Semafori

- Le operazioni sui semafori (in notazione ad oggetti) sono S.P() e S.V(). Non hanno alcun parametro ne' alcun valore di ritorno. Se invece volete usare la notazione procedurale le operazioni sono P(S) e V(S), unico parametro e' il semaforo e nessun valore di ritorno.
- Quando si usano semafori tutti gli accessi ai dati condivisi devono avvenire in mutua esclusione
- Le code dei semafori e il valore dei semafori sono strutture e variabili private gestite dall'implementazione del paradigma e non sono accessibili
- Se non altrimenti specificato i semafori sono generali e fair (FIFO)
- I semafori *Devono* avere un valore iniziale.
- E' sicuramente errato un programma che usa un semaforo solo con operazioni P e nessuna V o viceversa.

Monitor

- Le operazioni definite sulle variabili di condizione sono c.wait() e c.signal(). Non hanno parametri, ne' valore di ritorno.
- Bloccarsi (Busy Wait) all'interno di un monitor è deadlock
- Le code delle variabili di condizione, lo urgent stack sono strutture private gestite dall'implementazione del paradigma e non sono accessibili
- La routine di inizializzazione di un monitor non può contenere wait e signal
- se non altrimenti specificato la politica delle variabili di condizione è signal-urgent
- E' sicuramente errato un programma che su una variabile condizione effettua solo wait e mai signal. (viceversa la condizione e' inutile).
- E' sicuramente errato un programma che ha una wait come prima istruzione di ogni procedure entry

Message Passing

- Esistono tre paradigmi: sincrono, asincrono, completamente asincrono. Si usa quello indicato dall'esercizio.
- Se il testo indica "message passing asincrono" allora NON è "completamente asincrono"
- E' sicuramente errato un programma che usa solo send e nessuna receive o che usa solo receive e nessuna send
- Se non e' indicato diversamente i servizi di message passing sono FIFO
- Message Passing e' un paradigma a memoria privata: non possono esistere variabili condivise fra i processi.

In generale

- Chiamate di fantasia quali block, wakeup, restart non devono mai comparire in soluzioni di esercizi
- Semafori, monitor, message passing sono paradigmi indipendenti. Gli esercizi indicano quale usare. Non si possono *mischiare* primitive di paradigmi diversi.
- E' sicuramente errato un programma che usa variabili prima di assegnare loro un valore iniziale
- Soluzioni con semafori, monitor o message passing non devono contenere busy-wait (sono stati creati per evitarlo)
- E' possibile definire strutture dati senza implementarle a patto che (1) non contengano primitive concorrenti (sincronizzazione o comunicazione) (2) non siano miracolose, devono poter essere implementabili quindi i parametri devono contenere tutte le informazioni necessarie per fornire i servizi richiesti.