory `< i >` pozwalają zmienić pliki standardowych wejść/wyjść.
 - ▶ `find /etc`
 - ▶ `find /etc > wynik-stdout 2> wynik-stderr`
 - ▶ `echo 'tekst na standardowy strumień błędów' >& 2`
 - ▶ `read ZMIENNA < plik-z-wejsciem`

Mechanizm łącz nienazwanych / potoków



- Wynik uruchomienia komendy: `program1 | program2 | program3`.
- Wymiana danych odbywa się przez specjalny rodzaj pliku (w RAM).
- Mechanizm powszechnie stosowany do przetwarzania danych tekstowych.
- Szczegóły dla zainteresowanych: `man 7 pipe`.

Po co?

- Mnóstwo danych, z którymi pracujemy w powłoce, to dane tekstowe.
- Mamy też zestaw narzędzi, umożliwiających przetwarzanie tych danych.
- Potoki pozwalają uzyskać wynik bez angażowania plików tymczasowych.
- Jest to często szybsze i bardziej eleganckie podejście.



Część II

Podstawowe komendy

Program sort

- Narzędzie do sortowania zawartości z pliku lub standardowego wejścia.
- Przykłady użycia:

```
▶ ls -lah . | sort --human-numeric-sort --key 5
```

```
▶ sort --reverse /etc/services
```

NAME

sort - sort lines of text files

SYNOPSIS

```
sort [OPTION]... [FILE]...  
sort [OPTION]... --files0-from=F
```

DESCRIPTION

Write sorted concatenation of all FILE(s) to standard output.

With no FILE, or when FILE is -, read standard input.

Mandatory arguments to long options are mandatory for short options too.

Ordering options:

```
-b, --ignore-leading-blanks  
    ignore leading blanks
```

```
-d, --dictionary-order  
    consider only blanks and alphanumeric characters
```


Program uniq

- Narzędzie do usuwania powtórzeń w następujących po sobie liniach.
- Przykłady użycia:

```
▶ find . -printf '%f\n' | sort | uniq
```

```
▶ cut -d ' ' -f 1 /etc/services | sort | uniq -c
```

NAME

uniq - report or omit repeated lines

SYNOPSIS

uniq [OPTION]... [INPUT [OUTPUT]]

DESCRIPTION

Filter adjacent matching lines from INPUT (or standard input), writing to OUTPUT (or standard output).

With no options, matching lines are merged to the first occurrence.

Mandatory arguments to long options are mandatory for short options too.

-c, --count
 prefix lines by the number of occurrences

-d, --repeated
 only print duplicate lines, one for each group

Program grep

- Narzędzie do wyodrębniania linii/fragmentów pasujących do podanego wzoru.
- Przykłady użycia:
 - ▶ `grep "${USER}" /etc/passwd`
 - ▶ `ls -l /dev | grep --invert-match '[0-9]$'`

NAME

`grep`, `egrep`, `fgrep` - print lines that match patterns

SYNOPSIS

```
grep [OPTION...] PATTERNS [FILE...]  
grep [OPTION...] -e PATTERNS ... [FILE...]  
grep [OPTION...] -f PATTERN_FILE ... [FILE...]
```

DESCRIPTION

`grep` searches for `PATTERNS` in each `FILE`. `PATTERNS` is one or more patterns separated by newline characters, and `grep` prints each line that matches a pattern. Typically `PATTERNS` should be quoted when `grep` is used in a shell command.

A `FILE` of "-" stands for standard input. If no `FILE` is given, recursive searches examine the working directory, and nonrecursive searches read standard input.

In addition, the variant programs `egrep` and `fgrep` are the same as `grep -E` and `grep -F`, respectively. These variants are deprecated, but are provided for backward compatibility.

Program rev

- Narzędzie odwracające kolejność znaków w wierszach podanych plików.
- Przykłady użycia:

```
▶ echo 'nazwa.pliku.txt' | rev | cut -d '.' -f 2- | rev
```

NAME

rev - reverse lines characterwise

SYNOPSIS

```
rev [option] [file...]
```

DESCRIPTION

The `rev` utility copies the specified files to standard output, reversing the order of characters in every line. If no files are specified, standard input is read.

This utility is a line-oriented tool and it uses in-memory allocated buffer for a whole wide-char line. If the input file is huge and without line breaks than allocate the memory for the file may be unsuccessful.

OPTIONS

`-V, --version`
Display version information and exit.

`-h, --help`
Display help text and exit.

Program tac – odwrotność programu cat

- Narzędzie wyświetlające wiersze podanego wejścia w odwrotnej kolejności.
- Przykłady użycia:

▶ `tac ścieżka/do/pliku`

NAME

`tac` - concatenate and print files in reverse

SYNOPSIS

`tac` [OPTION]... [FILE]...

DESCRIPTION

Write each FILE to standard output, last line first.

With no FILE, or when FILE is -, read standard input.

Mandatory arguments to long options are mandatory for short options too.

`-b, --before`

attach the separator before instead of after

`-r, --regex`

interpret the separator as a regular expression

`-s, --separator=STRING`

use STRING as the separator instead of newline

Program tee

- Narzędzie pozwalające jednocześnie wyświetlić i zapisać podane wejście.
- Przykłady użycia:

▶ `grep '/bin/bash' /etc/passwd | tee wynik`

NAME

`tee` - read from standard input and write to standard output and files

SYNOPSIS

`tee [OPTION]... [FILE]...`

DESCRIPTION

Copy standard input to each FILE, and also to standard output.

`-a, --append`

append to the given FILES, do not overwrite

`-i, --ignore-interrupts`

ignore interrupt signals

`-p` diagnose errors writing to non pipes

`--output-error[=MODE]`

set behavior on write error. See MODE below

`--help` display this help and exit

Program tr

- Narzędzie pozwalające podmienić lub usunąć wybrane znaki z wejścia.
- Przykłady użycia:

```
▶ cat plik | tr '[a-z]' '[A-Z]' # małe litery na duże
```

```
▶ ss -tulpn | tr -s ' ' | cut -d ' ' -f 5
```

NAME

tr - translate or delete characters

SYNOPSIS

tr [OPTION]... SET1 [SET2]

DESCRIPTION

Translate, squeeze, and/or delete characters from standard input, writing to standard output.

-c, -C, --complement

use the complement of SET1

-d, --delete

delete characters in SET1, do not translate

-s, --squeeze-repeats

replace each sequence of a repeated character that is listed in the last specified SET, with a single occurrence of that character

Program cut

- Narzędzie pozwalające wybrać z wejścia określone fragmenty.
- Przykłady użycia:
 - ▶ `cut --delimiter=',' --fields=1,3,7 plik.csv`
 - ▶ `grep -v 'nologin' /etc/passwd | cut -d ':' -f 1,6`

NAME

cut - remove sections from each line of files

SYNOPSIS

cut OPTION... [FILE]...

DESCRIPTION

Print selected parts of lines from each FILE to standard output.

With no FILE, or when FILE is -, read standard input.

Mandatory arguments to long options are mandatory for short options too.

`-b, --bytes=LIST`
select only these bytes

`-c, --characters=LIST`
select only these characters

Program wc

- Narzędzie pozwalające zliczyć linie, słowa i znaki w podanym wejściu.
- Przykłady użycia:

```
▶ find katalog/ | wc --lines
```

```
▶ echo -n 'HASŁO' | wc --chars
```

NAME

```
wc - print newline, word, and byte counts for each file
```

SYNOPSIS

```
wc [OPTION]... [FILE]...
```

```
wc [OPTION]... --files0-from=F
```

DESCRIPTION

Print newline, word, and byte counts for each FILE, and a total line if more than one FILE is specified. A word is a non-zero-length sequence of characters delimited by white space.

With no FILE, or when FILE is -, read standard input.

The options below may be used to select which counts are printed, always in the following order: newline, word, character, byte, maximum line length.

```
-c, --bytes
```

```
print the byte counts
```


Część dla dociekliwych

Zadanie dodatkowe

Zadanie dodatkowe – dla zainteresowanych

Edytor strumieniowy `sed` – co i jak?

Jest o jedno z najbardziej znanych/popularnych narzędzi stosowanych w powłoce, dlatego warto dowiedzieć się jak go stosować. Proszę omówić do czego ono służy oraz jak przy jego pomocy wykonać kilka podstawowych operacji.

W szczególności – jak dokonać podmiany jakiegoś ciągu na inny, wstawić lub usunąć linię z zawartością w określonym miejscu (numeru wiersza lub przed/po linii z treścią pasującą do wzorca). W jaki sposób wykonać kilka operacji na raz oraz jak zapisać wynik przekształceń w nowym pliku, a jak zapisać je bezpośrednio w przetwarzanym pliku (tworząc przy tym jego kopię oraz nie tworząc jej – proszę rozważyć oba przypadki).