Caselle riservate

Ex. 1	
Ex. 2	
Ex. 3	
Ex. 4	
Ex. 5	
Ex. 6	
Tot.	

Sistemi Operativi

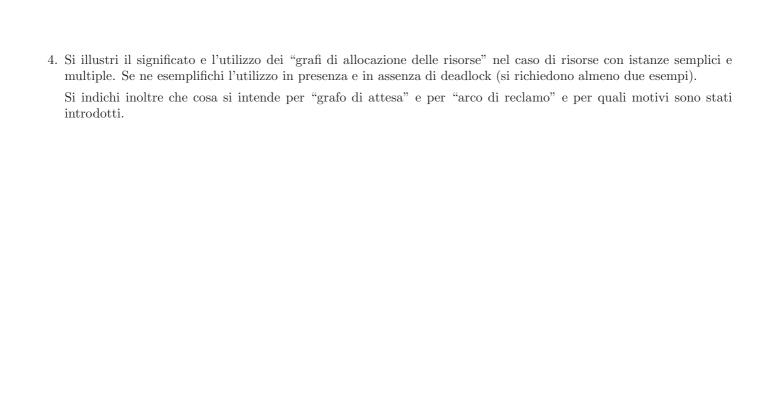
Compito d'esame 19 Giugno 2014

Matricola	. Cognome		Nome	
	Docente:	○ Laface	O Quer	
Non si possono consultare to oggetto di valutazione. Durata della prova: 75 minu	, 11	calcolatrici. Ripor	rtare i passaggi pri	ncipali. L'ordine sarà

1. Si descriva la struttura denominata stat per una directory entry. Si scriva inoltre un programma, in linguaggio C, in grado di ricevere due parametri sulla riga di comando e di copiare l'albero di direttori specificato dal primo parametro nell'albero di direttori specificato dal secondo parametro.

2.	. Si descriva l'utilizzo dei <i>segnali</i> nel sistema operativo UNIX/Linux con relativi vantaggi e svantaggi. Si descri in particolare le system call signal, kill, pause e alarm, riportando un esempio completo di utilizzo. Ind le differenze tra alarm e sleep.	vano icare

3. Si indichino le principali differenze tra processi e thread. Si descriva la gestione di tali entità da parte del sist operativo (struttura in memoria, etc.). Se ne indichino le caratteristiche principali e i relativi vantaggi e svanta Si indichino inoltre le principali differenze tra thread a livello utente e thread a livello kernel, descrivendo i modelli di thread normalmente disponibili.	aggi.



5. Uno script di shell riceve sulla riga di comando due parametri: il nome di un file e quello di un direttorio. Il file include un insieme di righe ciascuna contenente due stringhe, il nome di un direttorio e il nome di un file, separate da spazi. Il seguente è un esempio corretto di file:

```
/home/bin/ dos2unix.sh
/usr/bin/ old.c
/usr/bin/ new.c
```

Lo script deve ricercare tra i file elencati (e.g., /home/bin/dos2unix.sh, /usr/bin/old.c, etc.) quelli che contengono la stringa main (si osservi che la stringa main deve essere presente nel contenuto del file *non* nel suo nome) e per ognuno di questi:

- Chiedere all'utente se vuole effettuarne una copia.
- Leggere la risposta dell'utente da tastiera (del tipo si/no).
- In caso affermativo, copiare tale file nel direttorio specificato sulla riga di comando. In caso negativo, procedere con il file successivo.

Si osservi che all'interno di un ciclo di lettura da file tramite comando read, una ulteriore lettura da tastiera con lo stesso comando, può essere effettuata come read nomeVariabile </dev/tty.

6. In un programma concorrente tutti i thread producono il proprio output sullo stesso file. Al testo memorizzato dal processo principale, ogni thread secondario aggiunge il proprio inserendo il testo tra parentesi graffe e il proprio identificatore di thread tra parentesi quadre. Il successivo è un esempio di output generato in tale contesto:

```
... testo 1 (processo principale) ...
[123]{testo prodotto dal thread 123}
... testo 2 (processo principale)
ancora il processo principale ...
[245]{testo prodotto dal thread 245}
... testo 3 (processo principale) ...
[123]{...}
```

Si desidera scrivere uno script AWK in grado di separare l'output dei vari thread da quello del processo principale. Il nome del file che occorre parsificare viene passato allo script sulla riga di comando. Il file di nome "tid.txt" contiene l'elenco dei thread e il nome dei file in cui il relativo output deve essere memorizzato, con il seguente formato:

```
tid1 fileThread1
tid2 fileThread2
```

dove tid1, tid2, etc., sono numeri interi (e.g., 123, 245, etc.). L'output del processo principale deve essere memorizzato in un file con lo stesso nome di quello di ingresso ma con estensione ''log''.