

# Podstawy Programowania: Praca z uniksowym interpreterem poleceń oraz systemem X Window"

Dawid Krekora

16 grudnia 2020

# 1 Opis projektu

Głównym założeniem projektu było zapoznanie się z możliwościami które oferuje praca w powłoce systemowej BASH. Sprawozdanie zawiera opis działania skryptów operujących (w głównej mierze) na plikach dokonując na nich różnych modyfikacji, zgodnie z życzeniem użytkownika.

## 2 Opis działania skryptów

### 2.1 Wyświetlanie informacji o wielkości katalogów

Alias są komendami dostępnymi do użytku w powłoce systemowej Unix'a bądź Linux'a. Umożliwiają zastąpienie dłuższych wyrażeń, mało wygodnych do wpisywania, ich wersjami skrótowymi. Wyrażenie podstawione pod wybraną przez nas instrukcję zastępuje ją i może być użytkowane z taką samą skutecznością jak pierwowzór. Takie rozwiązanie generuje szereg korzyści dla użytkownika, takich jak zwiększenie efektywności pracy (krótszy czas wprowadzania poleceń) czy też wykluczenie szansy popełnienia błędów składni. W celu utworzenia aliasu należy otworzyć konsolę terminala, a następnie dokonać podstawienia, np.:

```
alias c='clear'
```

Alias który utworzyliśmy wprowadza ułatwienie przy czyszczeniu zawartości ekranu: od teraz zamiast wpisywać polecenie 'clear' możemy pisać 'c', zadziałają one z identycznym skutkiem.

Przedstawiona inicjalizacja aliasów ma jedną, zasadniczą wadę - aliasy są dostępne tylko podczas bieżącego cyklu pracy konsoli. W momencie gdy zamkniemy procesy i ją wyłączymy, przypisania stracą swoje własności. Dużo lepszym rozwiązaniem jest utworzenie ich permanentnych wersji, co wbrew pozorom nie jest aż tak trudne. W tym celu tworzymy (bądź edytujemy) plik systemowy `.bash_aliases` który jest domyślnie plikiem ukrytym znajdującym się w katalogu `/home/nazwa_uzytkownika/.bash_aliases`. Wpisujemy w konsoli:

```
gedit ~/.bash_aliases
```

a następnie w wyświetlonym pliku tekstowym:

```
alias nazwa_aliasu='polecenie'
```

Pierwszym zadaniem na laboratorium było utworzenie permanentnego aliasu `lhs` który wyświetli listę podkatalogów bieżącego katalogu, posortowaną według ich wielkości, z podaniem tej wielkości w formacie czytelnym dla człowieka (odpowiednio do wielkości katalogów w bajtach, kilobajtach, megabajtach). We wcześniej utworzonym pliku wpisujemy:

```
alias lhs="du -h | sort -h"
```

Po zapisaniu pliku i ponownym otwarciu konsoli alias zostanie poprawnie zaimplementowany.

## 2.2 Wykonywanie operacji na plikach o nazwie pasującej do wzorca

**Zadanie do wykonania:** Napisz polecenie powłoki, które z bieżącego katalogu i jego wszystkich podkatalogów usunie pliki pasujące do podanego wzorca.

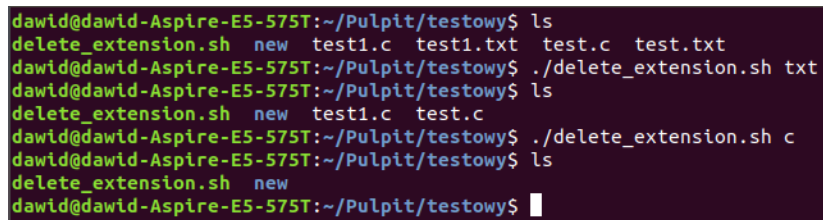
**Opis działania:** skrypt usuwa wszystkie pliki z wybranym przez użytkownika rozszerzeniem (podanym jako argument skryptu) z bieżącego katalogu oraz wszystkich jego podkatalogów.

**Implementacja:**

```
1  #!/bin/bash
2  find ./ -name "*. $1" -exec rm {} \;
3
```

Listing 1: Kod programu

Test skryptu:



```
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ls
delete_extension.sh new test1.c test1.txt test.c test.txt
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ./delete_extension.sh txt
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ls
delete_extension.sh new test1.c test.c
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ./delete_extension.sh c
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ls
delete_extension.sh new
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$
```

Rysunek 1: Test działania skryptu delete\_extension.sh

## 2.3 Przeszukiwanie ostatnio modyfikowanych plików

**Zadanie do wykonania:** Napisz polecenie powłoki, które w plikach zmodyfikowanych w ostatnim czasie (tygodniu/miesiącu) wyszukają wskazaną frazę.

**Opis działania:** Przeszukiwanie katalogu i podkatalogów w czasie podanym w pierwszym argumencie wywołania (-t dla 7 dni i -m dla 30 dni) w poszukiwaniu wyrażeń podanych jako drugi argument wywołania. Skrypt posiada zabezpieczenie przed wystąpieniami białych znaków w nazwie pliku (znaki te będą ignorowane).

**Implementacja:**

```
1  #!/bin/bash
2  case "$1" in
3  -t) find . -type f -mtime -7 -print0 | xargs -0 grep -i -l "$2";;
4  -m) find . -type f -mtime -30 -print0 | xargs -0 grep -i -l "$2";;
5  esac
6
```

Listing 2: Kod programu

Test skryptu:

```
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ls
'1 1.PNG' '4 4 .JPG' new search.sh
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ./search.sh -t bin
./search.sh
./new/delete_extension.sh
./new/powieksz.sh
./new/zmniejsz.sh
./new/kopowanie.sh
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ./search.sh -m gnome
./4 4 .JPG
./1 1.PNG
```

Rysunek 2: Test działania skryptu search.sh

## 2.4 Modyfikacja nazw plików

**Zadanie do wykonania:** Napisz polecenie powłoki, które wszystkim plikom w formacie JPEG/PNG z katalogu bieżącego zmieni rozszerzenia na pisane małymi literami. **Opis działania:** Po uruchomieniu skrypt zmienia rozszerzenia pisane wielkimi literami na małe. Ponadto informuje gdy nie znalazł rządanych plików w forlderze. **Implementacja:**

```
1 #!/bin/bash
2 for file in *.PNG; do
3 mv $file `basename $file .PNG`.png; done
4 for file in *.JPEG; do
5 mv $file `basename $file .JPEG`.jpeg; done
6
```

Listing 3: Kod programu

Test skryptu:

```
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ls
1.jpeg 2.png new powieksz.sh zmniejsz.sh
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ./powieksz.sh
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ls
1.JPEG 2.PNG new powieksz.sh zmniejsz.sh
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ./zmniejsz.sh
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ls
1.jpeg 2.png new powieksz.sh zmniejsz.sh
```

Rysunek 3: Test działania skryptu zmniejsz.sh

## 2.5 Porządkowanie plików

**Zadanie do wykonania:** Napisz skrypt, który ze wskazanego katalogu skopiuje wszystkie pliki ze zdjęciami w formacie JPEG/PNG, zmieni skopiowanym plikom rozszerzenia na pisane małymi literami, zastąpi występujące w nazwach plików spacje znakiem podkreślenia, zmieni rozmiar zdjęć do wskazanego argumentem podanym w wywołaniu skryptu i w końcu utworzy archiwum z tak przygotowanych plików.

**Opis działania:** Skrypt uruchamiamy z dwoma argumentami wywołania: pierwszy to ścieżka katalogu w którym chcemy przeprowadzić operacje, drugi to rozmiar obrazu który chcemy uzyskać po konwersji. Po uruchomieniu tworzy kopie wszystkich plików w formacie JPEG/PNG zapisując ich rozszerzenia małymi literami, a jeżeli w nazwach plików wystąpiły znaki białe, są one zastępowane przez podkreślenia. Po konwersji do ustalonego rozmiaru kopie są przenoszone do nowo powstałego archiwum o zadanej w skrypcie nazwie.

**Implementacja:**

```
1  #!/bin/bash
2  cd "$1"
3  for files in *.JPEG; do
4  cp "$files" "${files/.JPEG/.jpeg}"; done
5  for files in *.PNG; do
6  cp "$files" "${files/.PNG/.png}"; done
7  for files in *.jpeg *.png; do
8  mv "$files" `echo $files | tr ' ' '_`; done
9  for files in *.jpeg *.png; do
10 convert "$files" -resize "$2" "$files"; done
11 tar -cvf Photos.tar *.jpeg *.png --remove-files
12
```

Listing 4: Kod programu

**Test skryptu:**



```
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ls
'1 1 1.PNG' '2 2 2.JPEG' new test.sh
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ./test.sh ./ 200
2_2_2.jpeg
1_1_1.png
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ls
'1 1 1.PNG' '2 2 2.JPEG' new Photos.tar test.sh
dawid@dawid-Aspire-E5-575T:~/Pulpit/testowy$ ls
```

Rysunek 4: Test działania skryptu test.sh

### 3 Podsumowanie i wnioski

Skrypty swoją strukturą i pewną tendencją zachowań mają bardzo dużo wspólnego z innymi językami programowania. Bardziej restrykcyjnie podchodzą jednak do samej składni i mniej wybaczą ewentualne błędne decyzje użytkownika. Wynika to z faktu, że sama idea skryptów ma za zadanie zoptymalizować i ułatwić pewne operacje wykonywane w systemie. Operacje które przeprowadzaliśmy w ramach laboratorium ingerowały prawie zawsze w strukturę danych, brak odpowiednich zabezpieczeń mogły spowodować niekiedy nieodwracalne zmiany. Kolejną wadą którą udało się zaobserwować jest na pewno dosyć skomplikowana wymiana informacji z użytkownikiem. Można to tłumaczyć dosyć słabą znajomością pracowania w tym środowisku twórcy tego sprawozdania jednak same komunikaty sprawiają nieodparte wrażenie bardzo ogólnych".

Skrypty zostały w pełni przetestowane na komputerze lokalnym. Na serwerach zdalnych diablo i panamint wystąpiły problemy w działaniu niektórych (głównie wynikające z pewnych niedozwolonych implementacji składni takich jak np. polecenie `print0`). Wynikać to może z różnych wersji systemowych i braku udoskonalenia pewnych elementów języka na przestrzeni lat. Całokształt korzyści wynikających z użytkowania skryptów zostawia bardzo pozytywne wrażenie, warto zagłębić się w temat i samemu opracować kilka spersonalizowanych dla własnego środowiska. Oszczędność 10 sekund dziennie może wydawać się śmieszną ilością jednak na przestrzeni roku daje to całą godzinę, resztę pozostawiam do własnej interpretacji.