Sprawozdanie:

9. Wejście-wyjście terminalowe

Dominik Bober 303099

14 maja 2020

1 Zadanie: Alfabet Morse'a

- 1. Polecenie stty(1) służy do wyświetlania oraz zmiany ustawień linii terminala. Wyświetl nim aktualne atrybuty terminala. W jaki sposób wyświetlić wszystkie aktualne ustawienia?
- 2. Spróbuj w terminalu zmienić np. znak sterujący kasowaniem znaku na znak Ctrl+h (standardowo generowany jest Ctrl+?, gdy naciskany jest klawisz cofania):

```
stty erase [wciśnij Ctrl+h]
```

Sprawdź, wyświetlając ponownie aktualne atrybuty, czy nowy znak sterujący został ustawiony. Następnie napisz coś i spróbuj skasować znak. Aby przywrócić standardowy znak cofania wykonaj ponownie komendę podając znak cofania jako atrybut lub użyj stty sane aby wyczyścić wszystkie opcje.

- Jakie polecenie powłoki służy do odczytania nazwy pliku terminala? Jaka funkcja biblioteczna mu odpowiada (zwraca nazwę terminala)?
- Czy poleceniem stty można zmienić atrybuty innego terminala? Spróbuj zmienić liczbę kolumn w innym terminalu np. na 10.
- 3. W programie do napisania będzie należało zmodyfikować atrybuty terminala. Jeśli w trakcie pracy nad programem zmienią się ustawienia terminala, a program ich nie odtworzy, wiesz już, że działanie terminala może być inne niż oczekiwane standardowo, można wtedy użyć np. stty sane.

Aby wyświetlić wszystkie aktulane ustawienia służy komenda: stty [-F DEVICE |--file=DEVICE] [-a |--all]

Polecenie do wyświetlenia nazwy terminala: tty Można również użyć funkcji: ttyname()

2 Zadanie: Alfabet Morse'a

Przy użyciu odpowiednich funkcji i flag, ustanów nowe atrybuty terminala (zmodyfikuj odpowiednie wartości w polach struktury termios), aby w trakcie działania programu odczytywane były rzeczywiście pojedyncze znaki, a program dla każdej wczytywanej litery wyświetlał od razu jej odpowiednik w alfabecie Morse'a (przy czym program ma nie wyświetlać samej litery!).

Użyłem 2 flag: ICANON by znaki zostawały czytane pojedynczo oraz ECHO aby wprowadzane znaki nie zostawały wyświtlane.

Rysunek 1: Przykład działania programu tłumaczącego na alfabet Morse'a

3 Sesje

- 1. Użyj polecenia **setsid**, aby utworzyć nową sesję dla procesu potomnego. Uruchom program i sprawdź rezultat.
- 2. Jak przy użyciu polecenia ps wyświetlić sid procesu?
- 3. Przy użyciu pstree zaobserwuj, gdzie znajduje się proces.

Aby wyświetlić sid procesu przy pomocy ps można użyć flag ps -ejf

```
dominik@mimik: ~/Projects/c_so_projects
                                                                                            File Edit View Search Terminal Help
dominik@mimik:~/Projects/c_so_projects$ kill -9 7799
dominik@mimik:~/Projects/c_so_projects$ ./sessions
Rodzic: pid=7868 ppid=7078 sid=7078
Dziecko: pid=7869 ppid=7868 sid=7869
        Killed
^Z[1]
                                    ./sessions
[2]+ Stopped
                                 ./sessions
dominik@mimik:~/Projects/c_so_projects$ ps -ejf | (head -1 && tail -10)
           PID PPID PGID
                               SID C STIME TTY
UID
                                                              TIME CMD
                                                         00:00:02 [kworker/u24:1-p]
root
           7086
                           0
                                  0 0 13:54 ?
dominik
           7149
                 2544 2575
                               2575 0 13:56 ?
                                                         00:00:09 gedit /home/dominik/Project
s/c_so_projects/sessions.c
                                                         00:00:00 [kworker/6:1-eve]
00:00:00 [kworker/4:0-eve]
           7255
                           0
                                  0
                                     0 14:00 ?
root
root
           7265
                           0
                                  0
                                     0 14:00
                                                         00:00:00 [kworker/0:0-cgr]
00:00:00 [kworker/u24:0+e]
root
           7519
                     2
                            0
                                   0
                                     0 14:09 ?
root
           7601
                            0
                                   0
                                        14:11 ?
dominik
                                                         00:00:00 ./sessions
           7868
                 7078
                        7868
                               7078
                                     0 14:21 pts/0
                                                         00:00:00 ./sessions
dominik
                        7869
                               7869
           7869
                  7868
                                      0 14:21 ?
dominik
                                                         00:00:00 ps -ejf
           7879
                  7078
                         7879
                                7078
                                      0
                                        14:21 pts/0
dominik
           7880
                  7078
                         7879
                                7078
                                      0 14:21 pts/0
                                                         00:00:00 bash
                              so_projects$
```

Rysunek 2: Zrzut terminala zawierający output programu oraz ps

```
dominik@mimik: ~/Projects/c_so_projects
File Edit View Search Terminal Help
                      dbus-daemon
                      ibus-portal-
                                      -2*[{ibus-portal}]
                      pulseaudio-
                                     -2*[{pulseaudio}]
                      xdg-permission-
                                           -2*[{xdg-permission-}]
           systemd
                      (sd-pam)
                      -at-spi-bus-laun
                                           -dbus - daemon
                                           -3*[{at-spi-bus-laun}]
                      -at-spi2-registr
                                           -2*[{at-spi2-registr}]
                      -dbus-daemon
                      -dconf-service
                                         -2*[{dconf-service}]
                                           evolution-addre
4*[{evolution-addre}]
4*[{evolution-calen}]
                      evolution-addre
                      evolution-caler
                                           evolution-calen—
                                           4*[{evolution-calen}
                                           3*[{evolution-sourc}]
                      evolution-sourc
                      gedit---3*[{gedit}]
gnome-shell-cal---5
                                           -5*[{gnome-shell-cal}]
                                                   pstree
                      gnome-terminal-
                                           -bash
                                                   sessions
                                                                sessions
                                           3*[{gnome-terminal-}]
                      -aoa - daemon
                                     -3*[{goa-daemon}]
                      -aoa-identitv-se
                                           -3*[{goa-identity-se}]
                      gvfs-afc-volume
                                           -3*[
                                               {gvfs-afc-volume]
```

Rysunek 3: Zrzut terminala zawierający wycinek drzewo procesów z zaznaczonym miejscem występowania ./sessions

4 Zadanie: Totolotek

Przy użyciu funkcji setsid stwórz demona, który będzie pracował nawet po wylogowaniu się użytkownika. Demon co kilkadziesiąt sekund ma losować 6 liczb z przedziału [1,49] i zapisywać je do pliku 'wyniki_lotto'. Pouruchomieniudemona, zamkni

Podpowiedź: Aby zaimplementować demona postępuj następująco:

- 1. Stwórz proces potomny.
- 2. Zakończ przy użyciu funkcji exit proces rodzica.
- 3. Aby demon miał pełne uprawnienia do plików przez siebie utworzonych ustaw umask na 0.
- 4. Utwórz nową sesję dla demona. (Obsłuż ew. błąd: w razie gdy nie powiodło się utworzenie, proces powinien się
- 5. zakończyć!).
- 6. Ewentualnie można zmienić katalog bieżący dla demona np. na katalog główny lub /tmp (w razie gdyby np. w
- 7. międzyczasie aktualny katalog bieżący został usunięty).
- 8. Zamknij standardowe deskryptory plików. (Dlaczego?)
- 9. Napisz kod odpowiedzialny za działanie demona. Możesz w tym celu wywołać odpowiednie polecenia powłoki, np.

```
system("shuf -i1-49 -n 6 > wyniki_lotto.txt");
```

Mój program co 3 sekundy za pomocą wyrzej wymienionego polecenia wpisuje do pliku wyniki_lotto.txt 6 losowych liczb.

Tak jak w poleceniu zamkąłem 3 standardowe strumienie - STDIN, STDOUT i STDERR. Te kanały komunikacji są zbędne i są potencjalną luką w zabezpieczeniach, ponieważ demon z definicji nie może korzystać z terminala.

```
dominik@mimik: ~/Projects/c_so_projects
                                                                         File Edit View Search Terminal Help
dominik@mimik:~/Projects/c_so_projects$ cat wyniki_lottot.txt
cat: wyniki_lottot.txt: No such file or directory
4740 pts/0
              00:00:00 bash
5180 pts/0
              00:00:00 ps
dominik@mimik:~/Projects/c_so_projects$ cat wyniki_lottot.txt
30
32
14
36
dominik@mimik:~/Projects/c_so_projects$ cat wyniki_lottot.txt
12
32
14
36
dominik@mimik:~/Projects/c_so_projects$ cat wyniki_lottot.txt
39
14
26
49
dominik@mimik:~/Projects/c_so_projects$ cat wyniki_lottot.txt
16
35
30
19
dominik@mimik:~/Projects/c_so_projects$ cat wyniki_lottot.txt
28
17
25
dominik@mimik:~/Projects/c_so_projects$ cat wyniki_lottot.txt
28
17
dominik@mimik:~/Projects/c_so_projects$
```

Rysunek 4: Zrzut terminala zawierający wyniki działania programu totolotek