Università di Verona

Corso di Laurea in Informatica/Informatica Multimediale
Laboratorio di Sistemi Operativi
22 Luglio 2008

Esercizio 1 (16 punti)
Il comando UNIX "chmod" cambia i permessi di ogni dato *file* secondo i *permessi* dati, i quali possono essere o una rappresentazione simbolica delle modifiche da fare o un numero ottale che rappresenta il modello dei bit per i nuovi permessi.
Nel caso in cui i permessi vengono specificati con numeri ottali, la sintassi e' la seguente:

chmod `opzione` {`file`},

ove, `opzione` e' una stringa (nnn) di 3 numeri ottali.

Scrivere uno script bash che converta l'uso del comando **chmod** da modalita' *permessi in rappresentazione simbolica* in modalita' *permessi in ottale*.
Lo script deve ricevere 4 argomenti nella linea di comando:

- La stringa di 3 caratteri XXX indicante i *permessi in rappresentazione simbolica*, ove X={R,W,X} e (R= readable, W=writable, X= executable).
- Il *file* su cui applicare i nuovi permessi.
- Il *file_output.log*, ovvero il percorso completo dove salvare tutto ció che il comando manda sullo standard **output**.
- Il *file_error.log*, ovvero il percorso completo dove salvare tutto ció che il comando manda sullo standard **error**.

Lo script deve controllare che il numero di argomenti della linea di comando sia giusto ed in caso contrario deve visualizzare un messaggio indicante il corretto utilizzo dello script. Lo script deve quindi convertire i permessi dalla rappresentazione simbolica' ad ottale. Infine lo script deve visualizzare un messaggio di errore anche nel caso in cui la directory fornita come argomento alla linea di comando non esista.

Esercizio 2 (16 punti)
Scrivere un programma C in cui un processo P crea un figlio F.
Il padre P deve rimanere in attesa della pressione del tasto *Ctrl C* (da tastiera), dopodiche' deve far terminare il figlio (inviandogli un apposito segnale) e visualizzare il calcolo che il figlio ha compiuto.
Il figlio F deve calcolare la radice quadrata del pid del padre.
Usare il valore di ritorno (exit()) per la comunicazione tra figlio e padre.

Eventuali System Call da utilizzare:

```
#include <stdio.h>                          #include <unistd.h>
#include <sys/wait.h>                       #include <sys/types.h>
int pipe(int filedes[2]);          pid_t fork(void);          int kill(pid_t pid, int sig);
int execle(const char *path, const char *arg , ..., char * const envp[]);
int execl(const char *path, const char *arg, ...);
int execlp(const char *file, const char *arg, ...);
int execv(const char *path, char *const argv[]);
int execvp(const char *file, char *const argv[]);
int dup2(int oldfd, int newfd);
ssize_t read(int fd, void *buf, size_t count);
ssize_t write(int fd, const void *buf, size_t count);
int close(int fd);                          int dup(int oldfd);
int open(const char *pathname, int flags, mode_t mode);
int creat(const char *pathname, mode_t mode);
sighandler_t signal(int signum, sighandler_t handler);
pid_t wait(int *status);           pid_t waitpid(pid_t pid, int *status, int options);
```