Lab. di Sistemi Operativi, Gruppo I: M. Faella **Anno Acc. 2006-2007**

Esame del 4 maggio 2007

Università di Napoli "Federico II"

Nome e Cognome: Matricola:

1	2	3	4	tot
/36	/27	/22	/15	/100

1. *6*6 = 36 punti*

   Si supponga che l'output tipico del comando "ls -al" sia:

```
drwx------  2 utente staff     4096 2007-02-10 14:28 .
drwx------  2 utente staff     4096 2007-02-10 14:28 ..
lrwxrwxrwx  1 utente staff      102 2007-01-11 02:05 new.gif -> /pics/new.gif
-rw-r--r--  1 utente staff   319045 2007-01-16 01:48 reti8.pdf
lrwxrwxrwx  1 utente staff       11 2007-05-02 12:10 .bashrc -> /etc/bashrc
--w-------  1 utente staff      268 2007-04-26 10:37 .screenrc
drwxr-xr-x  4 utente staff     4096 2006-08-28 19:29 .xmms
```

   Si scriva un comando di shell UNIX per ciascuno dei seguenti compiti[1]:

   (a) Elencare i file per i quali né il gruppo di appartenenza né gli altri utenti hanno alcun permesso.

   (b) Elencare i file regolari il cui nome comincia con ".x" o con ".s".

   (c) Elencare i link simbolici contenuti nella directory corrente, specificando per ognuno di essi il file di destinazione, ed i file regolari a cui l'utente proprietario ha accesso in lettura. Esempio di output del comando:

```
new.gif è un link simbolico che punta a /pics/new.gif
reti8.pdf è un file regfolare a cui il proprietario puo' accedere in lettura
.bashrc è un link simbolico che punta a /etc/bashrc
```

   (d) Elencare tutti i file di proprietà di "utente" la cui data di ultima modifica è compresa tra il primo giugno 2007 e il 30 settembre 2007.

   Supponendo che la directory corrente contenga esattamente i file mostrati sopra, si determini l'output dei seguenti comandi:

   (e) `ls | sed 's/old/new/g' | grep 'e.*[[:digit:]]'`

   (f) `ls -al | awk '{ print $6 }' | sed 's/-.*-..//' | sort | uniq`

2. *27 punti*
   Si realizzi un script "rimuovi" che accetta come argomento un'estensione di file ed effettui le seguenti operazioni:

   - conti il numero di file che hanno questa estensione;
   - calcoli la dimensione totale dei file con questa estensione;
   - elenchi all'utente i nomi dei file (privati dell'estensione) la cui dimensione è maggiore di 1000 byte;
   - chieda all'utente conferma per la rimozione dei file di dimensione maggiore;
   - cancelli tutti i file se l'utente risponde affermativamente.

   Esempio di output dello script:

[1] Per "comando", si intende qualunque istruzione impartita al prompt della shell, che non contenga strutture di controllo, uso di variabili, o l'operatore ";".

```
$ ./rimuovi pdf
I 30 file pdf occupano in totale 789345 byte
I seguenti file pdf hanno dimensione maggiore di 1000:
uno due tre quattro cinque
Vuoi che li cancelli? (s/N) s
Ho cancellato 5 file
```

3. *22 punti*
   Usando soltanto le system call di I/O di basso livello, si implementi un programma C chiamato "split", che prende come argomento il nome di un file x ed un numero intero n, e divide il file x in due file chiamati "part1" e "part2" che contengono rispettivamente i primi n caratteri del file x e i restanti caratteri.

   Ad esempio, il comando "split pippo 1500" crea il file part1 in cui inserisce i primi 1500 caratteri del file pippo, e il file part2 in cui inserisce i restanti caratteri del file pippo. Se pippo contiene meno di 1500 caratteri, il file part1 conterrà tutto il contenuto di pippo, mentre il file part2 non verrà nemmeno creato.

4. *15 punti*
   La system call `raise` manda un segnale al processo corrente. Ovvero, `raise(SIGINT)` è equivalente a `kill(getpid(), SIGINT)`.
   Determinare il contenuto del file prova.txt dopo l'esecuzione del seguente programma, supponendo che il file non esistesse prima dell'esecuzione. Ignorare la mancanza delle direttive `#include`.

```
void foo(int s) {
  lseek(4, -2, SEEK_CUR);
  write(4, "aaa\n", 4);
  close(4);
}

int main(void) {
  int fd = open("prova.txt", O_RDWR | O_CREAT, S_IRUSR | S_IWUSR);
  dup2(fd, 4);
  signal(SIGUSR1, foo);

  write(STDOUT_FILENO, "inizio", 6);
  write(fd, "bbbb", 4);
  raise(SIGUSR1);
  write(fd, "ccc\n", 4);
  write(STDOUT_FILENO, "fine\n", 5);
  return 0;
}
```

Lab. di Sistemi Operativi, Gruppo I: M. Faella **Anno Acc. 2006-2007**

Esame del 9 luglio 2007

Università di Napoli "Federico II"

Nome e Cognome: Matricola:

1	2	3	4	tot
/33	/22	/31	/14	/100

1. *33 punti*
   Si supponga che l'output tipico del comando "ls -al" sia:

```
-rw-------  2 utente1 staff     4096 2007-02-10 14:28 doc1.pdf
-rw-r--r--  1 utente1 staff      102 2007-01-11 02:05 doc2.pdf
-rw-r--r--  1 utente2 users     3145 2007-01-16 01:48 rel.tex
-rw-------  1 utente2 users      268 2007-04-26 10:37 song1.mp3
-rw-r--r--  4 utente3 staff     4096 2006-08-28 19:29 text1.tex
-rw-r--r--  1 utente3 staff       11 2007-05-02 12:10 text2.tex
```

   Si scriva un comando di shell UNIX per ciascuno dei seguenti compiti[1]:

   (a) *(5 punti)* Elencare tutti i gruppi che posseggono almeno un file nella directory, eliminando i duplicati.

   (b) *(8 punti)* Elencare soltanto i nomi di tutti i file regolari che risultino eseguibili da qualcuno (sia esso l'utente, il suo gruppo o tutti) e che abbiano un nome senza estensione.

   (c) *(8 punti)* Contare le occorrenze del carattere "c" all'interno del file "rel.tex".

   Supponendo che la directory corrente contenga esattamente i file mostrati sopra, si determini l'output dei seguenti comandi:

   (d) *(6 punti)* `ls | sed 's/ex/ax/' | grep 'ta'`

   (e) *(6 punti)* `ls -al | awk '{ print $7 }' | sed 's/:..//' | sort | uniq`

2. *22 punti*
   Si realizzi uno script che elenchi il tipo di file regolari contenuti nella directory classificandoli tramite l'estensione ed elenchi per ogni estensione i gruppi che possiedono almeno un file con tale estensione e il numero di file da essi posseduti.

   Ad esempio, se la directory corrente contiene esattamente i file mostrati nell'Esercizio 1, lo script deve produrre il seguente output:

```
.tex: users 1 staff 2
.pdf: staff 2
.mp3: users 1
```

3. *31 punti*
   Usando soltanto le system call di I/O di basso livello e la funzione `strcmp`, si implementi un programma C chiamato "print_constants", che prende come argomento il nome di un file sorgente in C e stampa il nome di tutte le costanti simboliche (definite con `#define`) definite in quel file.

4. *14 punti*
   La system call `raise` manda un segnale al processo corrente. Ovvero, `raise(SIGINT)` è equivalente a `kill(getpid(), SIGINT)`.
   Dato il seguente programma:

[1] Per "comando", si intende qualunque istruzione impartita al prompt della shell, che non contenga strutture di controllo, uso di variabili, o l'operatore ";".

```
void foo(int s) {
  char c;
  lseek(4, -4, SEEK_CUR);
  read(4, &c, 1);
  write(STDOUT_FILENO, &c, 1);
  close(4);
  return;
}

int main(void) {
  int fd = open("prova.txt", O_RDONLY);
  char buf[10];

  dup2(fd, 4);
  signal(SIGUSR1, foo);

  read(fd, buf, 2);
  read(4, &buf[2], 2);
  if (buf[2] == 'a')
    raise(SIGUSR1);
  else
    write(STDOUT_FILENO, &buf[2], 1);

  printf("\n%d\n", read(fd, buf, 10));
  close(fd);
  return 0;
}
```

(a) *(7 punti)* Determinare l'output del programma supponendo che il contenuto del file "prova.txt" sia (4 caratteri):

```
ciao
```

(b) *(7 punti)* Determinare l'output del programma supponendo che il contenuto del file "prova.txt" sia:

```
carta
```