Nome_____ Cognome _____N. di matricola (10 cifre) _____ Riga ____ Col ____

UNIVERSITA' DEGLI STUDI DI BOLOGNA - CORSO DI LAUREA IN INFORMATICA
PROVA SCRITTA DI SISTEMI OPERATIVI
ANNO ACCADEMICO 2022/2023
19 luglio 2023

Esercizio -1: Essere iscritti su AlmaEsami per svolgere questa prova.

Esercizio 0: Scrivere correttamente nome, cognome, matricola e posizione in tutti i fogli prima di svolgere ogni altro esercizio. Scrivere esclusivamente a penna senza abrasioni. E' vietato l'uso delle penne cancellabili, della matita, dei coprenti bianchi per la correzione (bianchetto) e la scrittura in colore rosso (riservato alla correzione).
Il compito e' formato da tre fogli, sei facciate compresa questa. Le soluzioni che si vogliono sottoporre per la correzione devono essere scritte negli spazi bianchi di questi fogli. Non verranno corretti altri supporti.
E' obbligatorio consegnare il compito, e' possibile chiedere che esso non venga valutato scrivendo “NON VALUTARE” in modo ben visibile nella prima facciata.
Per svolgere questo compito occorre solo una penna e un documento di identità valido. La consultazione o anche solo la disponibilità di altro materiale comporterà l'annullamento del compito (verrà automaticamente valutato gravemente insufficiente).

Esercizio c.1: Scrivere il monitor sv con una sola funzione *entry* syncvalue che ha la seguente dichiarazione:
procedure entry int syncvalue(int key);
I processi che chiamano la syncvalue si bloccano sempre. Quando il valore del parametro key è diverso da quello della precedente chiamata il processo prima di bloccarsi riattiva tutti i processi in attesa. Il valore di ritorno è il numero di processi con lo stesso valore key sbloccati. Per esempio:
P chiama sv.syncvalue(42), si blocca.
Q chiama sv.syncvalue(42), si blocca.
R chiama sv.syncvalue(44) sblocca P e Q poi si blocca. Il valore di ritorno per P e Q è 2.
T chiama sv.syncvalue(46), sblocca R che ritorna 1 e si blocca.
Q chiama sv.syncvalue(46), si blocca.
P chiama sv.syncvalue(46), si blocca
V chiama sv.syncvalue(0), sblocca T, Q e P (valore di ritorno: 3) poi si blocca...

Esercizio c.2: Un servizio di message passing universale supporta tutti i modelli di message passing: sincrono, asincrono e completamente asincrono. Il funzionamento può essere deciso ad ogni spedizione o ricezione. La API prevede due funzioni:
void usend(bool blocking, msg_t msg, pid_t dest)
msg_t urecv(bool blocking, pid_t sender)
le funzioni sono bloccanti o meno a seconda del valore del parametro blocking.
Dato un servizio di message passing asincrono scrivere un servizio di message passing universale senza fare uso di processi server;.

Esercizio g.1: In un sistema monoprocessore lo scheduler di CPU usa l'algoritmo round robin con time slice di 4 ms. Nel sistema c'è un disco rotazionale D che usa l'algoritmo dell'ascensore (LOOK). La testina inizialmente si trova nel cilindro 0, l'operazione di seek ha una velocità di 1000 cilindri al secondo (i.e. 1 cilindro al ms) e ogni operazione di input o output sul cilindro corrente impiega 2 ms. In questo sistema sono in esecuzione i processi P1, P2, P3, P4 (che sono nella ready queue in questo ordine).
P1: 3ms calcolo, lettura da D al cilindro 3, 2 ms calcolo, lettura D cilindro 6, 1ms calcolo.
P2: 5ms calcolo, lettura D cilindro 6, 1ms calcolo
P3: 2ms calcolo, scrittura D cilindro 6, 1ms calcolo
P4: 1ms calcolo, lettura D cilindro 2, 1 ms calcolo

Esercizio g.2: rispondere alle seguenti domande (*motivando opportunamente le risposte!*):
a) Quando un sistema è in trashing il carico della CPU (o delle CPU) è alto o basso, perché?
b) poniamo per ipotesi che venga creato un nuovo processore csirv con un suo nuovo instruction set. Quali sono i passi da fare per portare Linux alla nuova architettura?
c) Trovare, se possibile, un grafo di holt che abbia 6 nodi, tutte le risorse abbiano molteplicità due (due unità per ogni classe di risorse) e tutti i processi siano in deadlock.
d) A cosa serve il meccanismo challenge/response nella gestione delle password? Come funziona praticamente?