Nuttige websites:

- Stackoverflow: http://stackoverflow.com/

- http://unix.stackexchange.com/

- Bc: syntax & vanalles: http://www.basicallytech.com/blog/?/archives/23-command-line-calculations-using-bc.html

- Regex: shorthand classes: Instead of specifying all the characters literally, you can use **shorthands** inside character classes: `[\w]`(lowercase) will match any "word character" (letter, numbers and underscore), `[\W]` (uppercase) will match anything but word characters; similarly, `[\d]` will match the 0-9 digits while `[\D]` matches anything but the 0-9 digits, and so on.

http://stackoverflow.com/questions/5925738/which-regular-expression-operator-means-dont-match-this

- Verander prompt: http://www.cyberciti.biz/tips/howto-linux-unix-bash-shell-setup-prompt.html

- sed: case conversion: https://www.gnu.org/software/sed/manual/html_node/The-_0022s_0022-Command.html

- Brace expansion: http://wiki.bash-hackers.org/syntax/expansion/brace

- if: hele hoop opties: http://tldp.org/LDP/Bash-Beginners-Guide/html/sect_07_01.html

- Git cheat sheet: http://byte.kde.org/~zrusin/git/git-cheat-sheet-large.png

- sed oneliners: http://sed.sourceforge.net/sed1line.txt (ook in nederlands beschikbaar)

- sed ongeveer alles: http://www.grymoire.com/Unix/Sed.html

- sed, maar ook handige site: http://www.computerhope.com/unix/used.htm

- Text processing commands: http://tldp.org/LDP/abs/html/textproc.html

Handige dingen voor computergebruik examen

- OVERAL COMMENTAAR BIJ SCHRIJVEN!!
- NOOIT rm -rf *

- Alle karakters op een nieuwe lijn

sed ‘s/\(.\)/\1\n/g

- Alle woorden op een nieuwe lijn

tr ‘ ‘ ‘\n’

- Alle woorden op een lijn

tr -d ‘\n’
tr ‘\n’ ‘ ‘
paste -sd “ “

SCRIPTS

1) OPTIES

```
#getopts gebruiken om argumenten te verwerken
while getopts ":vc:i:o:" opt; do
#eerste : onderdrukt gegenereerde foutmeldingen
#andere : argument is nodig
     case $opt in
       v  ) verbose=1
            ;;
       c  ) cycli="$OPTARG"
            ;;
       o  ) doeltaal="$OPTARG"
            ;;
       i  ) brontaal="$OPTARG"
            ;;
       \? ) echo "Gebruik: $0 [-v] [-c] [cycli] -o doeltaal -i
brontaal zin" 1>&2
           exit 1
           ;;
     esac
done

shift $((OPTIND - 1))
#Alle opties worden geschift, $1 is het eerste dat wordt meegegeven
```

2) IF

```
if [ test ]
     then
          #commando's
elif [ test ]
     then
          #commando's
else
     #commando's
fi

Numerieke vergelijking:    [[ -eq -ne -gt -ge -lt -le ]]
                           ((< <= > >=))
Strings vergelijken:       [ = == != ]
(zie hier: http://tldp.org/LDP/abs/html/comparison-ops.html)
```

3) CASE

```
case "$1" in
     ding1)     commando's
     ;;
     ding2)     commando's
     ;;
     ding3)     commando's
     ;;
     *)    commando's
     ;;
esac
```

4) WHILE LOOP

```
while commando (vaak [ $i -gt 8 ])
do
     commando
done
```

5) FOR LOOP

```
for VARIABELE in 1 2 3 of file1 file2 of $(commando)
do
     commando's
done
```

6) FUNCTIES

```
function functie {
     echo $1
}

functie Hallo
```

7) FILE DESCRIPTORS

```
exec 3<"bestand"

while read -u 3 line → leest bestand lijn per lijn in
do;done

stdin: fd 0, stdout: fd 1, stderror: fd 2
```

GIT COMMANDO'S

- Basis

master: default branch
origin: default upstream repository (remote repository ?)
HEAD: current checked out branch/commit
HEAD^: parent van HEAD
HEAD^2: 2e parent van HEAD
HEAD~4: 4e commit voor HEAD

- Repository creëeren

git init
git remote add [url]
pull, daarna push ok :p

- Repository die al bestaat clonen

git clone [adres] en hierachter nog een leuke naar zoals lekkere dieter :D

- Dingen tonen

git status: veranderingen in working directory
git diff: changes to tracked files
git diff $commit1 $commit2: veranderingen tussen 2 commits
git log: geschiedenis veranderingen
git blame $file: wie veranderde wat?
git show $commit
git branch: alle locale branches

- 'Undoen'

git revert HEAD: revert laatste commit, creeert nieuwe commit → met anderen
git reset HEAD: reset laatste commit
git commit -a --amend: laatste commit aanpassen

- Branches

git checkout: switch to branch
git branch naam: creeert branch
git checkout -b newbranch ander: creert nieuwe branch gebaseerd op ander
git branch -d naam: delete branch
git branch -f branch1 branch2: pointer branch 1 naar pointer branch2

- Commits aanpassen

git cherry-pick commit: neemt commit en zet hem onder uitgecheckte branch

git rebase (-i) HEAD~4: i is interactive, neemt de 4 vorige commits

- Remote: updaten

git fetch: laatste veranderingen, geen merge
git pull: laatste veranderingen, wel merge
git pull -rebase: laatste veranderingen, rebase
git pull origin bar:foo pullt bar naar foo
?git pull origin :foo delete foo

- publish

git commit -a: alle lokale veranderingen publiceren
git push: veranderingen naar origin pushen
git tag commit tag: taggen
git describe: omschrijft hoeveel commits van een tag de commit is
git push origin foo:bar pusht branch foo naar branch bar in remote
?git push origin :bar delete bar

- merge conflicts oplossen

git diff
git diff --base $file: base file
git diff --ours $file: jouw veranderingen
git diff --theirs $file: andere veranderingen

discard conflicting patch
git reset --hard
git rebase --skip

mergen na het oplossen van conflicts:
git add $file
git rebase --continue

OEFENINGEN OPLOSSINGEN

Oefeningen voor degene die denken dat ik niets uitsteek :p

==========

:2,$s/\t/ - /g replace tabs with [-]

:1,1s!^!#! add # before start and between line 1,1

:%s/.*/\L&/ change all lowercase first letter to uppercase first letter

:2,$s/\([A-Z]\)\([A-Z]*\)/\u\1\L\2/g keep first letter of first word as capital, other letters to small of first word

:g/N\./s/\([A-Z]\)\([a-z]*\)/\1\2(North) search to all lines containing N. and add after word (North)

:g/S\./s/\([A-Z]\)\([a-z]*\)/\1\2(South) search to all lines containing S. and add after word (South)

:2,$s/\(.*\) - \(.*\) - \(.*\) - \(.*\) - \(.*\) - \(.*\) - \(.*\) - \(.*\)/\3 - \2 - \8 - \7 - \5 - \6

==========

history | sort -k2,2 | uniq | cut -d ' ' -f4 | uniq -c | sort -n -r -k1,1 | head

==========

:6,$s/\([a-z][a-z][a-z]\)[a-z]*/\1\/200 replaces from line 6, like januari ==> jan/2000

:%s/\([0-9]*\.[0-9]*+\)[0-9]*,*[0-9]*%+\?/\1/g replaces 7.77+7?77% by 7.77+

:%s/\(.*\)+$/\1 removes + at the end

:g/.*+\([0-9]*\)\.[0-9]*+\1\.[0-9]*+\1\.[0-9]*+\1\.[0-9]*+\1\.[0-9]*+\1\.[0-9]*+\1\.[0-9]*/s/\(.*\)/\1+STABIEL

zoekt alles wat gelijk is en gaat dan alles nemen, en er de groep plus stabiel doen

:1,$! tr '0123456789' 'zyxwvutsrq' zet de cijfers om in letters.

:6,$s/\([^+]*\)\(+[^+]*\)\(+[^+]*\)\(+[^+]*\)\(+[^+]*\)\(+[^+]*\)\(+[^+]*\)\(+[^+]*\)\(+[^+]*+*[^+]*\).*/\1\2\r\1\3\r\1\4\r\1\5\r\1\6\r\1\7\r\1\8\r

gaat de groepen verwisselen van plaats, en er telkens hier en daar een nieuwe regel tussen steken.

==========

cat sanger.a sanger.c sanger.g sanger.t | sed 's/\(.\)/\1\n/g' | column -c 32 | tr -d '-' | tr -d '\t' | tr -d '\n'

column gaat de regels tellen en verdelen in kolommen.

gebruik sed 's/\n/ /g' om newlines om te zetten in spaties

==========

cat getallen.txt | sed "s/\(.*\)/echo \1\% \$(cat getallen.txt | head -n1)/" | bash - | bc | tail -n +2 | sed 's/[^0]//' | tr -d '\n' | wc -m

gaat alle getallen uitschrijven door echo, gevolgd door een %, gevolgd door het eerste cijfer. Vervolgens gaat het alles uitvoeren via bash - en
het commando bc. Vervolgens gaan we via tail -n +2 het eerste getal verwijderen, omdat dit niet meetelt. Vervolgens gaan we de nullen en new lines
verwijderen. Uiteindelijk gaan we dit tellen.

tail -n 2 toont de laatste 2 regels van het bestand
tail -n +2 toont vanaf de 2de regel tot de laatste van het bestand
sed '1d' gaat ook de eerste regel verwijderen, werkt sneller en efficiënter

==========

cat eiwitten.txt | egrep '^(([KR]P)|[^KR])*[KR] '

cat eiwitten.txt | egrep -v '^.*(([GALVI][DENQ])|[DENQ][GALVI]).* '

cat eiwitten.txt | egrep -v '^.*(.).?\1.* '

cat eiwitten.txt | egrep '^.*(([GALVI]([DENQ][GALVI]){3,})|([DENQ](([GALVI][DENQ]){3,}))).* '

==========

cat morse.txt | egrep '^[^-]*-([^-]*-[^-]*-)*[^-]* '

cat morse.txt | egrep '^(\.-)* |^(-\.)* '

cat morse.txt | egrep '^(.*)\1 '

cat morse.txt | egrep -v '--\.\.|\.\.--'

==========

Zo zal

:1,10s/a/A/

bijvoorbeeld in de eerste tien regels telkens de eerste “a” vervangen door een “A”, terwijl

:1,10s/a/A/g

alle “a”’s in de eerste tien regels zal vervangen

==========

:1,$s/{"StratenInGent":\[\(.*\)\]}/\1/g

:1,$s/},{/}{/g

```
:%s/\{/\r/g

:1,$s/}//g

:1,$s/\("[a-z]*[_]*[A-Z]*[a-z]*\"\:"\)*//

:1,$s/"//g

1,$s/,/;/g

:%s/\([^;]*\);\([^;]*\);\([^;]*\);\([^;]*;[^;]*;[^;]*;[^;]*;[^;]*;\)\([^;]*\);\([^;]*\);\([^;]*\);\([^;]*\)/\3;\8;\6;\1;\4

:g/[^;]*;[^;]*;[^;]*;[^;]*;[^;]*;[^;]*;;/d

:g/[^;]*;[^;]*;[^;]*;[^;]*;[^;]*;[^;]*;[^;]*;[^;]*;;/d1

:1,$!sort -t';' -k8,8r -k6,6r -k1,1


====================


cat multitap.txt | egrep '^([^13579]|( *([13579]) *(\3) *)+)* '

cat multitap.txt | egrep '^(([13579])(\2| )*)? *(([24680])(\5| )* *([13579])(\7| )* *)* *(([24680])(\8|
)*)? '

cat multitap.txt | egrep -i '(^3[^3| ^3])|([^3|^3 ]3[^3| ^3])|([^3|^3  ]3 ) '

cat multitap.txt | egrep -v '(.) *\1 *\1 *\1'| egrep -v '^.*(.) *(\1).*(.) *(\3).*(.) *(\5).* '


====================


find ~/Computergebruik/Oefeningenreeks_examen/ -type f -exec rename 's/_//' * \; 2>
/dev/null

find ~/Computergebruik/Oefeningenreeks_examen/* -type f -exec rename 's/_//' * \; 2>
/dev/null
```

het eerste verwijderd enkel in de map zelf, het tweede verwijderd ook de _ in de onderliggende mappen.

==========

find -name "*.txt" -exec wc -l {} \; | sort -r | sed 's/\([0-9]*\) \(\./[a-z]*\.[a-z]*\)/\1/g' | sed "s/\(.*\)*/\1 \+ /g" | tr -d '\n' | sed 's/\+ $//p' | bc 2> /dev/null

find . -type f -exec cat {} + | wc -l

==========

cat raven.txt | tr [:upper:] [:lower:] | tr ' ' '\n' | | sed 's/ ... //g' | sed 's/ .. //g' | sed 's/ .. //g' | sed 's/\-\([a-z]*\)\-/\1/g' | sed 's/-[^a-z]|[^a-z]-//g' | sed '/^$/d'

cat raven.txt | sed "s/[^A-Za-z'-]/\n/g" | sed "s/^\('\|-\)//g" | sort | uniq -c | sort -nr | head -n 30

cat raven.txt | dos2unix | tr [:upper:] [:lower:] | tr ' ' '\n'| sed '/^-$/d' | sed '/^$/d' | sort -r | sed "s/^\('\|-\)//g" | sed 's/,//g' | sed 's/`//g' | sed '/^..$/d' | sed '/^.$/d'

cat raven.txt | dos2unix | tr [:upper:] [:lower:] | tr ' ' '\n'| sed '/^-$/d' | sed '/^$/d' | sort -r | sed "s/^\('\|-\)//g" | sed 's/,//g' | hiermoetnogdikkeshittussenkomen | sed 's/`//g' | sed '/^..$/d' | sed '/^.$/d'

cat raven.txt | dos2unix | tr [:upper:] [:lower:] | tr ' ' '\n'| sed '/^-$/d' | sed '/^$/d' | sort -r | sed "s/^\('\|-\)//g" | sed 's/,//g' | sed 's/`//g' | sed '/^..$/d' | sed '/^.$/d' | uniq -c | sort -r | head -30 | sed 's/\(.*\) \(.*\)/\2:\1/g' | tr -d ' '

==========

echo {1..999..2} | tr ' ' '\n'

cat oneven.txt | sed -e 'n;s/^\(.*\)$/echo "scale=10;-4\/\1" | bc/' | sed -e 's/^\([0-9]*\)$/echo "scale=10;4\/\1" | bc /' | bash - | sed 's/^\(\.[0-9]*\)$/+\1/' | tr -d '\n' | sed 's/^\(.*\)$/echo "\1" | bc/' | bash -

==========

alle hexadecimale getallen bestaan uit 1 arabisch cijfer en 1 hexletter

cat hexdump.txt | egrep -cv '([0-9][0-9]|[a-f][a-f]).* \|'

2e hexadecimale cijfer van hexadec getal is telkens gelijk aan 1e hexadecimale cijfer van volgende hexadec getal

cat hexdump.txt | egrep '^.((.) \2)*. \|'

elk hexadec cijfer komt 2 keer voor in lijst hexadec getallen

cat hexdump.txt | egrep -vc '.*([^]).*\1.*\1.* \|.*'

woord bevat geen enkele hexletter uit lijst hexadec getallen

cat hexdump.txt | egrep -vci '.*(.).* \|.*\1'

==========

cat brain.txt | egrep '.*>...(.).*\1... '
cat brain.txt | egrep '.*(>.....>)+.* '
cat brain.txt | egrep -v '.*\(>\(\+\)+>\)*.*|.*\(>\(-\)+>\)*.* '
cat brain.txt | egrep -v '.*<(--)+[^-]*>.* '

cat brain.txt | egrep -c '^.*\]...(.).*\1... '

cat brain.txt | egrep -c '\[(.....).*\]\1.* '

cat brain.txt | egrep -v '^.*\].*(\-[^-+]+\-).* ' | egrep -v '^.*\].*(\+[^-+]+\+).* '

egrep -v '^.*\].*(\++[^-+]+\++).* '

```
egrep -v '^.*\].*(\+[^-+]+\+).* '

cat brain.txt | egrep -v '<([^-<>]*-[^-<>]*-[^-<>]*)*>' | egrep -v '<[^-]*>'




cat brain2.txt | egrep '.*(>.........*>)*.* ' | wc -l
cat brain2.txt | egrep '.*<([^\.][^\.](\.)*){3,}>.* ' | wc -l

cat brain2.txt | egrep '(+.......+)* '
```