

6i Esercizi (laboratorio lez 13)

Progr. Concorrente e Vari ONLINE

consiglio: installare e provare editor leggero meno buggato di gedit: **geany**
sudo apt-get update ; sudo apt-get install geany

Contenuti:

ESERCIZI SIMILI A QUELLI DELL'ESAME PROVA PRATICA

Esercizio 1061	ONLINE Programmazione concorrente	"soccorsi"
Esercizio 1063	script bash	"contaseparatamente.sh"
Soluzioni esercizi 1061 ONLINE, 1063		

ESERCIZI vari, per casa

Esercizio 64	- Programmazione Concorrente	"conigli"
Esercizio 1062	- Programmazione concorrente	"strumenti"
Esercizio 65	- script , stdout, stdin, ridirezionamenti, tail	
Soluzioni esercizi 64, 65		

Materiali per Prova Pratica

Download Materiale:

Scaricare il file con le **dispense** e gli **esempi** svolti a lezione

```
wget http://www.cs.unibo.it/~ghini/didattica/TREE4OS2021.tgz
```

Decomprimere l'archivio scaricato: `tar xvzf TREE4OS2021.tgz`

Viene creata una directory **TREE4OS2021** con dentro una sottodirectory **sistemioperativi** con dentro tutto il **materiale**.

Potete navigare tra il materiale con un normale browser aprendo l' URL

```
file:///home/studente/VOSTRADIRECTORY/TREE4OS2021/sistemioperativi/dispens  
eSistOp1617.html
```

o anche

```
file:///home/studente/VOSTRADIRECTORY/TREE4OS2021/sistemioperativi/index.ht  
ml
```

Esercizio 1061ONLINE - soccorsi (1)

Una scala in un'università è un po' malridotta.

Ogni professore che passa inciampa e si rompe un piede, cominciando ad urlare imprecazioni irriveribili. Tanto è lo strepitare che nasce il dubbio se si tratti di un ferimento o di una possessione demoniaca.

Nel dubbio, accorrono un medico ed un esorcista. I soccorsi al ferito possono essere effettuati solo dopo che entrambi (medico ed esorcista) hanno raggiunto il malcapitato professore.

L'intervento dei due soccorritori dura 2 secondi a partire da quando sono entrambi arrivati dal professore. Dopo che entrambi hanno terminato l'intervento, i due soccorritori se ne vanno e anche il professore si rialza e se ne va.

Poi il professore fa 4 secondi di lezione, poi ripassa sulla scala e cade di nuovo, e così via all'infinito.

Ci sono un medico, un esorcista e un professore.

Modellare ed implementare il sistema descritto, utilizzando dei thread POSIX per ciascuna figura (professore, esorcista, medico) ed avvalendosi delle opportune strutture dati per la sincronizzazione. Utilizzare come base i files forniti: **soccorsi.c** **DBGpthread.*** **prnterror.h**

Scrivere il Makefile per compilare e linkare i sorgenti. La mancanza del Makefile viene considerato un errore grave.

Occorre inserire il controllo di errore nelle chiamate a funzione delle librerie dei pthread. In caso di errore grave, terminare il programma producendo un avviso a video.

VEDERE DETTAGLI SULL'IMPLEMENTAZIONE DELLA BASE NELLA SUCCESSIVA PAGINA

Esercizio 1061ONLINE - soccorsi (2)

CONTINUAZIONE

Nella base fornita, l'implementazione del thread Professore è già completa.

Le variabili globali già inserite sono inizializzate nel main.

Potete aggiungere variabili globali, se vi servono.

Ricordate di inizializzare nel main le vostre variabili globali.

I thread Esorcista e Medico fanno operazioni simili, in pratica si distinguono solo per il nome.

L'implementazione dei thread Esorcista e Medico è perciò fatta chiamando una stessa funzione Soccorritore a cui vengono passati una stringa col nome del Soccorritore ed un valore 0 o 1 rispettivamente per l'Esorcista ed il Medico.

SOLUZIONI

http://www.cs.unibo.it/~ghini/didattica/sistemioperativi/ESERCIZI/es1061ONLINE_soccorsi.tgz

Esercizio 1063 - **contaseparatamente.sh**

Realizzare uno script bash **contaseparatamente.sh** che effettua le seguenti operazioni:

Lo script prende un numero variabile di argomenti, al massimo 9.

Ciascun argomento è il nome di un file.

Lo script scrive sullo **standard output** il numero totale di righe dei file passati come argomenti di indice pari.

Lo script scrive sullo **standard error** il numero totale di righe dei file passati come argomenti di indice dispari.

Realizzare poi uno script bash **lancia.sh** che:

lancia l'esecuzione dello script `contaseparatamente.sh` passando allo script come argomenti i primi 7 file che si ottengono come standard output del comando `ls -S1 /usr/include/*.h`

Il modo in cui lo script `lancia.sh` lancia l'esecuzione di `contaseparatamente.sh` deve ridirezionare sia lo standard output che lo standard error di `contaseparatamente.sh` sullo standard error di `lancia.sh` stesso.

Mi raccomando, verificare che lo script `lancia.sh` produca il suo output sullo standard error.

Soluzione Esercizio 1063 `es1063_contaseparatamente.sh`

http://www.cs.unibo.it/~ghini/didattica/sistemioperativi/ESERCIZI/es1063_contaseparatamente.tgz

da qui in avanti altri esercizi per casa

Esercizio 64 - figliarecomeconigli.c

In un programma non devono mai essere contemporaneamente in esecuzione più di 5 thread (circa) , più il main.

Il main di un programma crea un nuovo thread ogni 5 secondi.

Ciascun thread appena creato stampa a video il numero di thread presenti.

Ciascun thread figlio cerca di creare ogni secondo un nuovo thread nipote, ma se non può crearlo perché ci sarebbero troppi thread allora quel thread figlio termina.

Ciascun thread nipote cerca di creare e così via ...

Modellare ed implementare il sistema descritto, utilizzando dei thread POSIX ed avvalendosi delle opportune strutture dati per la sincronizzazione.

Scrivere il Makefile per compilare e linkare i sorgenti. La mancanza del Makefile viene considerato un errore grave.

Occorre inserire il controllo di errore nelle chiamate a funzione delle librerie dei pthread. In caso di errore grave, terminare il programma producendo un avviso a video.

Soluzione Esercizio 64 es64_figliarecomeconigli.c

http://www.cs.unibo.it/~ghini/didattica/sistemioperativi/ESERCIZI/es64_figliarecomeconigli.tg

Esercizio 1062ONLINE - strumenti

In una fabbrica ci sono diversi tipi di strumenti da usare, in particolare:

2 strumenti di tipo S1, 4 strumenti di tipo S2 e 2 strumenti di tipo S3.

Nella fabbrica si effettuano due tipi di lavorazioni, L1 e L2. La lavorazione L1 impiega 5 secondi, la lavorazione L2 solo 4 secondi. Una volta cominciata, la lavorazione non può essere sospesa.

In ciascuna lavorazione si usano alcuni strumenti per alcuni secondi, come specificato nella tabella.

Nella tabella la prima colonna da sinistra indica la lavorazione, la seconda colonna da colonna indica uno strumento usato in quella lavorazione, le successive colonne indicano i secondi in cui lo strumento di quella riga viene usato nella lavorazione a cui la riga appartiene.

Ad esempio, alla lavorazione L1 serve lo strumento S1 nei secondi 1, 2, 3, 4 e 5, inoltre serve lo strumento S2 nei secondi 2 e 3. Lo strumento S3 serve nei secondi 4 e 5.

Uno strumento può ovviamente essere usato da una sola lavorazione alla volta. Al termine di ciascuna lavorazione, la lavorazione rilascia tutti gli strumenti utilizzati, aspetta 1 secondo e poi ricomincia dall'inizio. Esistono 2 lavorazioni L1 e 2 lavorazioni L2.

	sec	1	2	3	4	5
L1	S1	1	1	1	1	1
	S2		1	1		
	S3				1	1
L2	S1				1	
	S2		1	1		
	S3	1	1	1	1	

Modellare ed implementare il sistema descritto, utilizzando dei thread POSIX per ciascuna

figura (lavorazioni L1 e L2) ed avvalendosi delle opportune strutture dati per la sincronizzazione. **Stabilire come assegnare gli strumenti alle lavorazioni, anche in modo non ottimo, ma adatto ad evitare deadlock.**

Scrivere il Makefile per compilare e linkare i sorgenti. La mancanza del Makefile viene considerato un errore grave. Occorre inserire il controllo di errore nelle chiamate a funzione delle librerie dei pthread. In caso di errore grave, terminare il programma producendo un avviso a video.

SOLUZIONI

Soluzione Esercizio 1062 strumenti

http://www.cs.unibo.it/~ghini/didattica/sistemioperativi/ESERCIZI/es1062_strumenti_complicato.tgz

Esercizio 65 - ridirezionamentocircolare.sh

Scrivere gli script **A.sh** **B.sh** **lancia.sh** ed altri se occorrono.

Lo script **A.sh** comincia scrivendo il valore 1 sul proprio standard output.

Successivamente lo script **A.sh** riceve continuamente sul proprio standard input delle righe di testo che contengono un valore numerico.

Per ciascuna riga ricevuta, lo script **A.sh** legge il valore numerico, lo incrementa di 1, attende 1 secondo, e scrive il valore incrementato sullo standard output.

Lo script **B.sh** riceve continuamente sul proprio standard input delle righe di testo che contengono un valore numerico.

Per ciascuna riga ricevuta, lo script **B.sh** legge il valore numerico e lo scrive sul proprio standard output senza incrementarlo. Inoltre, lo scrive anche in un file **B.txt**

Lo script **lancia.sh** deve lanciare i due script **A.sh** e **B.sh** (ed eventualmente altri script, se necessario) in modo tale che:

- l'output di **A.sh** vada a finire nello standard input di **B.sh**
- l'output di **B.sh** vada a finire nello standard input di **A.sh**

In tal modo, **A.sh** e **B.sh** continueranno a passarsi quel valore incrementandolo una volta sì e una no.

Soluzione Esercizio 65 **es65_ridirezionamentocircolare**

http://www.cs.unibo.it/~ghini/didattica/sistemioperativi/ESERCIZI/es65_ridirezionamentocircolare.tgz