12.05.2020r.
19.05.2020r. (gr. 1)
20.05.2020r. (gr. 2)

W ramach wstępu proszę o przypomnienie niektórym Waszym kolegom/koleżankom o oddawanie/przesyłanie prac. Wysyłacie je ZAWSZE na tego samego maila- **dawidkoch.szkola@gmail.com** W razie problemów czy innych spraw, które uniemożliwiają Wam oddanie pracy, proszę o kontakt przez tego maila. Gdy prace nie będą oddawane w terminie (i tak staram się zadawać stosunkowo mało, by Was nie przeładować obowiązkami) o sytuacji będą informowani Wasi wychowawcy.

Z TEGO TEMATU NIE MA PRACY DOMOWEJ

Programy narzędziowe w środowisku tekstowym systemu Linux

Cały dział znajduje się w książce od strony 187 do 195. Zagadnienia:

- Edytor tekstu vi,
- Skrypty powłoki,
- Archiwizacja zbiorów systemu Linux,
- Kompresja zbiorów systemu Linux,
- Publikacje elektroniczne systemu Linux.

Materiały uzupełniające i rozszerzające wiedzę w podręczniku:

1. Edytor tekstu vi:

https://mlodytechnik.pl/eksperymenty-i-zadania-szkolne/wynalazczosc/29700-edytory-tekstu

Vim możecie sobie zainstalować na Windowsie- https://download.komputerswiat.pl/biuro-i-praca/edytory-tekstu/vim

Tutorial- https://openvim.com/

2. Skrypty powłoki:

https://blog.helion.pl/blyskawiczny-kurs-pisania-skryptow-powloki/

https://devopsiarz.pl/bash/tutorial-kurs-pisania-dobrych-skryptow-bash-wstep-dobre-praktyki-shellcheck/#przyk%C5%82ad-dobrego-skryptu-wersja-poprawiona

https://www.youtube.com/watch?v=Y3YP6BiTjsg

https://bash.0x1fff.com/zaawansowane/organizacja-kodu.html

3. Archiwizacja zbiorów systemu Linux

https://pasja-informatyki.pl/sieci-komputerowe/linux-ubuntu-server-archiwizacja-kompresja-danych/
http://mfranczak.pl/archiwizacja-i-kompresja-zbiorow-systemu-linux/

4. Kompresja zbiorów system Linux

   http://mfranczak.pl/archiwizacja-i-kompresja-zbiorow-systemu-linux/

   https://linuxize.com/post/gzip-command-in-linux/

5. Publikacje elektroniczne systemu Linux

   Praktycznie na każdej lekcji podaję Wam strony z informacjami o Linuxie.
   Dwie kolejne:
   https://linux-magazine.pl/
   https://www.linux.pl/

17 Programy narzędziowe w środowisku tekstowym systemu Linux

ZAGADNIENIA

- Posługiwanie się edytorem tekstu vi
- Tworzenie skryptów powłoki
- Kompresja i archiwizacja
- Tworzenie archiwów w systemie Linux
- Kompresowanie i dekompresowanie plików i katalogów
- Publikacje elektroniczne dotyczące systemu Linux

17.1. Edytor tekstu vi

Edytor vi jest dostępny we wszystkich klonach systemu Unix/Linux, a w większości systemów jest edytorem domyślnym. Niektóre programy, np. cron, wykorzystują vi jako swój bazowy edytor opcji. Sposób obsługi vi jest różny od sposobu obsługi edytorów pracujących w środowisku Windows. Każdy użytkownik Linuksa powinien więc poznać vi przynajmniej w takim stopniu, aby mógł wykorzystać go do edycji plików konfiguracyjnych. Istnieje w zasadzie tylko jedna metoda nauczenia się obsługi – trzeba ćwiczyć w praktyce. Na potrzeby początkujących użytkowników vi opracowano samouczek, który w ciągu kilkudziesięciu minut pozwoli na przećwiczenie podstawowych możliwości edytora. Interaktywny samouczek programu vi w wersji angielskiej jest dostępny m.in. na stronie `http://www.openvim.com/tutorial.html` lub w wersji polskiej na wielu stronach przygotowanych dla początkujących użytkowników.

Istnieje kilka wersji edytora vi. Najpopularniejsza i najbardziej rozbudowana jest wersja VIM, która ma też najwięcej opcji i możliwości, lecz zasady pracy w obu programach są podobne. Edytor vi pracuje w środowisku tekstowym; może być uruchomiony za pomocą polecenia `vi`. Jeżeli trzeba utworzyć nowy lub otworzyć istniejący plik, można od razu podać jego nazwę, np. `vi plik.txt`.

Na ekranie można wydzielić dwa obszary robocze: **obszar poleceń** (ostatnia linia na dole ekranu) i **obszar edycji tekstu** (pozostałe linie). Edytor vi ma dwa tryby pracy: tryb poleceń do wydawania poleceń określających, co chce się zrobić, oraz tryb edycji, w którym edytuje się tekst. Po uruchomieniu vi domyślnie znajduje się w trybie poleceń. Aby przejść do trybu edycji, należy nacisnąć klawisz [INS] (insert); aby przejść do trybu poleceń, należy nacisnąć klawisz [Esc].

Aby zakończyć pracę z vi, należy nacisnąć: [Esc] `:q!` – bez zapisu tekstu do pliku, [Esc] `:wq` – z zapisem tekstu do pliku.

Przemieszczanie się po tekście odbywa się za pomocą klawiszy kursorów w obu trybach pracy, a dodatkowo w trybie poleceń można użyć przycisków [h] – lewo, [j] – dół, [k] – góra, [l] – prawo. Liczba opcji, jakimi dysponuje vi, jest bardzo duża. Aby poznać najczęściej używane, należy uruchomić samouczek i wykonać wskazane ćwiczenia.

17.2. Skrypty powłoki

Skrypty są zwykłymi plikami tekstowymi, w których są zapisane polecenia zrozumiałe dla powłoki. Zadaniem powłoki jest przetłumaczenie ich na polecenia systemu. Aby przygotować skrypt, należy:

- stworzyć plik, w którym trzeba umieścić kod, np.

```
touch naszskrypt.bat
```

- za pomocą dowolnego edytora tekstu wpisać do pliku kolejno wykonywane polecenia;
- nadać uprawnienia do wykonywania pliku dla uprawnionych użytkowników;
- uruchomić skrypt:

```
./naszskrypt.bat
```

gdzie symbol `./` oznacza, że skrypt znajduje się w bieżącym katalogu. Przykładowy, bardzo prosty skrypt może się składać z następujących poleceń:

```
#!/bin/bash
#Tu jest komentarz. echo „Hello World"
```

Pierwsza linia skryptu zaczynająca się od znaków **`#!`** ma szczególne znaczenie – wskazuje na rodzaj powłoki, w której skrypt ma być wykonany. Tutaj skrypt zawsze będzie wykonywany przez interpreter poleceń **`/bin/bash`**, niezależnie od tego, jakiego rodzaju powłoki używamy w danej chwili.

Znak **`#`** (hasz) oznacza komentarz – wszystko, co znajduje się za nim w tej samej linii, jest pomijane przez interpreter.

Trzecia linia spowoduje wydrukowanie na standardowym wyjściu (*stdout*), czyli na ekranie, napisu: **`„Hello World"`**.

W języku powłoki występują pewne słowa zastrzeżone, np.:

!	done	esac	function	Select	while	time
case	elif	Fi	If	Then	{	[
Do	else	for	in	until	}	]

Najczęściej używane polecenia w skryptach:

- **`echo`** – służy do wydrukowania napisu na standardowym wyjściu. Można też przekierować strumień danych do pliku;
- **`read`** – czyta ze standardowego wejścia pojedynczy wiersz;
- **zmienne programowe** (ang. *program variables*) – zmienne definiowane samodzielnie przez użytkownika, np. zmienna=„wartość" (nie może być spacji przed i po „="). Do zmiennej można odwołać się przez podanie jej nazwy poprzedzonej znakiem $, np. dla zmiennej x może to wyglądać następująco **`echo $x`**;
- **zmienne specjalne** (ang. *special variables, special parameters*) – to najbardziej prywatne zmienne powłoki. Są one udostępniane użytkownikowi tylko do odczytu (istnieją jednak wyjątki). Kilka przykładów zmiennych specjalnych:
  - **`$0`** – nazwa bieżącego skryptu lub powłoki;
  - **`$1..$9`** – parametry przekazywane do skryptu (uwaga! tu wyjątek – użytkownik może modyfikować ten rodzaj zmiennych specjalnych);
  - **`$?`** – kod powrotu ostatnio wykonywanego polecenia;
  - **`$$`** – PID procesu bieżącej powłoki;

- zmienne środowiskowe (ang. *environment variables*) – definiują środowisko użytkownika, dostępne dla wszystkich procesów potomnych, np.:
  - `$HOME` – ścieżka do katalogu domowego użytkownika;
  - `$USER` – login użytkownika;
  - `$HOSTNAME` – nazwa hosta użytkownika;
  - `$OSTYPE` – rodzaj systemu operacyjnego.
- instrukcja warunkowa `if` – sprawdza, czy warunek jest prawdziwy; jeśli tak, to wykonane zostanie polecenie lub będą wykonane polecenia znajdujące się po słowie kluczowym `then`. Instrukcja kończy się słowem `fi`. Składnia polecenia jest następująca:

```
if warunek then
poleceniel else
polecenie2 fi
```

- test – służy do sprawdzania warunków. Składnia polecenia:

```
test wyrażeniel operator wyrażenie2
```

może też być zapisane w nawiasach kwadratowych:

```
[ wyrażeniel operator wyrażenie2 ]
```

Między nawiasami a treścią warunku muszą być spacje, tak jak powyżej.
Polecenie `test` zwraca wartość `0` (*true*), jeśli warunek jest spełniony, i wartość `1` (*false*), jeśli warunek nie jest spełniony, np. `test -e plik`

Wybrane przykłady operatorów polecenia `test`:

`-e`	plik istnieje;
`=`	sprawdza, czy wyrażenia są równe;
`!=`	sprawdza, czy wyrażenia są różne;
`-d`	wyrażenie istnieje i jest katalogiem;
`-r`	można czytać plik;
`-w`	można zapisywać do pliku;
`-x`	można plik wykonać;
`-lt`	mniejsze niż;
`-gt`	większe niż;
`-ge`	większe lub równe;
`-le`	mniejsze lub równe.

- instrukcja `case` – pozwala na dokonanie wyboru spośród kilku wzorców. Sprawdzana jest wartość zmiennej po słowie kluczowym `case` i porównywana ze wszystkimi wariantami po kolei. Jeśli dopasowanie zakończy się sukcesem, zostanie wykonane polecenie lub będą wykonane polecenia przypisane do danego wzorca. W przeciwnym wypadku zostanie użyte polecenie domyślne oznaczone gwiazdką „*”. Składnia polecenia:

```
case zmienna in
„wzorzec1”) polecenie1 ;;
„wzorzec2”) polecenie2 ;;
„wzorzec3”) polecenie3 ;;
*) polecenie_domyślne
esac
```

Od razu nasuwa się możliwość zastosowania wewnątrz niej instrukcji **case**:

```
#!/bin/bash
echo „Co wybierasz?"
select y in X Y Z Quit
do
case $y in
„X")  echo „Wybrałeś X";;
„Y")  echo „Wybrałeś Y";;
„Z")  echo „Wybrałeś Z" ;;
„Quit") exit ;;
*) echo „Nic nie wybrałeś"
Esac
Break
done
```

- pętla **while** – najpierw sprawdza warunek, czy jest prawdziwy; jeśli tak, to wykonane zostanie polecenie lub będzie wykonana lista poleceń zawartych wewnątrz pętli. Jeśli warunek jest fałszywy, pętla zostanie zakończona. Składnia polecenia:

```
while warunek
do
polecenie
done
```

PRZYKŁAD 17.3

```
#!/bin/bash
x=1;
while [ $x -le 10 ] do
echo „Napis pojawił się po raz: $x"
x=$[x + 1]
done
```

- pętla **until** – sprawdza, czy warunek jest prawdziwy; jeśli jest fałszywy, to zostało wykonane polecenie lub była wykonana lista poleceń zawartych wewnątrz pętli, między słowami kluczowymi **do** a **done**. Pętla **until** kończy swoje działanie w momencie, gdy warunek stanie się prawdziwy. Składnia polecenia:

```
until warunek
do
polecenie
done
```

PRZYKŁAD 17.4

```
#!/bin/bash
x=1;
until [ $x -ge 10 ] do
echo „Napis pojawił się po raz: $x"
x=$[x + 1]
done
```

17.3. Archiwizacja zbiorów systemu Linux

Podstawowym narzędziem do obsługi archiwów w Linuksie jest program **tar**. Z **tar** tworzy nieskompresowane archiwum. W archiwum może znajdować się wiele p i katalogów. Program **tar** domyślnie tworzy archiwum rekurencyjnie (z podkatalog umieszczając w nim wszystko, co znajdzie we wskazanym katalogu (w tym pliki i ka ukryte).

Składnia polecenia **tar** jest następująca:

```
tar opcje nazwa_archiwum plik
```

Najczęściej używane opcje polecenia **tar** to:

- **c** – tworzenie archiwum;
- **v** – podczas przetwarzania archiwum wyświetlane będą nazwy zbiorów;
- **f** – użycie wskazanego pliku, jako archiwum;
- **x** – wyodrębnienie zbiorów z archiwum.

Przykłady obsługi archiwum pokazano na rys. 17.1.

/home/sylwia : bash

Plik Edycja Widok Zakładki Ustawienia Pomoc

```
root@sylwia-Kubuntu:~/PLIKI# ls -la
razem 12
drwxr-xr-x  3 root root 4096 lip 31 23:54 .
drwx------ 10 root root 4096 lip 31 23:50 ..
-rw-r--r--  1 root root    0 lip 31 23:53 plik1.txt
-rw-r--r--  1 root root    0 lip 31 23:53 plik2.txt
-rw-r--r--  1 root root    0 lip 31 23:53 plik6.txt
-rw-r--r--  1 root root    0 lip 31 23:53 plik8.txt
-rw-r--r--  1 root root    0 lip 31 23:53 plik.txt
drwxr-xr-x  2 root root 4096 lip 31 23:54 rozpakowane
root@sylwia-Kubuntu:~/PLIKI# tar -cvf arch.tar plik1.txt plik2.txt plik8.txt
plik1.txt
plik2.txt
plik8.txt
root@sylwia-Kubuntu:~/PLIKI# ls -la
razem 24
drwxr-xr-x  3 root root  4096 lip 31 23:56 .
drwx------ 10 root root  4096 lip 31 23:50 ..
-rw-r--r--  1 root root 10240 lip 31 23:56 arch.tar
-rw-r--r--  1 root root     0 lip 31 23:53 plik1.txt
-rw-r--r--  1 root root     0 lip 31 23:53 plik2.txt
-rw-r--r--  1 root root     0 lip 31 23:53 plik6.txt
-rw-r--r--  1 root root     0 lip 31 23:53 plik8.txt
-rw-r--r--  1 root root     0 lip 31 23:53 plik.txt
drwxr-xr-x  2 root root  4096 lip 31 23:54 rozpakowane
root@sylwia-Kubuntu:~/PLIKI# cp arch.tar rozpakowane/
root@sylwia-Kubuntu:~/PLIKI# cd rozpakowane/
root@sylwia-Kubuntu:~/PLIKI/rozpakowane# tar -xvf arch.tar
plik1.txt
plik2.txt
plik8.txt
root@sylwia-Kubuntu:~/PLIKI/rozpakowane# ls -la
razem 20
drwxr-xr-x 2 root root  4096 lip 31 23:57 .
drwxr-xr-x 3 root root  4096 lip 31 23:56 ..
-rw-r--r-- 1 root root 10240 lip 31 23:56 arch.tar
-rw-r--r-- 1 root root     0 lip 31 23:53 plik1.txt
-rw-r--r-- 1 root root     0 lip 31 23:53 plik2.txt
-rw-r--r-- 1 root root     0 lip 31 23:53 plik8.txt
root@sylwia-Kubuntu:~/PLIKI/rozpakowane#
```

Rys. 17.1. Polecenia obsługujące archiwizacje i wyodrębnianie zbiorów

PRZYKŁAD 17.5

Tworzenie archiwów zawierających zbiory

Aby utworzyć archiwum zawierające strukturę zbiorów utworzonych w przykładzie 15.1, wykonaj czynności przedstawione poniżej.

1. Zaloguj się na konto użytkownika.
2. W katalogu domowym wpisz polecenie

```
sudo  tar -cvf arch.tar styczen/
```

Polecenie to utworzy archiwum zawierające wszystkie zbiory z katalogu **styczen**. Podczas przetwarzania katalogu będą wyświetlane nazwy aktualnie przetwarzanego zbioru.

3. Utwórz katalog, w którym archiwum zostanie rozpakowane za pomocą polecenia

```
sudo  mkdir  rozpakowane
```

4. Skopiuj plik arch.tar do katalogu rozpakowane za pomocą polecenia

```
sudo  cp arch.tar rozpakowane/arch.tar
```

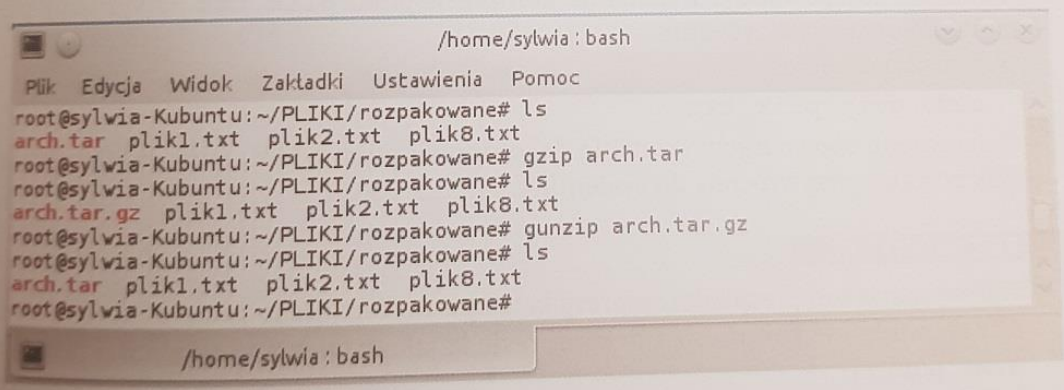
5. Zmień katalog na rozpakowane i wpisz polecenie

```
sudo  tar –xvf arch.tar
```

6. Wyświetl drzewo zbiorów za pomocą polecenia **tree**.

17.4. Kompresja zbiorów systemu Linux

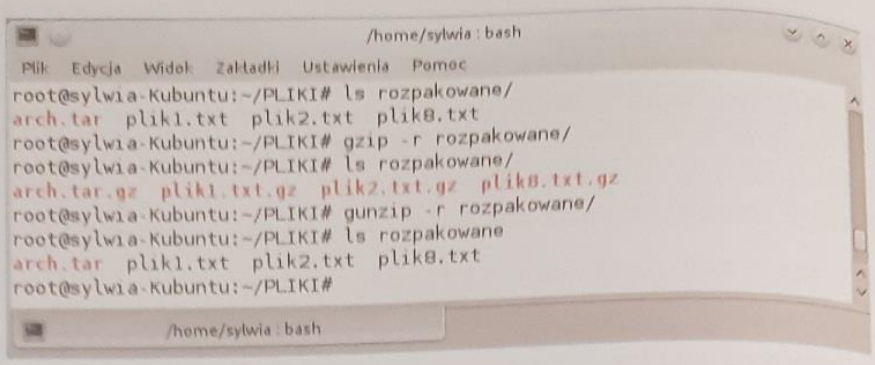
Istnieje wiele programów wykonujących kompresję zbiorów w Linuksie. Najczęściej używany jest program **gzip**. Przykład użycia programu **gzip** do kompresji pliku pokazano na rys. 17.2.



Rys. 17.2. Kompresja i dekompresja pliku

Warto zwrócić uwagę, że podczas kompresji plik oryginalny jest usuwany z systemu. Skompresowany plik archiwum otrzymał nazwę **arch.tar.gz** (czasami stosuje się skrócony zapis **tgz**). W takiej postaci jest rozprowadzanych wiele programów przeznaczonych dla Linuksa.

Kompresji można poddać pliki lub całe katalogi wraz z podkatalogami. Podczas kompresji katalogu każdy plik jest kompresowany oddzielnie (rys. 17.3).



Rys. 17.3. Kompresja i dekompresja katalogu

W Linuksie **tar** może dokonać dodatkowej kompresji utworzonego archiwum, korzystając z programu **gzip**, w takiej sytuacji należy użyć dodatkowej opcji **-z**. Przykład użycia polecenia tworzącego skompresowane archiwum za pomocą programu **tar**:

```
sudo tar -czvf arch.tgz katalog
```

Do dekompresji należy użyć polecenia:

```
sudo tar -xzvf arch.tgz rozpakowane
```

PRZYKŁAD 17.6

Wykonywanie kompresji i dekompresji pliku

Aby wykonać kompresję i dekompresję pliku, wykonaj czynności przedstawione poniżej.

1. Zaloguj się na konto użytkownika.
2. Wpisz polecenie:

```
sudo gzip plik.txt
```

Polecenie to spowoduje wykonanie kompresji pliku **plik.txt**. Oryginalny plik zostanie usunięty z systemu.

3. Wpisz polecenie:

```
sudo gzip plik.txt
```

Polecenie to spowoduje wykonanie dekompresji pliku **plik.txt.gz**. Oryginalny plik zostanie przywrócony do systemu.

PRZYKŁAD 17.7

Wykonywanie kompresji i dekompresji katalogu zawierającego pliki

Aby spowodować kompresję i dekompresję katalogu, wykonaj czynności przedstawione poniżej.

1. Zaloguj się na konto użytkownika.
2. Wpisz polecenie:

```
sudo gzip -r styczen
```

Polecenie to spowoduje kompresję katalogu **styczen**, utworzonego w ćwiczeniu 15.1. Każdy plik w katalogu i jego podkatalogach zostanie poddany kompresji niezależnie od pozostałych zbiorów.

3. Wpisz polecenie:

```
sudo gunzip -r styczen
```

Polecenie to spowoduje dekompresję plików w katalogu **styczen** i jego podkatalogach.

17.5. Publikacje elektroniczne dotyczące systemu Linux

Publikacje elektroniczne to nie tylko prezentacje multimedialne czy demonstracje. Coraz więcej firm i osób publikuje własne katalogi, biuletyny czy skrypty. Niski koszt produkcji i dystrybucji wydawnictw elektronicznych sprawił, że są dostępne dla każdego. Dla wielu twórców pojawiła się możliwość pokazania własnej pracy w atrakcyjny sposób i to od razu na bardzo dużą skalę.

Do wyróżniających się twórców w kwestii publikacji elektronicznych należy zaliczyć wszystkich tych, którzy zajmują się pisaniem skryptów i opracowań dla systemu Linux. Ogromną bazę wiedzy zamieszczono na stronie `https://www.linux.pl/`, na ktorej znajdziemy publikacje oraz wersje elektroniczne pozycji książkowych, obszerną dokumentację Linuksa z podziałem na kategorie i podręczniki najpopularniejszych dystrybucji.

Również sam system jest wyposażony w obszerną pomoc systemową udostępnianą w postaci podręcznika.

Podobnie jak dla systemu Windows, na stronie `https://www.dobreprogramy.pl` możemy wyszukać ciekawe demonstracje dotyczące konfiguracji i pracy w systemie Linux.

Liczne informacje uzyskamy również na stronie oraz `https://linux-magazine.pl//`

Tam też znajdziemy wiele porad praktycznych i sprawdzonych rozwiązań w pracy z systemem.

SPRAWDŹ SWOJĄ WIEDZĘ

1. Scharakteryzuj edytor tekstu vi.
2. Co to są skrypty powłoki?
3. Na czym polega archiwizacja zbiorów systemu Linux?
4. Jak kompensuje się zbiory w systemie Linux?