

Before we start:

- download the zip file **data-temp.zip** from Canvas
- Put the data-temp.zip into your home directory
- Unzip the file data-temp.zip and a directory called data-temp will be created.
- Open a terminal and cd to data-temp directory
- List the content of data directory. You will find data files temp.dat, temp-clean.dat and temp-clean1.dat

You can also unzip by typing at the terminal

**unzip data-temp.zip**

## Data sets: fields and field separator

Generally, a Dataset is organized in fields.

A **field is a unit of information** and can contain either numeric or non-numeric data arranged in rows or columns.

A field can be a group of columns or rows. Usually, fields are arranged in columns. In this course, we refer to fields as a group of columns

In the following example the data file has 5 fields separated by a space character. In this example the fields are:

1. First name
2. Last name
3. birth year
4. career
5. area of focus

```
Jane Bolden 1932 author economics  
John Talbot 1945 poet english
```

In this other example, the field separator is a comma, and there are 5 fields

```
Jane,Bolden,1932,author,economics  
John,Talbot,1945,poet,english
```

## Data sets: fields and field separator

- What is the field separator of temp-clean.dat?
- What is the field separator of temp-clean1.dat?

## wc command

**wc (options) filename**

Try:

**wc temp.dat**

**wc -l temp.dat**

`man wc` #find out other options of the wc command

## grep: select lines matching patterns

The grep utility searches and selects lines that match one or more patterns.

**grep (options) pattern filename**

### Options

-c option: return a count of selected lines

**grep -c pattern filename**

-v option: Selected lines are those not matching a pattern.

**grep -v pattern filename**

multiple -e options: Selected lines are those matching any of the specified patterns.

**grep -e pattern1 -e pattern2 filename**

```
grep CA temp-clean.dat      #extract all line containing CA
grep -c CA temp-clean.dat   #count number of lines containing CA
grep -v CA temp-clean.dat   #extract lines not containing CA
grep -e CA -e CO temp-clean.dat #extract lines containing CA or CO
```

## sort: sort files by lines

**sort (options) filename**

### Options of sort

- u** sort and remove duplicates
- knumber** sort lines based on a certain field *number*
- n** sort fields numerically (if you do not specify this, it will sort alphabetically)

**the default field separator is a sequence of blank space characters**

- tsep** specify *field separator character* (only if it is different than the default separator)

```
sort temp-clean.dat      #sort alphabetically based on 1st field
```

```
sort -n temp-clean.dat   #sort numerically based on 1st field
```

```
sort -k2 temp-clean.dat  #sort alphabetically based on 2nd field
```

```
sort -nk3 temp-clean.dat #sort numerically based on 3rd field
```

```
sort -t: -k3 temp-clean1.dat #specify field separator is : and sort  
alphabetically based on 3rd field
```

## The cut: cuts out selected fields

**cut options filename**

**the default field separator is a tab character**

**-f list of numbers** #specify fields

**-d sep** #specify sep as the field delimiter character instead of the tab character.

**cut -d: -f2 temp-clean1.dat** #select the 2<sup>nd</sup> field

**cut -d: -f2-3 temp-clean1.dat** #select range of fields from 2<sup>nd</sup> to the 3<sup>rd</sup>

**cut -d: -f1,3 temp-clean1.dat** #select specific fields, 1<sup>st</sup> and 3<sup>rd</sup>

## cut: cuts out selected columns

**cut options filename**

**-c list** #list of character positions

**cut -c 1 temp-clean.dat** #extract 1<sup>st</sup> column

**cut -c 1-3 temp-clean.dat** #extract a range of columns from 1<sup>st</sup> to 3<sup>rd</sup>

**cut -c 1,3,10-12 temp-clean.dat** #extract the 1<sup>st</sup>, the 3<sup>rd</sup> and from the 10<sup>th</sup> to the 12<sup>th</sup> columns - notice comma separation



## cat and paste

**cat** and **paste** commands can be used to join files together

Make two files, file1 and file2 and in file1 write Hello, and in file 2 Unix

```
cat file1 file2          # vertically concatenate
```

```
paste file2 file1       # horizontally concatenate
```

```
# default join separator for past is one tab
```

Explore the option **-d** of paste. Use file1 and file2

**sed** (*stream editor*) is a Unix utility that parses and transforms text

```
sed 's/word1/word2/' filename
```

```
sed 's/word1/word2/g' filename #If you add g in the end, it will  
replace all occurrences of word1 in word2
```

Make a file called sed-example and write in it:

```
I love bla. I said I love bla
```

Then run sed:

```
sed 's/bla/tea/' sed-example
```

```
sed 's/bla/tea/g' sed-example
```

To delete a word use

```
sed 's/bla//g' sed-example
```

```
printf "format" argument
```

```
printf "%[width].[precision]type" argument
```

**%type**

%s string

%i or %d integer

%f float or real number

%e scientific notation or exponential

\n new line

```
printf "%.2f\n" 1.6547 #format float with 2 decimal places
```

```
printf "%.3f\n" 1.6547 #format float with 3 decimal places
```

```
printf "%.0f\n" 1.6547
```

```
printf "%.3e\n" 1250000 #format in scientific notation with 3 decimal places
```

```
printf "%.1s\n" Two #output 1 character
```

## printf to format text: specify width

`printf "[%width].[precision]type" argument`

```
printf "%6.2f\n" 1.6547
```

1.65

```
printf "%-6.2f\n" 1.6547
```

1.65


format  
`%width.precisiontype`



`%6.2f`

## printf – format multiple arguments and include text

```
printf "%w.p$type1 %w.p$type2" argument1 argument2
```



```
printf "%-10.4s %.2f\n" Temperature 1.6547
```

```
Temp          1.65
```

```
printf "Mass %.2f  .. in %s\n" 65.4747 Kg
```

```
Mass 65.47  .. in Kg
```

```
var=58.590  
var1=Maria  
printf "%.2f %s\n" $var $var1
```

```
58.59 Maria
```

Start this exercise from data-temp directory. If you open two terminals, cd into data-temp in both terminals.

Make a file called **A4-textpro.bash** and in it answer the following questions. Follow Q&A format

### **Use the file temp.dat (1-8)**

1. Use an option of the wc command to print the total number of lines of the file temp.dat
2. Look at the first 10 lines of the file temp.dat. Use the appropriate Unix command

### **Use grep for the following points (3-8)**

3. Extract all lines containing keyword CA
4. Count the number of lines containing keyword UT
5. Extract all lines that do not contain #
6. Extract all lines containing either keyword CA or keyword CO
7. Count the number of lines extracted in the previous question (6)
8. Count the number of lines that do not contain either keyword UT or keyword CO

Answer the following in A4-textpro.bash.

### Use the file temp-clean.dat (9-12)

9. What is the field separator of temp-clean.dat? Answer in a comment line
10. Sort temp-clean.dat numerically according to the 3<sup>rd</sup> field
11. Extract the first field of temp-clean.dat
12. Extract the first 4 columns (first 4 characters of each line) of temp-clean.dat

### Use the file temp-clean1.dat. (13-17)

13. What is the field separator of temp-clean1.dat? Answer in a comment line
14. Sort temp-clean1.dat alphabetically according to the 2<sup>nd</sup> field
15. Extract the 2<sup>nd</sup> and 3<sup>rd</sup> field of temp-clean1.dat
16. Optional - Extract columns 1 to 3, and 6 to 9 of temp-clean1.dat

## Activity - Practice printf

In a script called **A4-printf.bash** do the following:

1. With printf produce this output by formatting numbers 13.436 and 100.0

```
13.44 100.00
```

Here you find parts of the code. Your task is to add the precision

```
printf "%f %f\n" 13.436 100.0
```

2. With printf, produce this output by formatting numbers 13.436 and 100.0  
(3 spaces between the numbers)

```
13.4   100.0
```

Here you find parts of the code. Your task is to add the precision and width

```
printf "%f %f\n" 13.436 100.0
```



## Activity - Practice printf

In **A4-printf.bash** do the following:

3. With printf produce this output by formatting numbers 13.436 and 100.0

*The largest voltage is 13.4 mV at time step 100 ms*

Below you find the partial code. Your task is to complete the code

```
printf "The largest voltage is %f mV at time step %f ms\n" 13.436  
100.0
```

4. Make 2 variables v1= 13.436 and v2=100.0 and use them with printf to produce the same output

*The largest voltage is 13.4 mV at time step 100 ms*

**Submit to Gradescope A4**

**A4-textpro.bash 1-15 mandatory 2 points**

**A4-printf.bash 1 point**