Programmazione di Sistema
30 gennaio 2018 (teoria)

Si prega di rispondere in maniera leggibile, descrivendo i passaggi e i risultati intermedi. Non è possibile consultare alcun materiale. Durata della prova 70 minuti. Sufficienza con punteggio >= 8. Prima e seconda parte possono essere sostenute in appelli diversi. La presenza a una delle due parti annulla automaticamente un'eventuale sufficienza già ottenuta (per la stessa parte): viene intesa come rifiuto del voto precedente.

1. *(3 punti)* Si consideri la seguente sequenza di riferimenti in memoria nel caso di un programma di 1000 parole: 171, 219, 165, 401, 874, 746, 312, 423, 260, 225, 78, 954.
   Si determini la stringa dei riferimenti a pagine, supponendo che la loro dimensione sia di 200 parole. Supponendo di utilizzare un algoritmo di sostituzione pagine di tipo second-chance (con limite di 3 frame disponibili), determinare quali e quanti page fault (accessi a pagine non presenti nel resident set) si verificheranno (è richiesta la visualizzazione del resident set dopo ogni riferimento). Si supponga che il bit di riferimento di una pagina venga inizializzato a 0 in corrispondenza a un page-fault.

2. *(4 punti)* Sia dato un file system basato su File Allocation Table (FAT). Si supponga, per semplicità descrittiva, che il file system contenga unicamente 20 blocchi di 4096 Byte (numerati da 0 a 19), nel quale siano immagazzinati 3 file (`"a.dat", "b.bmp", "c.exe"`), il cui primo blocco è, rispettivamente, il 7, il 4 e il 18. La FAT contiene i valori rappresentati in figura

5	13	10	8	9	-1	14	11	-1	0	-1	3	2	-1	12	6	1	16	17	15
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

   Si determinino i blocchi appartenenti a ognuno dei tre file. Si dica poi quanti e quali blocchi (in ordine) contiene la "free list". Si rappresenti infine la FAT dopo aver cancellato il file `b.bmp`.

3. *(3 punti)* Si descrivano brevemente, nell'ambito di un file system, i file organizzati come:
   - Sequenza di byte
   - Struttura a record
   - Struttura complessa, ad esempio file con indice.

   Di ognuno dei tipi di organizzazione, si dica se consente l'accesso sequenziale e/o quello diretto.
   Si dica poi se, in un sistema di tipo unix, quali delle organizzazioni possono (o non possono) essere realizzate mediante inode (motivare la risposta).

4. *(4 punti)* Sia dato un sistema operativo OS161. Si spieghi perché, in un contesto multi-core (sono presenti più CPU) non è possibile realizzare la mutua esclusione semplicemente disabilitando e riabilitando l'interrupt. Dato codice (ridotto alle parti essenziali) delle funzioni di semaforo P e V, riportate in seguito

```
void P(struct semaphore *sem) {
   spinlock_acquire(&sem->sem_lock);
   while (sem->sem_count == 0) {
     wchan_sleep(sem->sem_wchan, &sem->sem_lock);
   }
   sem->sem_count--;
   spinlock_release(&sem->sem_lock);
}
void V(struct semaphore *sem) {
   spinlock_acquire(&sem->sem_lock);
   sem->sem_count++;
   KASSERT(sem->sem_count > 0);
   wchan_wakeone(sem->sem_wchan, &sem->sem_lock);
   spinlock_release(&sem->sem_lock);
}
```

   Spiegarne brevemente il funzionamento. In particolare, si dica:
   - a cosa serve lo spinlock (in entrambe le funzioni)?
   - perché la P contiene un ciclo while anziché un `if (sem->sem_count == 0)`?
   - `Perché la wchan_sleep` riceve come parametro lo spinlock?
   - E' possibile che la chiamata alla `wchan_wakeone` svegli più di un semaforo in attesa su wait_channel?

   Spiegare la differenza tra condition variable e wait_channel.
   Spiegare brevemente perché gli argomenti al main ricevuti all'attivazione di un processo utente dalla funzione kernel `cmd_progthread(void *ptr, unsigned long nargs)`, non possono essere copiati direttamente nei parametri `argv e argc` del programma user.