



Mateusz Lach

BASH

PRAKTYCZNE SKRYPTY

Wykorzystaj Bash do swoich celów!

- Jak działa Bash, czyli do czego potrzebna Ci ta powłoka i jak ją inteligentnie wykorzystać
- Raport wydajnościowy, czyli jak szybko znaleźć pliki, które ktoś ostatnio zmodyfikował
- W małym kinie, czyli jak zaprojektować system rezerwacji miejsc

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Redaktor prowadzący: Michał Mrowiec

Projekt okładki: Studio Gravite / Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki

Fotografia na okładce została wykorzystana za zgodą Shutterstock.com

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<http://helion.pl/user/opinie/bashps>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Kody źródłowe wybranych przykładów dostępne są pod adresem:

<ftp://ftp.helion.pl/przyklady/bashps.zip>

ISBN: 978-83-283-1489-4

Copyright © Helion 2015

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

Rozdział 1. Wprowadzenie	9
Jak działa Bash?	10
Założenia tej książki	11
Układ rozdziałów	11
Konwencje programistyczne	12
Instalacja i konfiguracja środowiska pracy	12
OpenSUSE 13.2	12
Windows 7	14
Rozdział 2. Powitanie	19
Zmienne i ich wartości	25
Znak zachęty	27
Polecenie echo — wyświetlenie tekstu na ekranie	30
Stringi i ich łączenie	30
Polecenie cd — bieżąca ścieżka i jej zmiana	33
Ustalanie bieżącej ścieżki, czyli krótka wzmianka o zmiennej \$PWD	33
Zmiana bieżącej ścieżki	33
Polecenie chmod — zmiana uprawnień plików i katalogów	33
Tryb interaktywny/konwersacyjny i wsadowy	36
Polecenie read — pobranie danych od użytkownika	37
Rozdział 3. Uścisk dłoni	39
Rozdzielenie instrukcji i poleceń	42
Średnik i przetwarzanie sekwencyjne	42
Ampersand i przetwarzanie równoległe	43
Instrukcja warunkowa if	44
Operator porównania stringów (i nie tylko ich)	49
Wartości logiczne true/false, czyli brak typu w Bashu	50
Nawiasy kwadratowe w instrukcjach warunkowych	51
Brak nawiasów	51
Pojedyncze nawiasy	56
Podwójne nawiasy kwadratowe	58
Polecenie type — rozpoznawanie typu	60

Rozdział 4. Rozpoznanie płci	63
Parametry wejściowe	68
Operacje na stringach i notacja klamrowa	71
Konkatenacja a zmienne specjalne	72
Dzielenie ciągów znakowych na mniejsze	73
Podwójne rozwinięcie zmiennej	74
Operatory logiczne	75
Alternatywa	75
Negacja	76
Koniunkcja	77
Notacja z pojedynczymi nawiasami kwadratowymi	78
Poprawka drobnego błędu	80
Rozdział 5. Detektor liczb parzystych i nieparzystych	85
Zmienne liczbowe	87
Wyrażenia matematyczne w notacji z okrągłymi nawiasami	88
Wyrażenia matematyczne z poleceniem let	91
Wyrażenia matematyczne z poleceniem expr	92
Polecenie declare — oznaczenie typu dla zmiennej	93
Rozdział 6. Generowanie zbiorów liczb spełniających kryteria podzielności	97
Pętla while	99
Wyrażenia matematyczne w instrukcjach sterujących if oraz pętli while	101
Rozdział 7. Silnia	103
Pętla for	104
Zapis arytmetyczny	104
i++ oraz i--, czyli skrócony zapis inkrementacji/dekrementacji	107
Rozdział 8. Analiza zbioru plików pod kątem daty ich ostatniej modyfikacji	109
Pętla for — zapis operujący na zbiorach	112
Operatory logiczne poza instrukcją sterującą if	113
Wyrażenia Basha	116
Notacja \$()	116
Notacja z grawisami	117
Polecenie eval	118
Uwaga, niebezpieczeństwo!	119
Polecenie stat	121
Polecenie date	122
Konwersja z formatu czytelnego dla człowieka do czasu uniksowego	123
Konwersja z czasu uniksowego do formatu czytelnego dla człowieka	124
Rozdział 9. Czyszczenie pliku tekstowego ze zbędnych pustych wierszy	127
Operatory jednoargumentowe polecenia test w blokach warunkowych	130
Instrukcja continue	131
Polecenie cat	133
Strumienie i ich przekierowanie	135
Przekierowanie strumienia wyjścia do pliku	136
Przekierowanie strumienia błędów do pliku	137
Przekierowanie strumienia wyjściowego i błędów do różnych plików	137
Przekierowanie obu strumieni wyjściowych do tego samego pliku	138
Przekierowanie strumienia błędów do strumienia wyjścia i odwrotnie	139
Przekierowanie na strumień wejściowy	139

Rozdział 10. Sortowanie liczb	145
Pętla w pętli	149
Pobranie wszystkich parametrów wejściowych programu	151
Zmienne tablicowe	151
Inicjalizacja tablicy	152
Zapisywanie wartości w tablicach	153
Pobieranie wartości z tablic	154
Tablice asocjacyjne, czyli klucze tekstowe zamiast indeksów liczbowych	155
Wypisanie wszystkich wartości tablicy	156
Pobranie ilości elementów znajdujących się w tablicy	157
Przetworzenie każdego elementu tablicy w pętli	157
Rozdział 11. System rezerwacji miejsc w sali kinowej	161
Instrukcja break	164
Blok warunkowy case jako alternatywa dla if	166
Dopasowanie do wielu wartości w bloku case	168
Eskalacja wykonania na następny blok w case	169
Funkcje	170
Deklaracja funkcji	173
Wywoływanie funkcji	174
Funkcje mają priorytet	174
Funkcje mają pierwszeństwo	175
Funkcje mogą się nadpisywać	176
Parametry wejściowe	177
Zmienne w funkcjach	178
Wywołanie funkcji w wyrażeniu \$() Basha	183
Zwracanie i pobieranie wyników funkcji	185
Hermetyzacja funkcji	187
Przekazywanie tablic w parametrach do funkcji	192
Polecenie source	196
Rozdział 12. Sprawdzanie poprawności konfiguracji sieci komputerowej	199
Wprowadzenie teoretyczne	202
Systemy liczbowe	202
Sieć komputerowa	206
Stałe, czyli zmienne, których nie da się zmienić	207
Polecenie readonly	208
Polecenie declare	208
Operatory bitowe	208
Przesunięcia bitowe w lewo	209
Przesunięcia bitowe w prawo	210
Suma bitowa	211
Iloczyn bitowy	212
Notacja klamrowa — podmiana zawartości ciągu znakowego na inną	213
Rozdział 13. Ciągi Fibonacciego	215
Funkcje rekurencyjne w programowaniu	218
Wydajność funkcji rekurencyjnych	218
Brak ograniczenia w ilości wywołanych funkcji rekurencyjnych	219
Wydajność rekurencyjnych funkcji w ciągach Fibonacciego	219
Wykładnicza złożoność obliczeniowa	220
Liniowa złożoność obliczeniowa	222
Polecenie time	224

Rozdział 14. Prosty kalkulator ze wsparciem dla arytmetyki zmiennoprzecinkowej	229
Pętla until	233
Potoki	234
Polecenie printf zamiast echo	236
Liczby w wyniku	237
Konwersja pomiędzy systemami liczbowymi	237
Ciągi znaków w wyniku	239
Modyfikatory	239
Program bc	241
Arytmetyka zmiennoprzecinkowa	241
Precyzja liczb zmiennoprzecinkowych	242
Konwersja pomiędzy systemami liczbowymi	242
Rozdział 15. System rezerwacji miejsc w multipleksie kinowym	245
Tablice wielowymiarowe	248
Pętla select	248
Rozdział 16. Rekurencyjne wyszukiwanie dat w plikach	251
Polecenie grep	254
Wyrażenia regularne	255
Znaki specjalne	255
Alternatywa we wzorcach	255
Atomy	257
Zakresy znaków	258
Wielokrotne i opcjonalne dopasowania	260
Rozszerzenia perlowe	263
Podsumowanie	266
Polecenie sort	266
Tu masz string, czyli wyrażenie <<<	268
Rozdział 17. Rozwiązania ćwiczeń	271
Ćwiczenie 2.1	271
Ćwiczenie 2.2	272
Ćwiczenie 3.1	272
Ćwiczenie 4.1	273
Ćwiczenie 4.2	273
Ćwiczenie 5.1	274
Ćwiczenie 6.1	274
Ćwiczenie 6.2	275
Ćwiczenie 7.1	276
Ćwiczenie 8.1	276
Ćwiczenie 9.1	276
Ćwiczenie 10.1	277
Ćwiczenie 10.2	277
Ćwiczenie 10.3	278
Ćwiczenie 11.1	278
Ćwiczenie 12.1	279
Ćwiczenie 12.2	280
Ćwiczenie 13.1	282
Ćwiczenie 13.2	282
Ćwiczenie 13.3	283
Ćwiczenie 14.1	283
Ćwiczenie 15.1	284
Ćwiczenie 16.1	286
Skorowidz	287

Rozdział 7.

Silnia

Od kilku rozdziałów omawiamy kolejne elementy Basha na podstawie przykładów matematycznych. Może się to wydawać trochę nudne w porównaniu do naszych pierwszych lekcji, które dotyczyły programowania na przykładzie rad dotyczących dobrego zachowania. Zapewniam jednak, że odrobina matematyki jeszcze nikomu nie zaszkodziła, a pozwala ona tworzyć dobrane pod względem dydaktycznym przykłady.

Na potrzeby tego rozdziału uczynię mały wyjątek i zaczniemy od małego wprowadzenia teoretycznego, ponieważ nie każdy Czytelnik musi wiedzieć lub pamiętać, czym jest silnia. Jest ona prostym działaniem matematycznym i zapisuje się ją w postaci $n!$, gdzie n to dowolna liczba naturalna. Działanie to polega na obliczeniu iloczynu kolejnych liczb naturalnych, które są większe od 0 i nie większe od n , natomiast silnia z 0 z definicji wynosi 1, czyli:

$$0! = 1$$

$$1! = 1$$

$$2! = 1 \cdot 2 = 2$$

$$3! = 1 \cdot 2 \cdot 3 = 6$$

$$4! = 1 \cdot 2 \cdot 3 \cdot 4 = 24$$

$$5! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 = 120$$

Przykład realizacji tego zadania w Bashu przedstawiam w listingu 7.1.

Listing 7.1. *Silnia* — przykład realizacji w Bashu

```
#!/usr/bin/bash

i=1 #nie-zero bo tworzy iloczyn
wynik=1 #nie-zero bo tworzy iloczyn
while ((i <= $1))
do
    wynik=$((wynik * i))
    i=$((i + 1))
done
echo $wynik
```

W skrypcie z listingu 7.1 przedstawiam realizację zadania matematycznego przy użyciu jedynie znanych nam już mechanizmów i technik. W listingu 7.2 natomiast zostało wykonane to samo zadanie z zastosowaniem nieomówionej jeszcze notacji.

Listing 7.2. *Silnia — przykład realizacji z użyciem pętli for w języku Bash*

```
#!/usr/bin/bash

wynik=1 # nie-zero bo tworzy iloczyn
for ((i=1; $i <= $1; i=$((i+1)))
do
    wynik=$((wynik * $i))
done
echo $wynikecho $wynik
```

Listing 7.2 przedstawia alternatywną wersję skryptu z listingu 7.1. Spis nowości zawiera tabela 7.1.

Tabela 7.1. *Nowe terminy z listingu 7.2*

Termin	Opis
for ((; ;))	For to słowo kluczowe powłoki Bash, odpowiednik <code>while</code> . Działa dokładnie tak samo, różni się jednak zapisem, gdyż jego warunek jest w tym wypadku wyrażenia matematycznego podzielony na trzy sekcje, które zostaną omówione w części teoretycznej. Możliwe jest jeszcze łączenie pętli <code>for</code> ze zbiórami. To również omówię w części teoretycznej.
do	Słowo kluczowe powłoki Bash. Rozpoczyna pętlę <code>for</code> , tak samo jak <code>while</code> .
done	Słowo kluczowe powłoki Bash. Kończy pętlę <code>for</code> , tak samo jak <code>while</code> .
i++	Skrócony zapis dla zwiększenia licznika zmiennej o 1. Zostanie omówiony bardziej szczegółowo w rozdziale teoretycznym.

Schemat blokowy dla skryptu z listingu 7.2 przedstawiam na rysunku 7.1.

Pętla for

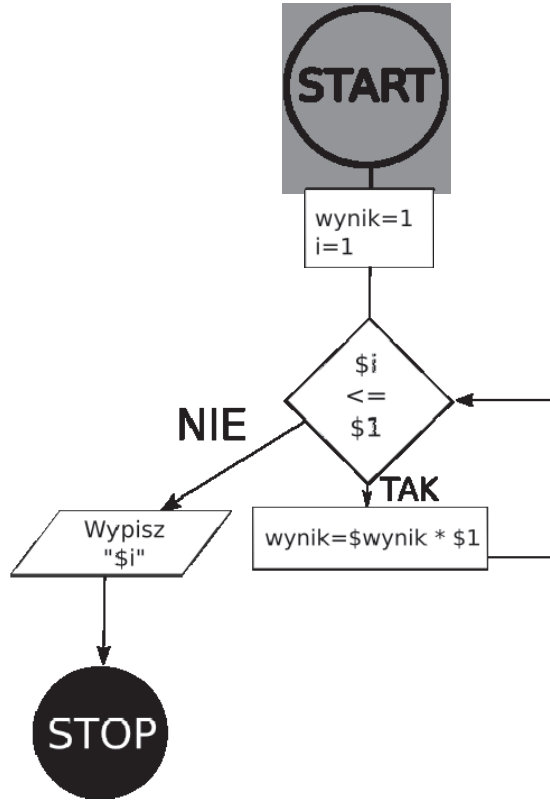
W Bashu istnieje kilka rodzajów instrukcji sterujących, które oferują wielokrotne wykonanie bloku kodu (tzw. pętle). Jedną z nich jest pętla `for`, która poza skróconym zapisem nie różni się niczym w porównaniu do pętli `while`.

Zapis arytmetyczny

Pętla `for` obsługuje kilka możliwości działania. Jedną z nich jest notacja wykorzystująca wyrażenia arytmetyczne. Ogólny zapis pętli w postaci pseudokodu zamieszczam poniżej:

```
dla (( instrukcje_startowe; warunek_wykonania_pętli; instrukcje_po_każdym_przebiegu ))
    wykonaj
    # blok pętli, który powinien zostać wykonany, jeśli warunek wykonania jest spełniony
koniec
```


Rysunek 7.1.
Schemat blokowy
dla skryptu
z listingu 7.2



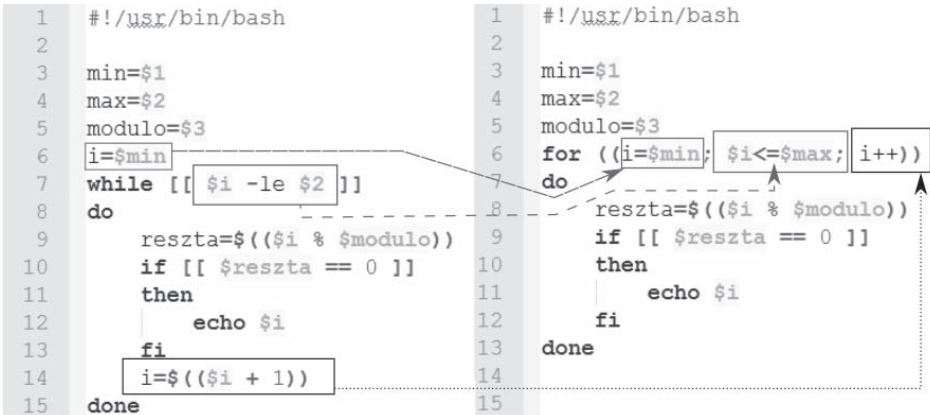
Na rysunku 7.2. znajduje się analiza porównawcza pętli `while` i `for`, której dokonałem na przykładzie skryptu z listingu 6.1 i stanowiącego jego alternatywną wersję skryptu z listingu 7.3.

Listing 7.3. Skrypt z listingu 6.1 przepisany na wersję z użyciem pętli `for`

```

#!/usr/bin/bash

min=$1
max=$2
modulo=3
i=$min
for ((i=$min; i<=$max; i++))
do
    reszta=$((i % modulo))
    if [[ $reszta == 0 ]]
    then
        echo $i
    fi
done
  
```



Rysunek 7.2. Analiza porównawcza skryptu z listingu 6.1 (po lewej) i skryptu z listingu 7.3 (po prawej). Dotyczy ona różnic występujących między instrukcjami sterującymi `for` i `while`

Na rysunku 7.2 widać różnice w zapisie między obiema pętlami. W mojej ocenie pętla `while` jest dużo bardziej czytelna dla początkujących programistów i spełnia w zasadzie wszystkie wymagania, realizując te same zadania co pętla `for` w zapisie arytmetycznym, jednakże w inny sposób. Jediną zaletą, która przemawia za pętlą `for`, jest liczba linii kodu potrzebnych do realizacji tego samego zadania. Muszę jednak przyznać, że z biegiem czasu coraz częściej korzystam z pętli `for` zamiast `while`, choć nie potrafię logicznie uzasadnić tego wyboru.

Warto jeszcze wspomnieć, że instrukcje startowe, warunkowe i końcowe przebiegu pętli `for` są instrukcjami opcjonalnymi. Oznacza to, że nie musimy ich wpisywać. Musimy jedynie pamiętać o oddzielających je średnikach. Może to posłużyć do tworzenia nieskończonych pętli, co przedstawiam w listingu 7.4.

Listing 7.4. Nieskończona pętla z użyciem pętli `for`

```

#!/usr/bin/bash

for ( (: ) )
do
    echo "Wykonuję się dopóty, dopóki mnie nie zamkniesz"
done

```

Z pozoru taka instrukcja jest nieprzydatna, ponieważ do momentu, kiedy sami nie zamknijemy programu, na przykład używając skrótu `CTRL+C`, program nie zakończy działania. Wbrew pierwszemu wrażeniu ma to swoje zastosowanie, ale o tym opowiem w kolejnych rozdziałach.

Cały zapis pętli `for` w wersji arytmetycznej (poza częścią będącą samym wyrażeniem) złożony jest ze słów kluczowych:

```

$ type -a for do done "(" ")"
for jest słowem kluczowym powłoki
do jest słowem kluczowym powłoki
done jest słowem kluczowym powłoki

```

```
-bash: type: ((: nie znaleziono
-bash: type: )): nie znaleziono
$
```

i++ oraz i--, czyli skrócony zapis inkrementacji/dekrementacji

W Bashu, tak jak w innych popularnych językach programowania, dostępny jest skrócony operator zwiększenia wartości zmiennej (inkrementacji; ang. *incrementation*) oraz zmniejszenia (dekrementacji; ang. *decrementation*). Ich zapis wygląda następująco:

i++ to zwiększenie wartości zmiennej \$i o 1, skrócony zapis dla $i=i+1$

i-- to zmniejszenie wartości zmiennej \$i o 1, skrócony zapis dla $i=i-1$

W odróżnieniu od innych języków programowania, w Bashu nie można używać tych zapisów tak po prostu. Można je stosować jedynie w notacji matematycznej, co zaprezentowałem w poniższym przykładzie:

```
$ a=4
$ a++
-bash: a++: nie znaleziono polecenia
$ declare -i b=4
$ b++ #nie działa również dla zmiennej typu integer
-bash: b++: nie znaleziono polecenia
$
```

Zapewne zastanawiasz się, gdzie się podział ulubiony znak dolara, który powinien poprzedzać odwołanie do zmiennej. O tym, że jest on w tym wypadku zbędny, świadczy ten oto przykład:

```
$ c=4
$ $c++
-bash: 4++: nie znaleziono polecenia
$
```

Bash zgłasza błąd, że polecenie 4++ nie istnieje. Dzieje się tak dlatego, że Bash nie zna tego zapisu i wpiery próbuje rozwinąć zmienną jej wartością, a następnie wykonać polecenie, które powstało z wartości zmiennej i doklejonego napisu ++. Spróbujmy udowodnić tę teorię:

```
$ a="echo "
$ a++
-bash: a++: nie znaleziono polecenia
$ $a++ #Bash wykonał polecenie echo i wypisał dwa plusy na ekran
++
$
```

Jak widać na powyższym przykładzie, Bash rozwinął zmienną i wykonał polecenie, które powstało z wyniku połączenia obu stringów. To właśnie dlatego na ekranie pojawiły się dwa plusy.

Używanie skróconego zapisu w Bashu jest możliwe jedynie w parze z wyrażeniami matematycznymi:

```
$ i=1
$ ((i++))
$ echo $i
2
$
```

Jednak również w tym przypadku nie wstawia się dolara przed nazwą zmiennej, gdyż efekt będzie taki sam jak w zapisie bez notacji arytmetycznej. Aby zrozumieć, dlaczego tak się dzieje, trzeba zrozumieć, jak działa interpreter. Najpierw podmieniane są wartości zmiennych, a następnie Bash próbuje wykonać polecenie. A oto inny dowód na tę teorię:

```
$ polecenie="echo Interpreter "
$ $polecenie"Bash"
Interpreter Bash
$
```

Innym wytłumaczeniem powodu, dla którego nie wstawiamy dolara, jest fakt, że zmieniamy wartość zmiennej numerycznej, zwiększając lub zmniejszając ją o 1. W pełnym zapisie wartość do zmiennej przypisujemy również bez znaku dolara, ponieważ ten stosujemy jedynie w celu odwołania do wartości zmiennej, np. `i=4` lub `i=$((i+2))`.

Ćwiczenie 7.1

Napisz rozwiązanie do listingu 6.1, używając pętli `for` zamiast `while`.



Uwaga

Propozycje rozwiązań znajdują się na końcu książki, w ostatnim rozdziale, zatytułowanym „Rozwiązania ćwiczeń”.

Skorowidz

A

adres IPv4, 201, 206
algorytm sortowania, 149
alternatywa, 75, 81
 we wzorcach, 255
ampersand, 43
analiza
 działania potoków, 235
 parametrów wejściowych, 69
 porównawcza skryptu, 106
 przekazywania tablic, 194
 zbioru plików, 109, 110
arytmetyka zmiennoprzecinkowa, 229, 241
asercje o zerowej długości, 264, 265
atomy, 257

B

backslash, 27, 32
Bash, 10
bezpieczeństwo, 119
biblioteka funkcji dla skryptu, 246
bit wykonywalności, 23
blok
 case, 166, 167
 else – fi, 46
 warunkowy, 130
błąd, 80
 składni, 53
błędna lista parametrów, 188
brak nawiasów, 51

C

ciąg Fibonacciego, 215
ciągi
 znakowe, 73
 znaków w wyniku, 239

cyfra, 86
czas uniksowy, 123
czyszczenie pliku tekstowego, 127
czytanie HEREDOC, 142

D

data modyfikacji pliku, 109
deklaracja funkcji, 173
dekrementacja, 107
detektor
 liczb, 85
 parzystości, 86
dodawanie, 88
 uprawnień, 35
dokumenty
 HEREDOC, 142
 inline, 140
domyślny prompt, 27
dopasowania, 256
 opcjonalne, 260
 wielokrotne, 260
dopasowanie
 w bloku case, 168, 169
dostęp do
 odczytu, 34
 uruchamiania, 34
 zapisu, 34
dynamiczne typowanie, 50
działanie
 bloku case, 167
 instrukcji break, 165
 polecenia return, 186
 potoków, 235
dzielenie, 89
 ciągów znakowych, 73

E

edytor Notepad++, 22, 142
 elementy
 schematów blokowych, 41
 tablicy, 157
 eskalacja wykonania, 169

F

formatery
 liczbowe, 238
 polecenia date, 125
 polecenia stat, 123
 tekstowe, 240
 funkcje, 164, 170, 174
 hermetyzacja, 187
 nadpisywanie, 176
 nadpisywanie zmiennych lokalnych, 182
 parametry wejściowe, 177, 188
 pierwszeństwo, 175
 pobieranie wyników, 185
 priorytet, 174
 przekazywanie tablic, 192, 194
 rekurencyjne, 218
 użycie zmiennych, 189
 widoczność zmiennych lokalnych, 181
 wywoływanie, 174, 183
 zmiennie globalne, 178
 zwracanie wartości, 185, 190

G

gadatliwa wersja skryptu, 221
 generowanie zbiorów liczb, 97
 grawis, 117
 grupy znakowe, 262, 263

H

HEREDOC, 140
 hermetyzacja funkcji, 187

I

identyfikator procesu, 184
 iloczyn
 bitowy, 212
 logiczny, 77
 ilość
 elementów dokładna, 261
 elementów dowolna, 261
 wywołanych funkcji rekurencyjnych, 219

implementacja
 ciągów Fibonacciego, 222
 iteracyjna, 226
 rekurencyjna, 226
 indeks liczbowy, 155
 inicjalizacja
 tablicy, 152
 zmiennej, 20
 inkrementacja, 100, 107
 instalacja
 programu Cygwin, 15
 środowiska pracy
 OpenSUSE 13.2, 12
 Windows 7, 14
 instalator pakietów, 230
 instrukcja
 break, 163, 164
 case, 163
 continue, 129, 131, 163
 if, 39, 44, 51, 101, 113
 instrukcje sterujące, 39
 interfejs graficzny KDE, 13
 interpretacja, 10
 interpreter, 10

J

języki słabo typowane, 50

K

kalkulator z menu, 229, 230
 kierunek wykonywania skryptów, 23
 klucze tekstowe, 155
 kod wyjścia, 55
 komentarz, 24
 komunikacja, 185
 konfiguracja
 sieci, 199
 środowiska pracy
 OpenSUSE 13.2, 12
 Windows 7, 14
 koniec pliku, 129
 koniunkcja, 77
 konkatenacja, 31, 72
 konsola, 10
 konwersja, 50
 liczby dziesiętnej, 205
 niejawna, 50
 systemów liczbowych, 237, 242, 244
 z czasu uniksowego, 124
 z formatu czytelnego dla człowieka, 123
 znaku końca linii, 23
 kryteria podzielności, 97

L

liczba
 dwójkowa, 204
 nieparzysta, 85
 parzysta, 85
 liczby w wyniku, 237
 logi, 252
 lookingahead, 264
 lookingbehind, 264

Ł

łączenie stringów, 30, 31

M

maski sieci, 207
 mnożenie, 89
 modulo, 90
 modyfikatory, 239

N

nadpisywanie zmiennych lokalnych, 182
 nawiasy
 klamrowe, 261
 kwadratowe, 40, 51
 podwójne, 58
 pojedyncze, 78
 okrągłe, 88
 pojedyncze, 56
 negacja, 76
 nieskończona pętla, 163
 notacja
 \$(), 116
 \${}, 202
 klamrowa, 71–74, 213
 liczbowa, 35
 literowa, 35
 z grawisami, 117
 z okrągłymi nawiasami, 88
 z podwójnymi nawiasami, 58, 59
 z pojedynczymi nawiasami kwadratowymi, 78
 notacje instrukcji warunkowych, 52
 Notepad++, 142

O

obliczanie
 elementów ciągu, 216
 potęgi, 171

obsługa menu, 245
 odejmowanie, 88
 okno terminala, 10
 opcje polecenia chmod, 36
 opcjonalne wystąpienie elementu, 262
 operacje
 koniunkcji, 77
 matematyczne, 86
 na stringach, 71, 73
 na tablicach, 158
 operator
 ||, 66, 75
 -ge, 146
 -le, 99
 koniunkcji, 77, 79
 mniejszości, 146
 mniejszy-równy, 99
 modulo, 86
 negacji, 76
 porównania, 49
 przypisania, 21
 operatory
 bitowe, 208
 do przetwarzania tekstu, 74
 jednoargumentowe, 130
 leniwe, 114
 logiczne, 66, 75, 80, 113
 oznaczenie typu, 93

P

pakiety środowiska Cygwin, 230
 parametry
 polecenia
 bc, 241
 cat, 134
 date, 124
 echo, 30
 grep, 254
 read, 38, 140
 stat, 122
 type, 60
 wejściowe, 68, 177
 funkcji, 188
 programu, 151
 pętla, 232
 for, 104, 112
 for operująca na zbiorach, 113
 nieskończona, 106, 163
 select, 28, 248
 until, 233
 w pętli, 149
 while, 99, 101, 233

- pobieranie
 - parametrów wejściowych, 151
 - wartości z tablic, 146, 154
 - podmiana zawartości ciągu znakowego, 213
 - podpowłoki, 184
 - podwójne rozwinięcie zmiennej, 74
 - polecenia w HEREDOC, 141
 - polecenie
 - cat, 128, 133
 - cd, 22, 24, 33
 - chmod, 24, 33, 35
 - date, 111, 122
 - declare, 93, 208
 - echo, 20, 24, 30, 129
 - eval, 118
 - exit, 55
 - expr, 92
 - grep, 254
 - let, 91
 - local, 180
 - printf, 236, 238, 240
 - read, 24, 37, 38, 140
 - readonly, 208
 - return, 186
 - source, 163, 196
 - stat, 111, 121
 - test, 53, 78, 129
 - time, 224
 - type, 60
 - typeset, 94
 - porównanie
 - operatorów logicznych, 80
 - wywołania funkcji, 183
 - potęgowanie, 90
 - potoki, 234
 - precyzja
 - alternatywy, 258
 - liczb zmiennoprzecinkowych, 242
 - priorytety w zmiennych specjalnych, 72
 - program
 - CMD, 14
 - Cygwin, 14
 - PowerShell, 14
 - prompt, 20, 27
 - przekazywanie
 - tablic
 - przez nazwę, 194
 - przez wartości, 192
 - przekierowanie
 - na strumień wejściowy, 139
 - obu strumieni wyjściowych, 138
 - strumienia, 135, 232
 - błędów, 137, 139
 - do pliku, 136–138
 - wyjścia, 136–139
 - przeliczanie liczb
 - binarnych, 204
 - dziesiętnych, 205
 - przełączanie interpretacji wyrażeń regularnych, 254
 - przesunięcia bitowe
 - w lewo, 209
 - w prawo, 210
 - przetwarzanie
 - równoległe, 43
 - sekwencyjne, 42
 - przypisanie do zmiennej, 86
 - puste wiersze, 127
- ## R
- rekurencyjne wyszukiwanie dat, 251
 - reszta z dzielenia, 90
 - rozdzielenie instrukcji i poleceń, 42
 - rozpoznawanie typu, 60
 - rozszerzenia perlowe, 263
 - rozwiązania ćwiczeń, 271
 - rzutowanie, 50
- ## S
- schemat blokowy, 41, 42, 45, 47, 68, 98, 105, 112, 133, 165, 217, 253
 - serwer DHCP, 199
 - sieć komputerowa, 206
 - silnia, 103
 - składnia
 - \$(), 111
 - HEREDOC, 141
 - skrótowa forma instrukcji warunkowej, 111
 - skrypt rozpoznający plęć, 63, 66
 - słowo kluczowe
 - do, 99
 - done, 99
 - else, 40
 - fi, 40
 - if, 40
 - then, 40
 - while, 99
 - sortowanie liczb, 145
 - sprawdzanie
 - poprawności konfiguracji sieci, 199
 - zgodności adresów, 201
 - stałe, 207
 - standardowe wyjście, 20
 - stringi, 30
 - w instrukcjach warunkowych, 56

strumień, 135
 błędów, stderr, 136
 wejścia, stdin, 135
 wyjścia, stdout, 20, 135
 wyjścia podpowłoki, 186
 suma bitowa, 202, 211
 system
 liczbowy, 202
 binarny, 204
 dziesiętny, 203, 237
 ósemkowy, 237
 szesnastkowy, 237
 operacyjny
 OpenSUSE, 12
 Windows 7, 14
 rezerwacji miejsc, 161, 245

Ś

ścieżka, 22, 24, 33
 średnik, 40, 42

T

tablice, 152
 asocjacyjne, 155
 ilość elementów, 157
 indeks, 155
 inicjalizacja, 152
 pobieranie wartości, 154
 przetworzenie każdego elementu, 157
 wielowymiarowe, 248
 wypisanie wszystkich wartości, 156
 zapisywanie wartości, 153
 termin
 \$, 20
 read, 24
 terminal, 10
 tryb
 gadatliwy, 220
 interaktywny, 10, 36, 196
 odpluskwiania kodu, 28
 tylko do odczytu, 202
 wsadowy, 10, 36, 196
 tylda, 22

U

uprawnienia, 33–35
 nieefektywne, 34
 uruchamianie skryptu, 21
 usuwanie pustych wierszy, 127
 użycie potoków, 244

W

walidacja, 120
 wartości logiczne, 50
 wartość zmiennej, 25, 37
 wczytywanie dokumentów HEREDOC, 141
 wersja gadatliwa, 220
 weryfikacja zawartości pliku tekstowego, 128
 widoczność zmiennych lokalnych, 181
 właściciel pliku, 33
 wrapper, 56
 wstrzyknięcie kodu, 120
 wydajność funkcji rekurencyjnych, 218
 wykonanie podpowłoki, 185
 wykrycie trybu pracy, 36
 wypisywanie
 ciągu znaków, 225
 wartości tablicy, 156
 wyrażenia
 Basha, 116
 matematyczne, 88
 w instrukcjach sterujących, 101
 w pętli while, 101
 z poleceniem expr, 92
 z poleceniem let, 91
 regularne, 253, 255
 alternatywa, 255
 atomy, 257
 grupy znakowe, 262
 rozszerzenia perlowe, 263
 zakresy znaków, 258
 znaki specjalne, 255
 wyrażenie
 \$(), 183
 <<, 140
 wyszukiwanie dat w plikach, 251
 wyświetlanie napisu, 19, 30
 wywoływanie funkcji, 174
 wzorce, 255

Z

zagnieżdżone instrukcje warunkowe, 44, 46, 48
 zakończenie skryptu, 55
 zakresy
 liczbowe, 258
 literowe/znakowe, 258, 260
 zapisywanie wartości w tablicach, 153
 zastosowanie zmiennych lokalnych, 180
 zbiór liczb, 97
 zgodność adresów IPv4, 201
 złożoność obliczeniowa
 liniowa, 222
 wykładnicza, 220

- zmiana
 - bieżącej ścieżki, 33
 - uprawnień plików i katalogów, 33
 - wartości zmiennej, 26
- zmienna, variable, 21, 25
 - \$?, 53
 - \$@, 146
 - \$PWD, 33
- zmienne
 - globalne, 178, 185
 - liczbowe, 87
 - lokalne, 180
 - specjalne, 72, 184
 - tablicowe, 146, 151
 - tylko do odczytu, 207
 - w funkcjach, 178
 - w HEREDOC, 141
- znak
 - #, 24
 - &, ampersand, 43
 - @, 27, 124
 - =, 21
 - gwiazdki, 261
 - kontynuacji wiersza, 27
 - końca linii, 22
 - równości, 40
 - średnika, 40
 - tyldy, 22, 33
 - ucieczki, escape character, 28, 32
 - zachęty, prompt, 20, 24, 27
 - zapytania, 262
- znaki
 - dopasowujące, 263
 - specjalne, 28–31, 255
 - specjalne Perla, 263
 - zwracanie wartości, 185

Ż

- źle zaprojektowane
 - parametry wejściowe, 188
 - użycie zmiennych, 189
 - zwracanie wartości, 190

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

BASH

PRAKTYCZNE SKRYPTY

Powłoka Bash jest bardzo wygodnym narzędziem, pozwalającym na automatyzację wielu różnych czynności, kłopotliwych dla administratora systemu. Jest też uniwersalna: choć powstała jako powłoka dla systemów uniksowych, można używać jej także na komputerach wyposażonych w Windows. Jeśli chcesz sprawdzić, do czego warto użyć Basha, i zobaczyć, jak działa powłoka w konkretnych sytuacjach, koniecznie zajrzyj do tej książki. W przystępny, klarowny sposób omawia ona podstawowe zagadnienia związane z Bashem, a nade wszystko zawiera praktyczne, gotowe skrypty i liczne ćwiczenia.

W książce znajdziesz zestawy instrukcji pozwalających efektywnie wyszukiwać w dużej bazie określone pliki, sortować, wykonywać skomplikowane obliczenia, generować zbiory liczb, stosować funkcje rekurencyjne i sprawdzać poprawność konfiguracji sieci komputerowej. Każdy rozdział rozpoczyna się od krótkiego opisu konkretnego problemu, który można rozwiązać dzięki skryptowi Basha. Taki układ na pewno docenią wszyscy, którzy nie mają zbyt wiele czasu na poznawanie teorii, gdy natychmiast potrzebna jest im praktyka. Sprawdź, zastosuj i oszczędź swój czas!

- Konwencje programistyczne
- Instalacja i konfiguracja środowiska pracy
- Zmienne, stringi, podstawowe polecenia
- Tryb interaktywny/konwersacyjny i wsadowy
- Instrukcje, operatory, pętle i tablice
- Parametry wejściowe, zmienne liczbowe i wyrażenia matematyczne
- Analiza zbioru plików pod kątem daty ich ostatniej modyfikacji
- Sortowanie liczb i funkcje
- Sprawdzanie poprawności konfiguracji sieci komputerowej
- Ciągi Fibonacciego
- Prosty kalkulator ze wsparciem dla arytmetyki
- Systemy rezerwacji miejsc
- Rekurencyjne wyszukiwanie dat w plikach

Niech Twój komputer
działa za Ciebie!

Helion

36826 numer katalogowy

księgarnia Internetowa



<http://helion.pl>

zamówienia telefoniczne



0 801 339900



0 601 339900

Sprawdź najnowsze promocje:

- <http://helion.pl/promocje>
 - Książki najchętniej czytane:
 - <http://helion.pl/bestsellery>
- Zamów informacje o nowościach:
• <http://helion.pl/nowosci>

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
<http://helion.pl>

sięgnij po WIĘCEJ



KOD KORZYŚCI

ISBN 978-83-283-1489-4



9 788328 314894

Informatyka w najlepszym wydaniu

cena: 59,00 zł