Prova Pratica 020
turno 1 gruppo 2

2017 febbraio 20

i file da consegnare **devono** essere collocati nella directory **CONSEGNA** dentro la home directory dell'utente studente.

Prova Pratica 020 - turno 1 gruppo 2

Download Materiale:

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

wget http://esameso.csr.unibo.it/TREE4OS1617.tgz

Decomprimere l'archivio scaricato: tar xvzf TREE4OS1617.tgz

Viene creata una directory **TREE4OS1617** 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/TREE4OS1617/sistemioperativi/dispenseSistOp1617.html

Esercizi d'esame: per chi ha difficoltà a superare la prova pratica, ho previsto due tipi di prove:

A. **una prova COMPLICATA**, e' la modalità normale che vi permette di raggiungere un **voto massimo** (nella prova pratica stessa) di **30Lode** ,
B. ed **una prova SEMPLICE**, un po' **meno complicata**, che però vi permette di raggiungere un **voto massimo di 24** perché l'esercizio di programmazione concorrente é meno difficile.

Scegliete voi quale prova svolgere in funzione della vostra preparazione.

La prova COMPLICATA è composta dagli esercizi **55 e 56**,

La prova SEMPLICE è composta dagli esercizi **54 e 56**.

Come vedere l'esercizio 56 è comune alle due prove.

Svolgete SOLO gli esercizi della prova che vi interessa.

I file da consegnare **devono** essere collocati nella directory **CONSEGNA** dentro la home directory dell'utente studente.

Esercizio Esame Pratica **- 54 - notifica (semplice)**

Un programma crea inizialmente **10** thread **notifica**, ciascuno dei quali creerà un thread figlio di tipo **notifica**, il quale, a sua volta creerà un thread figlio **notifica** e così via all'infinito.
Il main deve continuamente rimanere in attesa di ricevere da ciascuno dei thread **notifica** in esecuzione il thread identifier di quel thread e poi deve chiamare la join per quel thread.

Più nel dettaglio, ciascun thread di tipo **notifica** deve essere joinable e :

- appena inizia deve aspettare 1 secondo,
- poi deve passare al main il proprio thread identifier,
- poi deve creare un figlio,
- poi deve terminare.

NB: attenti a non perdersi dei thread identifier!

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.

Esercizio Esame Pratica **- 55 - alieni (complicato)**

Un gruppo di **5** alieni collaborativi ma claustrofobici ha dei problemi di spazio sul pianeta Zog, così occupa un piccolissimo monolocale.

La casa può contenere al massimo 2 alieni e all'inizio è vuota.
Per mantenere l'occupazione è indispensabile che **in ogni istante ci sia almeno un alieno in casa**.
Gli alieni aspettano di entrare in casa **in un gruppetto** fuori dalla casa.
A causa della claustrofobia, gli alieni cominciano a morire appena entrati in casa, così **cercano di uscire appena possibile** dalla casa, **nell'ordine con cui sono entrati**, per morire all'aperto.
Subito dopo essere uscito di casa, ciascun alieno controlla il numero di alieni in attesa fuori di casa. Se l'alieno morente si accorge che ci sono meno di 3 alieni in attesa fuori casa, l'alieno crea altri 2 (thread) alieni e poi muore.

Modellare ed implementare il sistema descritto, utilizzando dei thread POSIX per ciascuna figura (alieno) ed avvalendosi delle opportune strutture dati per la sincronizzazione.
Il programma crea i 5 thread alieni iniziali.
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.

PRECISAZIONE sul prossimo Esercizio **56 - cercastudente**

Nel prossimo esercizio 56, vi si chiede di usare due file matricola.txt ed email.txt forniti nell'archivio archivio020.tgz.

I diversi campi nelle righe dei due file sono separati da TABULAZIONI (TAB).

Se vi viene piu' facile, potete modificare quei file mettendo degli spazi (SPACE) al posto del TAB.

Fate come credete sia piu' facile per voi

Esercizio Esame Pratica - **56 - cercastudente**

Sia dato un file di testo matricola.txt in cui, in ciascuna riga e' contenuto il nome di uno studente, il suo cognome, il suo numero di matricola e il suo corso di laurea. I vari campi sono separati da tabulazioni (tab). Non esistono due studenti con la stessa matricola. Possono esistere piu' studenti con lo stesso nome, ma devono per forza avere matricole diverse. Vedere esempio a lato ----->

Sia dato un file di testo email.txt in cui, in ciascuna riga e' contenuto il numero di matricola di uno studente e la sua email universitaria, separati da tab. Non possono esistere due email uguali. Non possono esistere due matricole uguali. Vedi esempio a lato ---->

(esempi contenuti in archivio)

esempio file matricola.txt

Luca	Andreucci	578	informatica
Vittorio	Ghini	666	teologia
Giovanni	Pau	999	informatica
Luca	Andreucci	123	lettere

esempio file email.txt

578	LA@studio.unibo.it
666	VG@studio.unibo.it
999	GP@studio.unibo.it
123	LA2@studio.unibo.it

Scrivere uno script **cercastudente.sh** a cui vengono passati due argomenti **in questo ordine**: la EMAIL di uno studente e la MATRICOLA di quello studente. Lo script usa al suo interno il file matricola.txt, scopre il NOME e il COGNOME dello studente la cui matricola è stata passata come argomento, e stampa a video la stringa:

la account EMAIL appartiene allo studente NOME COGNOME.

dove EMAIL, NOME e COGNOME sono rispettivamente la email, il nome e il cognome dello studente.

Scrivere poi uno script **tuttistudenti.sh** che utilizza al suo interno il file email.txt. Lo script legge una dopo l'altra tutte le righe del file, da ciascuna riga estrae le informazioni EMAIL e MATRICOLA, e per ciascuna riga chiama lo script cercastudente.sh passandogli come argomenti la EMAIL e la MATRICOLA dello studente.