Kurs administrowania systemem Linux 2019

Lista zadań na pracownię nr 4

Na zajęcia 18 marca 2019

Zadanie 1 (po 1 pkt. za każde rozwiązanie). Rozważany w zadaniu 1 z poprzedniej listy program `mp3play` wypisywał listę utworów w konsoli tekstowej (wykorzystując np. instrukcję `select` powłoki). W przypadku dużej liczby utworów takie rozwiązanie jest niewygodne. Wolelibyśmy, by lista utworów została wyświetlona w oknie (na pełnoekranowej konsoli tekstowej lub w środowisku graficznym) tak, by użytkownik mógł nawigować wśród utworów za pomocą klawiszy strzałek oraz klawiszy `<PgUp>`, `<PgDn>`, `<Home>`, `<End>` itp., wyszukiwać tytuły na te liście wpisując kilka początkowych liter itp. oraz zatwierdzać wybór klikając na odpowiedni przycisk (w środowisku graficznym), bądź naciskając klawisz `<Enter>`.

Wiele programów pozwala na bardzo łatwe tworzenie okien dialogowych zarówno w konsoli tekstowej (np. `whiptail`, `dialog`) jak i w środowisku graficznym (np. `zenity`). Użyj ich w swoich rozwiązaniach. Możesz też wypróbować np. `gxmessage`, `kdialog`, `yad`, `smenu` lub inny podobny program, według uznania i własnych preferencji.

Zadanie 2 (1 pkt). Niezbyt rozgarniętego początkującego użytkownika Linuksa irytuje konieczność poprzedzania nazwy programu ciągiem `./`, jeśli chce uruchomić program znajdujący się w bieżącym katalogu. Dodał zatem do swojego pliku `~/.profile` wiersz postaci

```
PATH=.:$PATH
```

Przygotuj archwium `lolcats.tar` zawierające kilka zdjęć kotów wraz z niewielkimi dodatkami, tak by wykonanie ciągu poleceń

```
$ tar xf lolcats.tar
$ ls
```

wyrządziło użytkownikowi *wielką krzywdę*. Dobrym kandydatem na *wielką krzywdę* jest wypisanie tekstu „You've been pwnd!" (polskie tłum.: „mam cię") lub uruchomienie programu `sl` (w Debianie instalowanego przez pakiet o tej samej nazwie).

Zadanie 3 (po 1 pkt. za każdy racjonalny eksploit). Opracuj (możesz się inspirować istniejącymi rozwiązaniami) inne eksploity podobne do opisanego w zadaniu 2 (np. *privilege escalation* wykorzystujący względne ścieżki do programów wywoływanych w skryptach, programy wykorzystujące SUID, nadużycia `sudo` itp.). *Uwaga:* przesłanie użytkownikowi informacji, że uruchomienie polecenia `rm -rf ~` powoduje wyświetlenie atrakcyjnych *lolcatów* oraz *Albański Wirus Komputerowy* mimo wielkiej siły rażenia nie są racjonalnymi eksploitami, gdyż odwołują się wyłącznie do socjotechniki i nie wykorzystują żadnych podatności technicznych.

Zadanie 4 (1 pkt). Wykorzystaj program `getopt(1)` w skrypcie `hwb`, którego zadaniem jest wypisywanie na ekranie pozdrowień typy „Hello, world!". Dla każdego argumentu *imię* program powinien wypisać wiersz postaci

```
Hello, imię!
```

Program powinien obsługiwać przynajmniej następujące opcje:

`-c`, `--capitalize` wypisanie imienia bądź słowa `world` wielką literą;

`--color=`[`never` | `auto` | `always`] kolorowanie imion (nigdy, tylko gdy standardowy strumień wyjściowy jest konsolą, zawsze), por. podobną opcję programu `ls(1)`;

`-g` *`text`*, `--greeting=`*`text`* zastąpienie słowa `Hello` podanym tekstem;

`-h`, `--help` wypisanie krótkiej ściągi;

`-v`, `--version` wypisanie nazwy i wersji programu oraz copyrightu;

`-w`, `--world` wypisanie dodatkowo wiersza `Hello, world!`

Zadanie 5 (1 pkt). Wykorzystując funkcję `getopt_long(3)` zaprogramuj w C program `hwc` działający podobnie do skryptu z zadania 4.

Zadanie 6 (1 pkt). Napisz stronę podręcznika `hwb(1)`. Zastosuj konwencje opisane w rozdziale DESCRIPTION na stronie `man(1)` (w szczególności umieść co najmniej rozdziały NAME, SYNOPSIS, DESCRIPTION, OPTIONS, EXIT STATUS, EXAMPLE, AUTHORS i SEE ALSO oraz przestrzegaj konwencji zapisywania niektórych fraz kursywą i pismem pogrubionym).

Zadanie 7 (1 pkt). Przygotuj dokumentację `hwb.texi` zgodnie z konwencjami dokumentacji GNU. Wygeneruj wersje `info`, `html` i `pdf`.

Zadanie 8 (2 + 5 pkt. bonusowych). Napisy zawierające znaki narodowe zamienia się na ciągi znaków ASCII stosując jedną z dwóch technik: 1) pomijanie znaków spoza kodowania ASCII, 2) usuwanie znaków diakrytycznych (´ , ` , ¨ itp.) i zastępowanie znaków nieposiadających łacińskich odpowiedników specjalnymi znakami, np. `?`. Pierwszy sposób prowadzi często do zupełnie nieczytelnych napisów (np. `Za gl ja wia` — „Zażółć gęślą jaźń żółwia”, `d` — „Łódź” itp.) lub wyjątkowo niezręcznych sytuacji (np. serwis `ikea.pl` generuje nazwy plików z instrukcją montażu mebli usuwając polskie znaki z nazw mebli i dodając rozszerzenie `.pdf`, co nie jest zbyt fortunne w przypadku takich artykułów, jak „płyta główna”). Drugi sposób jest nieco lepszy, czasem jednak też prowadzi do niejednoznaczności (np. „połowa kółka, pięć gosposi i kąt”, „okaż mi łaskę” itp.).

Napisz skrypt `ascify`, który uruchomiony w danym katalogu wyszuka nazwy zawierające znaki spoza zbioru ASCII oraz znaki mające specjalne znaczenie dla powłoki (w tym metaznaki, znaki używane we wzorcach nazw plików itp.), zaproponuje dla każdego z nich zmianę na nazwę nie zawierającą wyżej wymienionych znaków, np.

`Np. zażółcić gęślą jaźń żółwia?.pdf` → `Np_zazolcic_gesla_jazn_zolwia.pdf`

i w razie akceptacji użytkownika wykona tę zmianę. Uwagi i przypadki brzegowe:

- Skrypt powinien radzić sobie z kolizjami nazw, np. dodając na końcu nazwy unikatowy numer.
- Warto rozważyć opuszczanie znaków interpunkcyjnych, kompresowanie ciągów spacji itp.
- Rozważ wykorzystanie programów `iconv(1)` i `sed(1)`.

Rozszerzenia (1 punkt bonusowy za każde):

1. Opcje:

   `-y`, `--yes` zmiana nazw będzie wykonana bez pytania użytkownika o zgodę;
   `-n`, `--dry-run` wypisuje proponowane nazwy, ale nie pyta użytkownika o akceptację i nie dokonuje zmiany;
   `-h`, `--help` wypisanie krótkiej ściągi;
   `-v`, `--version` wypisanie nazwy i wersji programu oraz copyrightu;
   `-d`, `--directory` wykonanie zmiany nazw także dla nazw katalogów;
   `-r`, `--recursive` wykonanie zmiany nazw także dla plików znajdujących się w podkatalogach;
   `-e` *`enc`*, `--encoding=`*`enc`* interpretacja oryginalnej nazwy według podanego kodowania *`enc`*.

Domyślnym kodowaniem znaków powinno być UTF-8.

2. Umożliwienie ręcznej edycji zaproponowanej nazwy: „yes/no/edit”.
3. Sensowna transliteracja cyrylicy, np.

   `Батогов Е. В.: Gentoo Linux. Сборник статей.pdf` →
   `Batogov_Je_V_Gentoo_Linux_Sbornik_statjej.pdf`

4. Strona podręcznika `ascify(1)` dla programu.
5. Dokumentacja `texi` programu. Wygeneruj wersje `info`, `html` i `pdf`.

Uwaga: Pliki skopiowane z Internetu (por. powyższy przykład dokumentacji Gentoo) są zapisywane przez przeglądarkę pod oryginalnymi nazwami. Program `ascify(1)` bardzo by się przydał do uporządkowania skopiowanych plików. Postaraj się napisać go tak, żeby faktycznie był użytecznym narzędziem. Prowadzący bardzo go potrzebuje.