

<http://service.felk.cvut.cz/courses/x36054>

sunray1.felk.cvut.cz

shell bash

<http://pitbul.wz.cz>

chemi.muni.cz/vm19n/vyuka/2lebec/prihaz.kdm

ls -l dlouhý výpis
-a i syst. soubory

pwd aktuální adresář

cd změna adr.

mkdir

rmdir

ps výpis běžících procesů

ps -aef všechny

ps -aef | grep pepa vypise řádky obsahující pepa

finger pepa - informace o uživateli pepa

find . -name x.txt hledání v abs. adresáři

find /tmp -name x.pwa a podadresářích
"x.sh"

spuštění příkazů na pozadí - na konec řádky &

cat m.java - výpis souboru

cat m.java > x.java přesměrování výstupu do souboru

find /var -name "*.c" > m.txt

man passwd

man -k passwd

vypadnout &
- výpis, ve kterých všech
man stránkách je to slovo

cp x y

kopírovat x do y

cp x /tmp

move x y

přejmenování

move x /tmp

přesun

cp -x adr1 adr2

more x.txt

vypis souboru postupne

less ~~txt~~ x.txt

— " —

konec y

kill -l výpis signálů

kill 1234 pošle signál procesu

bez notání to je SIGTERM

kill -9 1234 zrušení procesu

echo "reco..." >> příkaz

přidat text do souboru

> soubor prepise

history

historie příkazů

soubor bash-history

ssh sunray1 - přihlášení na jiný stroj
ssh pepa sunray1 - s g

(2)

Editors

joe

Ctrl+K, H nápověda

vi

x máme knabz, dot radky

:w uložit

:q ukončit

i, A přechod do vkládání textu

Ctrl+Z posazení úlohy

fg návrat

vim

emacs

Ctrl+X, Ctrl+S uložení

Ctrl+X, K

znovu zavření pole

příklad

gcc -Wall p.c -o p
|
all warning output

uharabete
viz soubor p2.c

int main (int argc, char *argv[])

poté slova
na přík. řádce slova na
přík řádce

viz prog p3.c

pole

char p[5]

velikost současně deklarace

viz prog. p6.c

alokace paměti

malloc

free

int *g;

g++;

uvěští uharabete o velikost toho objektu;

viz prog p5.c

Debugger

kompiloval s param -g

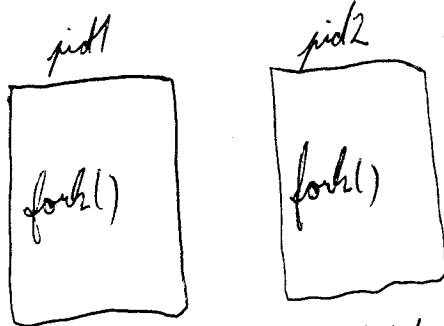
gdb

3

Procesy

fork()

vytvoreni noveho procesu



návrat. hod
z fork()

parent

child

pid2

0

pr. fork.c

exec()

natahne prog. ze souboru a prepise
jim kod procesu

pr. p-exec.c

Úkol 1

```
pid1=fork()
pid2=fork()
pid3=fork()
```

1

```
for( )
:
return 0;
```

2

```
for( )
:
;
```

3

```
for( )
:
;
```

Signály

pr. signal.c

Vlákna

běží v rámci jednoho procesu, sdílí paměť

```
main t1 t2
}
}
}
```

POSIX

~~pthread~~

~~pthread_create(...)~~

~~pthread_join(...)~~

pr. vlakna.c

překládat s -lpthread

(4)

IPC - meziprocesová komunikace

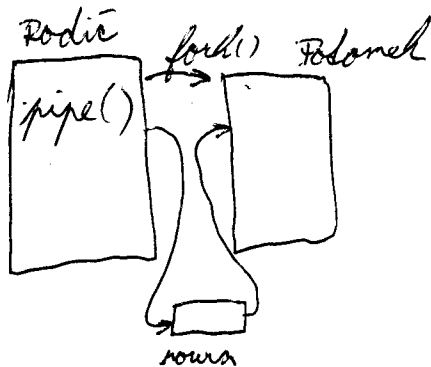
pipe(...) vytvoří komunikační kanál
mezi procesy

int `pd[2]`

`pipe(pd);` - deskriptory pro čtení/zápis vrátí v `pd[]`

zápis `pd[1]`

čtení `pd[0]`



`close(pd[0])` zavření roury

pr. `rouna.c`

př. rovnad.c - nahradí standardní vstup

dup(pd[03]); vloží descriptor na první
volné místo v tabulce

popen na zjednodušení

př. rovnadsmerna.c obousměrná komunikace

sdilená paměť

procesy nemusi být ve vztahu

shmget

shmat

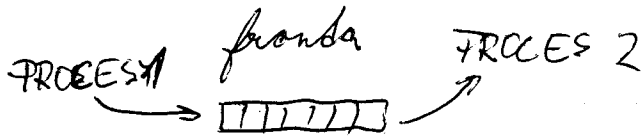
shmofs

shmctl

př. sdilenapamet.c

(5)

Posílání zpráv



msgget()
msgsnd()
msgrcv()
msgctl()

měly by se kontrolovat návratové hodnoty,
hlavně v směrnatce

pr. fronta.c, frontaprijem.c

Mutex

pthread_mutex_

oprávně
I

tester

střídavě pracují
na výrobku

I

I

I

-init()

-lock()

-destroy()

-unlock()

pr. mutex.c

Podmíněné proměnné

pthread_cond-

-^w čekání na změnu
hodnoty proměnné

řadatel

úředník

I - wait()

I - signal()

I

- init

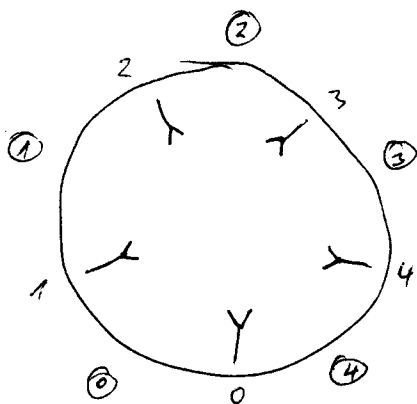
- destroy

řadatel potvrdí, až úředník odemkne
mutex

př. podm.prom.c

(6)

Žeroucí filosofové



O... filozof
Y... vidlička

filozof - žere - má 2 vidličky
- myslí - odloží vidličky

void *filoraf (void *sid)

id = (int) sid

// kichame id filorafa