.csh je c shell skript ale ta pripona tam nemusi byt, to je len pre nas ako ludi, aby sme vedeli co to je, lebo v UNIXe su tieto subory takzvane **obycajne subory**

**priklady su v /home/OS/ucebnove/01-04/os\_2-priklady --- tam si treba vsetky pozriet aj a.csh**

**chmod XXX /subor/ -** read write execute, kde RWX musime v dvojkovej uvazovat, cize

4 je read, 2 je write, 1 execute, 0 je ze nic. Prava zmenu prav si nemozem odobrat, len pristupove prava si mozem menit

**chmod 777 /subor/ -** lebo 7 je binarne 111 111 111 preto to da vsetky pristupove prava

**chmod u+r /subor/ ---** nastavi suboru prava ze pride ownerovi/userovi pravo read

**chmod g+x /subor/ ---** nastavi groupe ze prida pravo na execute

**chmod o+w /subor/ ---** nastavi others ze pride pravo na write

**./subor.csh ---** spusti skript s nazvom subor.csh

**bc –** je kalkulacka

v PS v STAT pismenko **S** znamena ze Sleep

ctrl+z len zastavi program a prikazom **jobs** si viem pozriet co bezi

ctrl+c to uplne zrusi ten job

**kill** **%cislo –** vypne process podla cisla pod prikazom jobs

**kill id –** vypne process podla IDcka z prikazu ps -xl

**fg %cislo –** to iste fg id ale podla toho jobs  
**fg id** – foreground da do popredia process s IDckom

**bg %cislo –** to iste co bg id ale cislo je podla toho jobs

**bg id –** da do pozadia ten beziaci process a ja popri tom mozem aj pracovat, lebo inak by script bezal a ja nemozem nic robit

mozem aj spustit script a dat vystup do suboru prikazom **cesta/script > suborVystupovy &**

ampersant spravi, ze to pojde do background, bez ampersantu by to slo do suborVystupovy ale on bezi ten script a ja nemozem popri tom pracovat

ak chcem aby script aj vypisoval na obrazovku a aj do suboru tak dam na to prikaz

**./z.csh | tee /subor/**

v skripte prikazy

#toto je komentar

sleep /pocet sekund/

while (podm)

end

foreach **u** (steinmul lesko paulen)

who | grep -w $**u**

**tak to u**je vlastne substitucia a ono tam doplna slova zo zatvorky

set **premenna** = “Ahoj svet“

@ pocet = 3 **je to iste ako** set pocet = 3 **ale ked dam @ pocet = 3+3 tak to vykona ze 6**

**ale ked dam set pocet = 3+3 tak to tam da aj tak len 3 a dalsie znaky uz nebere**

**set ahoj = 2 druhaPramenna = 4 tretiePremenna = “text“**

takze byvaju ulohy ze do nazvu suboru da \* a ta potom robi taketo substitucne srandy a preto to bude treba poriadne eskejpovat uvodzovkami

\* - je uzaver ze nula a viackrat sa tam daco nachadza

+ - je uzaver ze minimalne raz a viackrat sa daco tam nachadza

daky backthick je substitucia prikazu a ten znak dam vedla jednotky nalavo ``````

**echo “$pocet $veta“ ---**  vypise obsah premennej pocet, da vela medzier a da premmennu veta a dolar hvori, ze je to premenna, lebo bez dolaru by to vypisalo ako text

unset /premenna/ - odoberie premennu

cisto prikaz set mi zobrazi vsetky premenne

@ pocet -- **dekrementuje premennu**

**tr /znakyPovodne/ /znakyNove/** – nahrazduje znaky

cize premenuj subor z AaaaB na tak aby vsetky znaky boli male pismenka cize prikaz **echo mv -i “$f“ “`echo ‘$f‘ | tr A-Z a-z`“**

**Brano nam moze dat aj priklad ze mame premenovat subor s nazvom “A B“ a zrazu nam kvoli substitucii tie medzery zmiznu, tak to musime vyeskejpovat uvodzovkami na spravnom mieste a to takto: echo mv -i “$f“ “`echo ‘$f‘ | tr A-z a-z`“**

**a teraz na dalsia specialitka je ked nazov suboru je “A‘B“ tak volako ten apostrof treba precitat**

**a to je zaroven aj domaca uloha**

treba si davat pozor na syntax lebo kazdy interpreter (bash/cshell/...) to inak interpretuje

ale ak napisem prikaz do skriptu prvy prikaz ze **#! /bin/csh**  - a vdaka tomu to vie, ze to ma vykonavat v cshell

**SEMINAR**

ked pisem cestu alebo nazov suboru a TABulator ma nechce doplnit, tak ked stlacim ctrl + d, tak mi vypise vsetky moznosti

**UVODZOVKY**

**‘ ’ - odstrania funkcie specialnych charakterov okrem ‘ ’ a !**

**“ ” – odstrania funkcie specialnych charakterov okrem $, “ , `, !**

**uvodzovky hovoria, ze tie specialne znaky nie su urcene pre shell ale pre pogram napr pre grep**

**\specZnak – ten backslash prepne spec funkciu specialnych znakov, ak ma byt povodne zapnuta, tak ju vypne, ak je vypnuta, tak ju zapne, preto napr v grepe pred dolarom vypina funkciu dolaru ale pred znakmi ako > alebo < alebo + zapina ich specialnu funkciu znamu pre grep, inak by ich bral ako text dolaru aj pre shell aj pre grep**

ked chcem napisat specialne znaky ako napr $ a pod.. tak ak ich chceme brat ako

v grepe: --a teda aby sme v grepe mohli pouzit dolar ako koniec riadku, tak ho musime eskejpnt apostrofmi alebo backslashom

**“” – tieto uvodzovky platia len pre shell ale on ten shell to posiela este do grepu a grep ma svoje vlastne specialne znaky, preto grepovske specialne znaky musim backshlasovat aj v tych ‘’ lebo to vlastne eskejpovavam pre grep, napr aj bodka ma pre grep vzynam ze je to hocijaky znak**

**?** – znamena, ze to co je pred nim tam moze ale nemusi byt, automaticky je --

**!**  - je koniec riadku, automaticky je **zapnuta**

**$** - je koniec riadku, automaticky je **zapnuta**

**+** - je 1 az viac krat to co je pred tym pismenko, alebo slovo v zatvorke, automaticky je vypnuta

\* - je 0 az viackrat to co je pred tym pismenko, alebo slovo v zatvorke, automaticky je vypnuta

**^** - zaciatok riadku, automaticky je **zapnuta**

> - koniec slova, automaticky je vypnuta

< - zaciatok slova, automaticky je vypnuta

**.** – hocijake pismenko, automaticky je **zapnuta**

**{cislo}** – znak co sa nachadza pred tym hlada cislo krat,napr o\{3\} najde ooo automaticky je vypnuta

zobacik ^ v hranatych zatvorkach je negacia v grepe [^[:space:]] tu je to negacia

**[abc] –** v grepe najdi znak a alebo b alebo c

egrep je daky extended grep a platia tam ine specialne znaky

triedy znakov najdem aj v manual grepu…lower upper alpha, cize ak chcem napriklad len velke pismena v grepe, tak dam [:upper:]

**v priecinku /etc/passwd su udaje o studentoch**

**uniq –** bez prepinacov len urobi, ze to co sa opakuje rata len raz

**sort –** bez prepinacov usporiada veci podla abecedy

**last –** vypise prihlasenia pouzivatelov za posledny /mesiac – v nasom systeme/

**tr -s “znak” ---** viacere vyskyty daneho znaku scvrkne len na jeden, ze ak je napr bbbbbbb tak po tomto prikaze uz bude iba b

**grep "^[a-z]\+14" /etc/passwd | wc -l** --- zrata ludi s menom zakoncenych na 14 v priecinku etc/passwd

**grep -c “\>[.\!?]” /etc/passwd** --- to -c robi to, ze namiesto vypisu na obrazovku ich zrata pocet sediacich vyskytov

**grep -c "\<[[:upper:]]\+\>" ccero10.txt** --- vyhlada pocet slov pozostavajucich z velkych pismen, vnutorna zatvorka je znacenie triedy a vonkajsie zatvorky sposobuju to co uz viem, ze rovnaka funkcia ako [a-z]

**last | tr -s " " | cut -d " " -f1,3** --- takze to tr **-s** zmeni viacere vyskyty medzier len na jednu medzeru a tym cutom vypisem si len 1 a 3 stlpec

**last | tr -s " " | cut -d " " -f1,3 | grep 'ynet\.sk$' | cut -d " " -f1 | sort | uniq | wc -l** --- z lastu najprv zrusime viacere medzere, potom vyrezem prvy a treti riadok a potom vyberiem len tie, co na konci riadku (aby to nebolo meno) koncia na ynet.sk a potom si vyrezem len prvy stlpec a prikazom sort ich usporiadam podla abecedy a potom prikazom uniq spravim, ze tych co sa viac opakuju vypise len raz a potom ich spocitam podla riadkov

na osa.fiit.stuba.sk – rozkliknem plusko pri slove Literatura a mozem studovat

**pre cshell:**

! – pre cshell znamena historia , napr !g – spusti posledny prikaz zacinajuci geckom, !! – spusti posledny pouzity prikaz

$ - pre cshell znamena premenna

` - pre cshell znamena, ze tam substitujem prikaz

? – pre cshell znamena najde jeden jednoduchy znak hocijaky

\* - pre cshell znamena ze najde hocijake

[abc] – pre cshell vzorky a alebo b alebo c

[a-z] – pre cshell znamena ze od a az po zet

{s1,s2} – pre cshell vzorky najde hocijaky zo zadanych retazcov

cize napr **find . -type f -name "[a-z]\*.txt" | wc -l** --- najde subory a spocita ich len tie, ktore zacina malym pismenkom, potom ide hocico a konci to bodka txt