Laboratorio di Sistemi Operativi: Prof.ssa A. Rescigno **Anno Acc. 2004-2005**

Esercizi di Verifica Aprile 2005

Università di Salerno

Nome e Cognome: Matricola:

Classe 1 p-p	Classe 2 p-d

1	2	3	4	5	6	7	tot
/10	/20	/12	/10	/17	/18	/13	/100

Spazio riservato alla correzione

1. *10 punti*
   Scrivere il comando Bash per gestire le seguenti situazioni:

Descrizione	**Comando**
visualizzare i nomi dei file presenti nella directory root	
richiamare l'ultimo comando dato che inizia per ls	
visualizzare la lista di tutti gli alias definiti	
ridirigere in un file di nome FF il contenuto di un file EE dopo averlo ordinato	
stampare i nomi di tutti i file presenti nella directory padre	

Descrivere quale é l'effetto dei seguenti comandi Bash:

Comando	**Descrizione**
`ls *[!0-9][0-9]`	
`!!`	
`PS2='\w '`	
`alias readtxt='cat *.txt'` `alias readtxt`	
`echo $PATH:..`	

2. *20 punti*
   Sia

   ```
   ssize_t read9(void *buff, size_t nbytes);
   ```

   una funzione che legge solo da files con file descriptor uguale a 9, cioé `read9(buff, nbytes)` si comporta in maniera identica a `read(9, buff, nbytes)`.

   Allo stesso modo sia

   ```
   ssize_t write9(const void *buff, size_t nbytes);
   ```

   una funzione che scrive solo su files con file descriptor uguale a 9, cioé `write9(buff, nbytes)` si comporta in maniera identica a `write(9, buff, nbytes)`.

   Sia $FILE1$ un file di testo presente nella cwd.

   Scrivere un programma C che utilizzando esclusivamente system calls (tranne `read` e `write`) e le funzioni `read9` e `write9` faccia quanto segue:

   1. (6 punti) crei un file di nome $FILE2$ in cui sia possibile scrivere solo alla fine e che abbia gli stessi permessi di $FILE1$;

   2. (14 punti) legga gli ultimi 512 bytes di $FILE1$ e li scriva in $FILE2$.

3. *12 punti*
   1) Illustare con un disegno la struttura di un File System mettendo in evidenza la directory */home*, che contiene: un solo file di nome *pippo.txt*;

   2) Mostrare sempre con un disegno che cosa accade nel File System se nella directory */home* si creano due nuovi file: *pluto.txt* link simbolico a *pippo.txt*, e *paperino.txt* hard link a *pippo.txt*.

   3) Mostrare sempre con un disegno che cosa accade nel File System se cancelliamo il file *pluto.txt*.

   4) Mostrare sempre con un disegno che cosa accade nel File System se cancelliamo il file *pippo.txt*.

4. *10 punti*
   Si assuma di compilare ed eseguire i seguenti programmi:

   a)
   ```
   int main(void) {
       char buf[ ]="Ho finito il main\n";
       printf("Ho iniziato il main\n");
       write(1,buf,10);
   }
   ```
   Dire quale e' l'output nei due casi sottostanti, giustificando le risposte.
   a) a.out
   b) a.out > file
      cat file

   b)
   ```
   int main(void) {
       char buf[ ]="Ho finito il main\n";
       printf("Ho iniziato il main\n");
       fflush(1);
       write(1,buf,10);
   }
   ```
   Dire quale e' l'output nei due casi sottostanti, giustificando le risposte.
   a) a.out
   b) a.out > file
      cat file

5. *17 punti*
   Si consideri il seguente programma e si supponga di compilarlo.

```
#include<sys/types.h>
#include<fcntl.h>
#include<unistd.h>
#include"ourhdr.h"

int main(){
        if (access("prova.txt", R_OK)<0)
                        err_ret("access error per prova.txt");
        else  printf("access OK\n");
        if (open("prova.txt", O_RDONLY)<0)
                        err_ret("open  error per prova.txt");
        else  printf("open OK\n");
        if (open("prova.txt", O_WRONLY)<0)
                        err_ret("open error per prova.txt");
        else  printf("open OK\n");
        exit(0);
}
```

   Se fosse

```
-r-xr-xr-x 1 rescigno 10932 Jun 4 10:45 a.out
-r-------- 1 rescigno 1891  Jun 4 09:45 prova.txt
```

   ed assumendo che il real user ID sia `studente` rispondere alle domande seguenti

   1) puó `studente` mandare in esecuzione `a.out`?

   2) che cosa succede dando `a.out`.

   3) assumndo ora di settare a 1 il set-user-id di `a.out`, dire se ci sono cambiamenti dando `a.out`.

   In tutti i casi la risposta va giustificata.

6. *18 punti*
   Si dica, giustificando la risposta, che cosa accade quando si compila e si manda in esecuzione il seguente programma.

```
#include     <stdio.h>
int main(void)
{
 pidi_t   pid1,pid2;
 int  s;
        s=0;
        pid1=fork();
       if (pid1==0) { atexit(ex-1); s=s+1;
                      printf("user: %d\n",s);
                      return(0);
                    }
        else   { pid2=fork();
                 if (pid2==0) { atexit(ex-1); s=s+2;
                                printf("user: %d\n",s);
                                _exit(0);
                              }
                else {
                     atexit(ex-1);  atexit(ex-2);
                     wait();
                     wait();
                     printf("user: %d\n",s);
                     exit(0);
                }
    }
}
static void ex-1(void)
     {
     printf("student\n");
     }

static void ex-2(void)
     {
     printf("prof ");
     }
```

7. *13 punti*
   Scrivere un programma C in cui:

   • un processo padre manda ad un processo figlio il segnale SIGALRM ed esce senza aspettare che il figlio termina;

   • il processo figlio resta in attesa di un qualche segnale e cattura il segnale SIGALRM scrivendo sullo standard output il proprio pid.