# Università degli Studi di Napoli Federico II Corso di Laurea in Ingegneria Informatica Esame di Sistemi Operativi Proff. Cinque, Cotroneo, Natella

### COMPITO SVOLTO

Lo studente completi il programma a corredo di questo documento, seguendo le seguenti indicazioni.

La prova sarà valutata come segue:

- A: Prova svolta correttamente.
- **B**: Il programma non esegue correttamente, con errori minori di programmazione o di concorrenza.
- C: Il programma non esegue correttamente, con errori significativi (voto max: 22).
- INSUFFICIENTE: Il programma non compila o non esegue, con errori gravi di sincronizzazione.

### Istruzioni per la consegna dell'elaborato

L'elaborato dovrà essere svolto in una cartella dal nome: Cognome Nome Matricola Docente

#### Esempio:

In alternativa, è consentito creare il file compresso "tar" tramite l'interfaccia grafica.



All'interno della macchina virtuale, aprire il browser all'indirizzo: **TBD** 

Nel form, cliccare su "aggiungi file", selezionare il file compresso contenente il proprio svolgimento, e indicare il proprio nome ed email @studenti.unina.it.

Attendere una notifica del docente, e quindi scollegarsi dalla piattaforma di VirtualClassroom.

# Testo della prova

Si realizzi in linguaggio C/C++ la seguente applicazione multiprocesso basata su code di messaggi UNIX.

La applicazione simula l'aggiornamento di un tabellone di un aeroporto per le partenze e per gli arrivi.

```
typedef struct {
   /*long tipo;
   int ID;
   char direzione [10];
   char citta_partenza [20];
   char citta_arrivo [20];
}Volo;
```

Un processo **Controllore** gestisce i voli in arrivo e in partenza inviando un messaggio su una sola **coda di messaggi sincrona** verso un processo **Proxy**. Ogni qual volta il processo proxy riceve un volo da gestire, invia un messaggio a 2 processi, **Gestore Arrivi** e **Gestore Partenze** che gestiscono rispettivamente gli arrivi e le partenze. La comunicazione tra il proxy e i processi gestori deve avvenire tramite una sola **coda di messaggi asincrona**. Il *Proxy* invierà un messaggio (attendendo 1 secondo tra un messaggio e il successivo) al *Gestore Arrivi* ogni qual volta riceve un volo che contiene la stringa "arrivo" nel campo direzione; viceversa, se il processo Proxy avrà ricevuto un volo contenente la stringa "partenza" nel campo direzione invierà il messaggio al *Gestore Partenze*. I processi gestori una volta ricevuto il messaggio potranno stampare a video tutte le informazioni relative al volo ricevuto.

Si crei un unico eseguibile, in cui un processo padre avvia tutti i processi necessari. Il processo Controllore dovrà gestire 10 voli (5 partenze e 5 arrivi). Il processo padre attende che tutti i processi abbiano terminato l'esecuzione per poi terminare a sua volta.

