Stefan Kurtz,
Dirk Willrodt

Universität Hamburg
Zentrum für Bioinformatik

Programmierung in der Bioinformatik
Winter
Übungen zur Vorlesung: Ausgabe am 14.10.2013

Ablauf des ersten Übungstermins am 14.10.13:

für alle nicht MsC Bioinformatik Studenten und alle die noch keine ZBH Rechneraccount haben:

- Einführung in die Benutzung der ZBH-Rechner
- Ausgabe von Schlüsselkarten

Die Bioinformatikstudenten haben wärend der Zeit eine Orientierungsveranstalltung.

Einige der Aufgaben können nur auf den Rechnern im ZBH-Pool durchgeführt werden. Sie können sich natürlich die evtl. notwendigen Dateien auch auf Ihren Rechner zuhause kopieren.

Am zweiten Termin am 21.10.13 haben Sie dann die möglichkeit die Übungsaufgaben zu lösen.

Die Übungen sind Pflichttermine und Übungsaufgaben werden größtenteils direkt bearbeitet.

Aufgabe 1.1 Im Verzeichnis `/projects/lehre/programming/data/` finden Sie zwei Dateien `ecoli.seq` und `swiss`. Ermitteln Sie mit Hilfe der Befehle `cat`, `tr` und `wc` die Anzahl der Zeilen in diesen Dateien, sowie die Anzahl der Zeichen ohne `newline`. Hinweis: Für den zweiten Teil der Aufgabe sollen Sie "pipes" verwenden (das Symbol `|`), damit die Ausgabe eines Programmes als Eingabe für ein anderes Programm dienen kann.

Aufgabe 1.2 Erzeugen Sie zwei Dateien `datei1` und `datei2` mit verschiedenen Inhalten. Mit welcher möglichst kurzen Sequenz von Unix-Kommandos kann man die Namen der beiden Dateien vertauschen? Welche Probleme können dabei auftreten?

Aufgabe 1.3 Entwickeln Sie eine Folge von Kommandos, die den Anfang einer Datei im Fastaformat auf die Standard-Ausgabe formatiert ausgibt. Die Ausgabe soll nicht die Kopfzeile (markiert durch ein > am Anfang der Zeile) der Eingabe-Datei enthalten, und dann eine bestimmte Anzahl von Zeichen enthalten. Der extrahierte Bereich der Basen oder Aminosäuren soll auf höchstens 60 Zeichen pro Zeile formatiert werden. Die Ausgabe der ersten 123 Zeichen der Datei `/projects/lehre/programming/data/ychrIII.fna` soll also das folgende Ergebnis liefern:

```
CCCACACACCACACCCACACCACACCCACACACCACACACACCACACCCACACACCCACA
CCACACCACACCCACACCACACCCACACACCCACACCCACACACCACACCCACACACACC
ACA
```

Benutzen Sie für Ihre Lösungen Pipes und die Programme `grep`, `tr`, `head` und `fold`.

Aufgabe 1.4 1. Suchen Sie mit dem Kommando `find` im Verzeichnis `/usr/include` und den Unterverzeichnissen alle Dateien, die mit `std` beginnen und mit `.h` enden. Wieviele solche Dateien gibt es?

2. Benutzen Sie die Kommandos `touch` und `find`, um in Ihrem Home-Verzeichnis alle Dateien aufzulisten, die am aktuellen Tag verändert wurden.
3. Legen Sie ein Verzeichnis mit insgesamt etwa 10 Unterverzeichnissen und Dateien auf verschiedenen Hierarchie-Ebenen an. Benutzen Sie nun `find`, um in Ihrem Directory alle Unterdirectories (jedoch keine normalen Dateien) für die Gruppe und für alle anderen Benutzer lesbar und ausführbar zu machen. Mit `ls -lR` listet man den kompletten Verzeichnisbaum inklusive aller Rechte auf.
4. Geben Sie einen `find`-Aufruf an, um in allen Dateien eines Unterverzeichnisses, die mit `.txt` enden nach dem Wort `UNIX` zu suchen.

Aufgabe 1.5 Wenn man Dateien löscht, kopiert oder ihre Namen ändert, dann muß man sehr vorsichtig mit den entsprechenden Kommandos `rm`, `cp` und `mv` umgehen, damit man nicht versehentlich Dateien löscht. Daher bieten alle drei Kommandos eine Option, die dazu führt, dass vor dem Löschen oder Überschreiben einer Datei der Benutzer diese Aktion mit `y` bestätigen muß. Finden Sie heraus, um welche Option es sich handelt. Mit Hilfe des Kommandos (*tcsh*-syntax):

```
alias rm 'rm <fragebeimloeschenoption>'
```

erreicht man, dass für `rm` diese Option automatisch verwendet wird. Tragen Sie für alle drei Kommandos entsprechende `alias`-Einträge in die Datei `~/.tcshrc` ein. Nach dem Sichern der Datei und dem Befehl

```
source .tcshrc
```

bzw. nach jedem einloggen sind die `alias`-Einträge dann aktiv.

Wenn man unter Unix ein Programm `p` ausführen möchte, wird der Installationsort (Pfad) anhand der Umgebungsvariablen `PATH` ermittelt. Es werden alle in der Umgebungsvariablen `PATH` gelisteten Verzeichnisse (Ausgabe mittels `echo $PATH`) durchsucht. Sei `d` der erste Pfad, der eine ausführbare Version des Programms `p` enthält. Dann wird das Programm `p` im Verzeichnis `d`, notiert durch `d/p` ausgeführt. Wenn man ein Programm aufrufen möchte, das in keinem der in `PATH` spezifizierten Pfade vorhanden ist, so muss der entsprechende Pfad dem Programm vorangestellt werden. Dabei kann man die Notation für relative Pfade (' . ', ' . . ') verwenden. D.h. Programme im aktuellen Verzeichnis werden mit vorangestelltem `./` aufgerufen. Durch

```
setenv <Variable> <Wert>
```

lassen sich in der tcsh Umgebungsvariablen wie `PATH` festlegen. Nutzen Sie `setenv` um in der Datei `.tcshrc` das aktuelle Verzeichnis permanent dem Pfad hinzuzufügen. Was für Probleme kann diese Praxis mit sich bringen?

Wir raten sogar davon ab dies zu tun, fügen Sie statt dessen *$HOME/bin* hinzu, sofern dies nicht automatisch geschieht. Vergessen Sie auf keinen Fall den Inhalt der ursprünglichen `PATH` Variablen anzufügen. Sie können mehrere Einträge in der Variablen durch `:` trennen.

Was kann man sonst noch in der Datei `.tcshrc` eintragen?

Aufgabe 1.6 Gegeben sei ein Verzeichnis mit sehr vielen Dateien. Der Befehl `ls -l` liefert eine recht unübersichtliche Ausgabe. Außerdem scrollt die Ausgabe immer aus dem Fenster heraus. Wie kann man mit `ls` besonders leicht die zuletzt geänderten Dateien anzeigen lassen?

Die Lösungen zu diesen Aufgaben werden am 28.10.2013 besprochen.