Esercitazioni di Informatica B

Riepilogo

Stefano Cereda
stefano.cereda@polimi.it

18/12/2018

Politecnico di Milano

Acquisizione dati da fogli di calcolo

La funzione *xlsread* consente di aprire in MATLAB file .xls creati con Microsoft Excel.

Proviamo ad usarla.

Stazione ferroviaria - TdE 29 Gennaio 2018 i

Si consideri il sistema di gestione dei treni in una stazione. In ogni stazione ci sono al più *N_BANCHINE* e ogni banchina consente la sosta e il transito di un treno. Ad ogni treno in stazione viene assegnato uno dei seguenti stati:

- *fuoriStazione*: il treno, in attesa di essere assegnato a una banchina, si trova fuori dalla stazione
- *inIngresso*: il treno è stato assegnato a una banchina ed è in marcia a velocità ridotta per entrare in stazione
- *inSosta*: il treno è fermo a una banchina
- *attesaOut*: il treno è fermo a una banchina in attesa di poter partire
- *inUscita*: il treno sta abbandonando la banchina procedendo a velocità ridotta.

Stazione ferroviaria - TdE 29 Gennaio 2018 ii

Si assuma che siano già stati introdotti i tipi *Stazione* e *Banchina* (si veda codice qui sotto). In particolare, *Stazione* contiene l'insieme e il numero delle banchine presenti in stazione (*banchine*), la coda dei treni (al più 20) in attesa all'ingresso (*codaTreni*, *nTreniInCoda*) e una variabile Booleana che dice se c'e' qualche treno in manovra in stazione (*bloccata*). *Banchina*, invece, contiene il numero della banchina e i dati del treno in sosta, se la banchina è in stato *occupato*. Ciascun treno, oltre ad avere il proprio stato e il numero della banchina eventualmente assegnata, ha un campo *minutiAttesaOut* che indica da quanti minuti è in attesa in stato *attesaOut*.

Stazione ferroviaria - TdE 29 Gennaio 2018 iii

```c
#define N_BANCHINE 10
#define N_TRENI_CODA 20

typedef char stringa[20];
typedef enum { falso, vero } boolean;
typedef enum { libero, occupato } statoBanchina;
typedef enum
{fuoriStazione, inIngresso, inSosta, attesaOut, inUscita
    } statoTreno;

typedef struct {
    stringa nome;
    statoTreno stTreno;
    int nBanchinaAssegnata;
    int minutiAttesaOut;
} Treno;
```

Stazione ferroviaria - TdE 29 Gennaio 2018 iv

```
16
17 typedef struct {
18     int numero;
19     statoBanchina stBanchina;
20     Treno trenoSosta;
21 } Banchina;
22
23 typedef struct {
24     Banchina banchine[N_BANCHINE];
25     int nBanchine; /* indica il numero di banchine
       presenti nella stazione */
26     Treno codaTreni[N_TRENI_CODA];
27     int nTreniInCoda;
28     boolean bloccata;
29 } Stazione;
```

Stazione ferroviaria - TdE 29 Gennaio 2018 v

Si supponga che sia stata definita la variabile *stazioneMI* di tipo *Stazione* e che questa variabile sia stata riempita in modo opportuno per rappresentare lo stato corrente della stazione di Milano, si risponda alle seguenti richieste:

- Dichiarando tutte le variabili necessarie, si definisca un frammento di programma che determina e stampa l'indice (nell'array di banchine) della banchina in cui è in sosta il treno che è in attesa da più tempo di poter partire.
- Si definisca un frammento di programma che, se la stazione non è bloccata dalle manovre di qualche treno e se vi è almeno un treno in coda per entrare in stazione, blocca la stazione, estrae dalla coda il primo treno che vi era stato inserito, gli assegna lo stato *inIngresso*, e aggiorna la coda in modo tale che poi contenga solo i rimanenti treni in coda. Se per esempio, prima

Stazione ferroviaria - TdE 29 Gennaio 2018 vi

di questa operazione la coda è cosi costituita: stazioneMI.codaTreni[0] = FR2092, stazioneMI.codaTreni[1] = IR097, stazioneMI.codaTreni[2] = FA2673 (per semplicità abbiamo riportato solo il nome del treno, ma, ovviamente, codaTreni dovrebbe contenere l'intera struct corrispondente), dopo l'operazione si deve avere che stazioneMI.codaTreni[0] = IR097, stazioneMI.codaTreni[1] = FA2673.

Trading - TdE 16 Febbraio 2018 i

Un'agenzia di trading online vuole memorizzare l'andamento del valore dei titoli che controlla. La memorizzazione viene effettuata in **500** istanti temporali equidistanti. I dati vengono salvati nel file MATLAB **log.mat** che contiene:

- la matrice **titoli**, le cui righe rappresentano i diversi titoli controllati e le cui colonne rappresentano i vari istanti in cui sono stati memorizzati i valori di tali titoli (quindi ogni cella della matrice contiene il valore di un titolo in un dato istante)
- il vettore colonna **andamento**, con lo stesso numero di righe della matrice **titoli**, che contiene un valore numerico per ogni titolo, indicativo del suo andamento complessivo crescente o decrescente

Trading - TdE 16 Febbraio 2018 ii

1. Scrivere in linguaggio MATLAB una funzione **splittaMatrice** che:
   - riceva in input una matrice **titoliTot** (con la stessa struttura di **titoli**), un vettore **andamentoTot** (con stessa struttura di vettore **andamento**) e uno scalare **soglia**;
   - fornisca in output due matrici **titoliOver** e **titoliUnder** (ognuna con la stessa struttura di **titoliTot**). **titoliOver** include solo le righe di **titoliTot** corrispondenti agli elementi di **andamentoTot** con valore maggiore o uguale di soglia. **titoliUnder**, invece, include le righe di **titoliTot** corrispondenti agli elementi di **andamentoTot** con valore minori di soglia.

Trading - TdE 16 Febbraio 2018 iii

2. Scrivere in linguaggio MATLAB uno script che:
   - 2.1 legga dal file **log.mat** i due dati memorizzati: **titoli** e **andamento**
   - 2.2 richiami la funzione **splittaMatrice** per separare titoli nelle due matrici **titoliOver** e **titoliUnder**, per un valore di soglia pari a 0
   - 2.3 crei un vettore **x** che contenga i 500 istanti di memorizzazione
   - 2.4 disegni su due grafici separati (che includano il titolo del grafico e il nome dei due assi) l'andamento dei titoli in **titoliOver** e **titoliUnder**, in funzione di x.

Processi - Tde 16 Febbraio 2018

Si assuma di avere un sistema operativo con un quanto di tempo di 25 ms e un processo P1 che riesce a decodificare 80 byte di segnale audio per ogni millisecondo di esecuzione.

Supponendo che una corretta riproduzione dell'audio richieda che vengano decodificati almeno 128 kbit/s, si risponda alle seguenti domande **giustificando le risposte**:

1. Se il processo P1 è l'unico attivo, può rispettare la velocità di codifica richiesta?
2. Supponiamo di attivare contemporaneamente a P1 altri 3 processi che abbiano lo stesso quanto di tempo e la stessa priorità di P1 e che ciascun cambio di contesto fra processi implichi un ritardo di 1 ms. Il processo P1 riuscirà a rispettare il limite richiesto per la corretta riproduzione?

Soluzione a Processi - Tde 16 Febbraio 2018

1. Se P1 è l'unico attivo, avrà l'intera CPU per se, quindi potrà decodificare 80 * 8 * 1000 bit/s = 640 kbit/s; quindi può rispettare la velocità di codifica richiesta.
2. Se sono attivi altri 3 processi, P1 non avrà l'intero tempo a disposizione, ma solo una parte di esso. In ogni secondo ci sono 1000 ms/(25+1) ms = 38 quanti di tempo circa che devono essere divisi tra i quattro processi. Se tutti hanno la stessa priorità e proseguono la loro esecuzione senza attendere dati dalle periferiche, ogni processo ha a disposizione 38/4=9 quanti di tempo circa. Quindi il nostro processo in un secondo decodificherà: 80 * 8 * 9 *25 bit/s = 144000 bit/s = 144 kbit/s, una quantità maggiore di quella minima richiesta. Di conseguenza, il nostro processo riuscirà a riprodurre correttamente l'audio.

Binario - TdE 29 Gennaio 2018

Un cardiopatico ha uno smartwatch con 48 byte di memoria liberi e vorrebbe tenere in memoria l'ECG relativo alla sua attività cardiaca durante la notte. Il singolo dato è rappresentato dalla differenza di potenziale generata dal battito cardiaco (in Volt) ogni ora, e può essere un intero in intervallo [-1000 +2000] (estremi compresi). Si assuma che lo smartwatch utilizzi una codifica in Complemento a 2 e si risponda alle seguenti domande:

1. Quale è il numero B di bit necessari per registrare ogni singolo dato corrispondente a un'ora di attività cardiaca?
2. Quante ore è possibile registrare ancora nella memoria libera, supponendo che ciascuna cella di memoria possa contenere esattamente B bit? Quanta memoria è necessaria per tenere in memoria 128 ore?
3. Supponiamo ora che siano state registrate le differenze di potenziale generate dal battito cardiaco delle prime 2 ore della notte, e che queste siano [-135, -978]. Si codifichino questi numeri in Complemento a 2 con B bit e si calcoli la loro somma

Soluzione a Binario - TdE 29 Gennaio 2018 i

1. $2^{10} = 1024 < 2000 < 2048 = 2^{11}$ Con B = 12 bit si riesce a coprire l'intervallo $[-2^{11}, 2^{11} - 1]$ e quindi si riesce a registrare l'attività cardiaca di un'ora.
2. Essendo la memoria disponibile pari a 48 byte = 48 * 8 bit = 384 bit e poiché per ogni ora sono necessari 12 bit per tenere in memoria la rappresentazione della differenza di potenziale generata dal battito cardiaco, si possono tenere in memoria ancora al massimo 48 * 8 / 12 = 32 ore. Se si quadruplicasse la memoria (i.e. 48 * 4 = 192 byte) otterrei spazio sufficiente per 32 * 4 = 128 ore

Soluzione a Binario - TdE 29 Gennaio 2018 ii

3. Tramite l'algoritmo delle divisioni successive $135_{10} = 10000111_2$, quindi il numero decimale -135_{10} in codifica Complemento a 2 con B = 12 bit è $-135_{10} = 111101111001_{CP2}$. Dato che $978_{10} = 1111010010_2$, poichè -978 è negativo, devo cambiare il valore di tutti i bit ottenuti e sommare 1 per trasformare tale codifica in un numero negativo in Complemento a 2 con B = 12 bit, ovvero $-978_{10} = 110000101110_{CP2}$. La somma di tali due valori rimane nell'intervallo di valori ammissibili con 12 bit, dunque è possibile registrare la somma delle differenze di potenziale generate dal battito cardiaco delle prime 2 ore della notte nello smartwatch.

Cornici - TdE 29 Gennaio 2018 i

Si consideri il seguente problema: si vuole creare una matrice quadrata che sia organizzata come quella in figura.

12	12	12	12	12
12	13	13	13	12
12	13	14	13	12
12	13	13	13	12
12	12	12	12	12

1. Si scriva in linguaggio MATLAB una funzione iterativa **cornici** che, data la dimensione N della matrice e un numero di partenza P, restituisca al chiamante una matrice quadrata NxN così definita: la matrice contiene nella cornice più esterna il numero P e numeri crescenti nelle cornici più interne.

Cornici - TdE 29 Gennaio 2018 ii

2. Si scriva inoltre uno script in linguaggio MATLAB che acquisisca da tastiera la dimensione desiderata N e il numero di partenza P, invochi la funzione **cornici** con gli opportuni parametri e infine stampi a video la matrice risultante.
3. Si implementi la funzione ricorsiva **corniciRic** in modo che presenti lo stesso comportamento della funzione **cornici**