max planck institut informatik

Universität des Saarlandes
FR Informatik

Uwe Waldmann

January 22, 2008

Tutorials for “Unix for Advanced Users”
Exercise sheet 10

Exercise 10.1:
What does the command

```
perl -ne 's/^([^ ]* [^ ]*) .* /$1 /; print $_;'
```

do intuitively?

Exercise 10.2:
You have a file in which each line contains an item and its price in the form

```
Flight                          $577.90
Taxi from airport to hotel       $24.50
Hotel (3 nights)                $330.00
Taxi from hotel to airport       $26.00
```

Write a Perl program that computes the sum of all the prices. (Remember that Perl converts strings into numbers on the fly if needed.)

Exercise 10.3:
You have a file in which each line contains data about a politician in the form

```
Gordon Brown, Prime Minister, United Kingdom
George W. Bush, President, USA
Angela Merkel, Chancellor, Germany
Nicolas Sarkozy, President, France
```

Write a Perl program that converts this file into the form

```
Brown, Gordon
Bush, George W.
Merkel, Angela
Sarkozy, Nicolas
```

You may assume that everything after the first comma is not part of the name and that the last name does not contain spaces. One substitution command should be sufficient.

Exercise 10.4:
How do the two commands

```
perl -ne 'chomp; if (/^[^#]/) {print $_, "\n";}'
```

and

```
perl -ne 'chomp; if (! /^#/) {print $_, "\n";}'
```

differ? How can you get the same result using `grep` instead of Perl? How can you get the same result using `sed` instead of Perl?

Exercise 10.5:
Suppose that the value of the shell variable `A` contains at least one hyphen. What does the bash command

```
B=${A#*-}
```

do? How could you get (almost) the same effect in a classical Bourne shell using the `expr` command? (Challenge question: why *almost*?)