Minimální požadavky

- Znát základní příkazy (viz dále)
  - jméno
  - význam (funkce)
- Umět dohledat detaily příkazu v manuálových stránkách
  - Syntax
  - Význam a funkci přepínačů
  - Argumenty, vstup a výstup příkazu
- Znát funkčnost v rámci uváděných přepínačů (ne samotné přepínače)
- Rozumět zápisu a zpracování příkazové řádky

Minimální požadavky: cvičení 1, 2

- Cvičení 1
  - Syntaxe příkazové řádky
  - Manuálové stránky (**`man`**)
  - Vzdálená práce a přenos souborů (**`ssh, scp`**)
- Cvičení 2
  - Zpracování příkazové řádky
  - Rušení významu spec. znaků (**`" ' \`**)
  - Oddělovače příkazů (**`| & ;`**)
  - Přesměrování vstupu a výstupů (**`> < >>`**)
  - Náhrady (**`~ $ ``)**
  - Náhrada jmen souborů (**`* ? []`**)
  - Editace textu v lib. editoru (otevřít, vložit, smazat, uložit, ukončit)

Minimální požadavky: cvičení 3, 4

- Cvičení 3
  - Informace o aktuálním adresáři (**`pwd`** nebo **`$PWD`**)
  - Informace o souborech (**`ls [-l -d], file`**)
  - Příkazy pro práci se soubory (**`touch, mkdir, ln, cp, mv, rm, rmdir`**)
- Cvičení 4
  - Základní filtry (**`tee, cat, head, tail, cut [-d -f], sort [-n -t -k], uniq, cmp`**)

Minimální požadavky: cvičení 5, 6

- Cvičení 5 a 6
  - Regulární výrazy bez `\( \)`
    - Jednoznakové RE (**`znak \znak . [] [^]`**)
    - Opakování RE (**`*`**)
    - Ukotvení RE (**`^ $`**)
  - **`grep [-i -v]`**
  - **`sed [-n]`**, příkazy sedu **`p, d, s`**
  - **`awk`** (**`print`**, proměnné **`FS, NR, NF, $0, $1, ...`**, řetězce (v """))
- Cvičení 7
  - identita uživatele, základní přístupová práva souborů a adresářů, vyhodnocování přístupových práv (**`chmod, umask`**)
  - **`find`** (princip!, **`-name,-size,-o,-a,\(\),!,-print,-exec`**)
  - Archivace a komprese (**`tar [ctxf], gzip`**)

Minimální požadavky: cvičení 7, 8, 9

- Cvičení 8
  - Procesy (**`ps [-e -f]`**)
  - Signály (význam signálů KILL a TERM) (**`kill, trap, ^C, ^Z`**)
  - Úlohy (**`fg, bg, jobs`**), rozdíl mezi úlohou a procesem
  - Návratový kód (**`exit`**)
  - **`test`** (rozdíl v testování čísel a řetězců, testování souborů)
- Cvičení 9 a 10
  - Základní celočíselná aritmetika (alespoň jeden z **`expr, (()), bc`**)
  - Poziční parametry ($0, $1, ..., $*, $@, $#) (**`shift`**)
  - Složené příkazy (**`if, case, while, for, continue, break`**)
  - Funkce (**`function, return`**)
  - Uživatelský vstup (**`read`**), psaní, spouštění a ladění skriptů

Příklad testu – analýza skriptu I

- Podívejte se na skript `/etc/init.d/volmgt`. Kolik se předpokládá parametrů? Při jaké hodnotě 1. parametru se vrátí návratový kód 0 ?

```
case "$1" in
'start')
        if [ -f /etc/vold.conf -a -f /usr/sbin/vold -a \
            "${_INIT_ZONENAME:=`/sbin/zonename`}" = "global" ];
        then
                echo 'volume management starting.'
                svcadm enable
svc:/system/filesystem/volfs:default
        fi
        ;;
'stop')
        svcadm disable svc:/system/filesystem/volfs:default
        ;;
*)
        echo "Usage: $0 { start | stop }"
        exit 1
        ;;
esac; exit 0
```

Příklad testu – analýza skriptu II

- Podívejte se na skript `/etc/init.d/uucp`. Za jaké podmínky se provede příkaz na řádku 14 (rm ...)? Kam se vypíše chybový výstup tohoto příkazu? Skončí skript při vyvolání až po provedení příkazu rm nebo může skončit dříve?

```
#!/sbin/sh
#
# Copyright (c) 1997 by Sun Microsystems, Inc.
# All rights reserved.
#
#ident  "@(#)uucp       1.9     97/12/08 SMI"

if [ -z "$_INIT_PREV_LEVEL" ]; then
        set -- `/usr/bin/who -r`
        _INIT_PREV_LEVEL="$9"
fi

if [ $_INIT_PREV_LEVEL = S -o $_INIT_PREV_LEVEL = 1 ]; then
        /usr/bin/rm -rf /usr/spool/locks/* >/dev/null 2>&1 &
fi
```

Příklad testu – analýza skriptu III

- Podívejte se na skript `/etc/init.d/kfbinit` na řádky 31-33 (31 je `devpath=/dev`). Co bude na řádce 33 (`devs=...`) v proměnné `devpath`. Co se přiřadí do proměnné `devs`?

```
#
    devpath=/dev
    test -d /dev/fbs && devpath=/dev/fbs
    devs=`/bin/ls $devpath | grep 'kfb[0-9]*[0-9]$'`

#
```

Příklad testu – syntéza skriptu I

- Napište funkci shellu *is_user*, která pro zadaný argument zjistí, zda se jedná o uživatelské jméno v systému (z /etc/passwd i ypcat passwd). Pokud ano, vrátí návratovou hodnotu 0. Pokud ne, vrátí nenulovou hodnotu.

```
function is_user() {
    [ -z "$1" ] && return 2
    grep "$1" /etc/passwd >/dev/null && return 0
    ypcat passwd | grep "$1" >/dev/null && return 0
    return 1
}
```

Příklad testu – syntéza skriptu II

- Napište skript, který pro zadané argumenty zjistí, zda se jedná o uživatele systému (použijte funkci *is_user*). Skript vypíše na stdout pouze uživatele systému.

```
for user in "$@"
do
   if is_user "$user"; then echo "$user"; fi
done
```

```
while [ $# -gt 0 ]; do
   if is_user "$1"; then echo "$1"; fi
   shift
done
```

Příklad testu – syntéza skriptu III

- Napište skript, který pro každého přihlášeného uživatele vypíše seznam procesů, které daný uživatel spustil.

```
UZIVATELE=`finger | tail +2 \
              | awk '{ print $1 }' | sort -u`


for UZIV in $UZIVATELE
do
  echo "Uzivatel: $UZIV"
  ps -U $UZIV | awk 'NR>1 { print "\t" $4 }' | sort
done
```

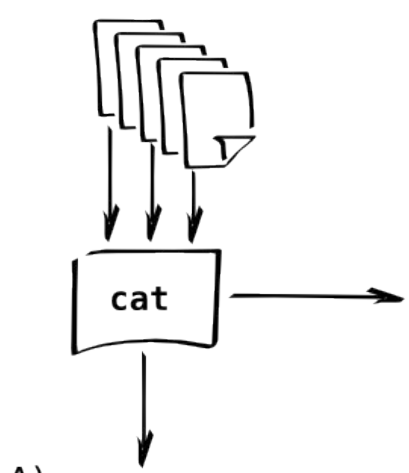
Časté chyby – stdin/args !

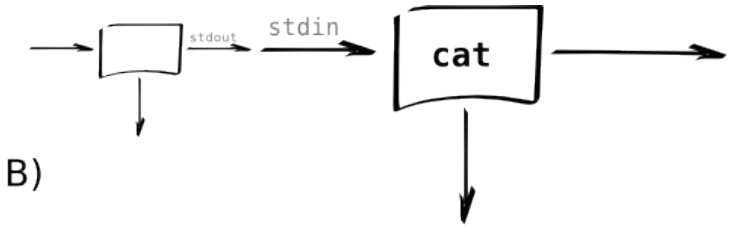
- Vstup dat pomocí stdin a argumentů

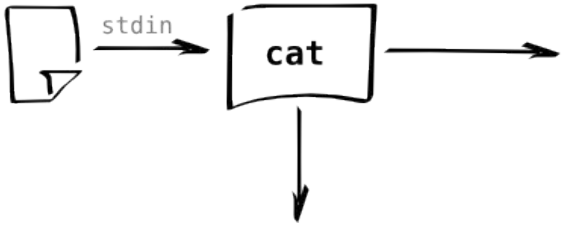
A) `cat **s1 s2 s3**`

B) `ls | cat`

C) `cat < **s1**`







- Některé programy čtou stdin, některé ne
  - Většina filtrů funguje podle A), B) i C)

`grep RE **s1 s2 s3**; ypcat passwd | grep RE; grep RE **<s1**`

  - Většina programů manipulujících se soubory funguje podle A)

`mkdir **a1 a2 a3**; cp **s1 s2 s3 a1**`

Časté chyby – oddělovače příkazů !

- Co znamenají následující řádky?
  - `p1 p2`
  - `p1 ; p2`
  - `p1 & p2`
  - `p1 | p2`
  - `p1 > p2`
  - `p1 < p2`
  - `p1 >> p2`
  - `p1 && p2`
  - `p1 || p2`

- Syntaxe: find ***adresáře*** *predikáty ...*

```
find /usr -name '*grep'
```

- Použití predikátu `-exec`

```
find ~ -name '*.sh' -exec chmod u+x {} \;
```

- Použití predikátu `-print`

```
find ~ -name '*.sh' -exec chmod u+x {} \; -print
```

- Použití logického operátoru

```
find ~ \( -name '*.sh' -o -name '*.bash' \) -ls
```