



Politecnico di Milano

Facoltà di Ingegneria Civile, Ambientale e Territoriale

Informatica [097256] [091461] [091456]

Prof. G. Boracchi

Allievi Ingegneria Civile e Ambientale

Il prova in itinere (3 Febbraio 2017)

Cognome e nome	
Matricola	
Firma	

Domanda	1	2	3	4	5	TOT
Punteggio max	6	6	8	6	6	32
Punteggio						

La **durata** della **prova** è di **2 ore**. Non è consentito consultare libri o appunti, non è consentito l'uso di calcolatrici.

Scrivere solo sui fogli distribuiti utilizzando il retro delle pagine in caso di necessità e cancellando le parti di brutta con un tratto di penna. Non separare questi fogli.

Per tutti gli esercizi non è sufficiente fornire il risultato, ma **è necessario mostrare il procedimento seguito. Non dilungarsi però in spiegazioni prolisse.**

Gli allievi sono invitati a privilegiare **chiarezza, proprietà di linguaggio e sinteticità** nelle risposte agli esercizi, con l'obiettivo di **dimostrare la loro conoscenza degli argomenti.**

Domanda 1. Sistema Operativo (6 punti).

Si consideri un sistema monoprocessore con i 4 processi in stato di pronto aventi la seguente durata:

P1 75ns

P2 40ns

P3 20ns

P4 40ns

La CPU adotta una politica di tipo Round Robin con quanto di tempo 30ns, nessun processo esegue chiamate al supervisor.

Si assuma che i processi siano in ordinati in una FIFO da P1 a P4, tuttavia P3 e P4 hanno priorità alta, mentre P1 e P2 hanno priorità normale. A parità di priorità viene mandato in esecuzione il processo avanti nella FIFO.

Si descriva la sequenza di esecuzione fino a quando tutti i processi terminano, riportando

- dopo quanti ns termina ogni processo a partire dall'inizio
- quanti context switch richiede
- il tempo di attesa medio per ogni processo

Il sistema sopra descritto ha 16MB di memoria virtuale e pagine da 4KB (parola = 1Byte). Assumendo che l'indirizzo fisico sia di 16bit, si risopnda alle seguenti domande:

- quanto è grande la memoria fisica?
- di quanti bit è composto l'indirizzo virtuale?
- quante pagine fisiche e quante pagine virtuali sono presenti?

Si assuma ora che il sistema sia dotato di memoria cache L2 con Hit Time 20ns e MissPenality 100ns. Si dica il minimo valore dell' Hit Rate che garantisce un tempo di accesso medio di 40ns

SOLUZIONE

Sequenza di ingresso nella CPU

Processo in Esecuzione	Durata Es.	Tempo trasc.	Termina/CS
P3	20	20	Termina
P4	30	50	CS
P4	10	60	Termina
P1	30	90	CS
P2	30	120	CS
P1	30	150	CS
P2	10	160	T

P1	15	175	T
----	----	-----	---

Tempo medio attesa:

$\frac{1}{n} \sum_i^n (T_i - D_i)$ dove D_i è la durata prevista del processo P_i , T_i è l'istante in cui termina

$$((175 - 75) + (160 - 40) + (60 - 40) + (20 - 20)) / 4 = 80ns$$

Memoria fisica: $2^{16} = 65KB$

Bit indirizzo virtuale: $\log(16MB) = 24$

numero di bit offset = $\log_2(4K) = 12$

numero di bit pagina virtuale = $24 - \text{offset} = 12$

numero pagine virtuali = $2^{12} = 4096$ pagine virtuali

numero di pagine fisiche = $16 - \text{offset} = 4$

numero pagine fisiche = $2^4 = 16$ pagine fisiche

Tempo medio di attesa = $\text{HitRate} * \text{HitTime} + (1 - \text{HitRate}) * \text{MissPenalty}$

$$x * 20ns + (1 - x) * 100ns < 40ns$$

$$x > 0.75$$

Domanda 2. Matlab: interpretazione del codice (6 punti).

Si consideri la seguente funzione

```
function [] = funzRic(a, b)
if b == a
    c(1:a) = '-';
    c(b) = '+';
    disp(c);
else
    c(1:a) = '-';
    c(b) = '+';
    disp(c);
    funzRic(a, b+1);
end
```

Si stampi a schermo l'output del seguente script (riportando i valori inseriti)

N.B. Inserire 10 in caso di 0

```
1. >> a = input(['inserire l'ultima cifra del tuo numero di
    matricola']);
2. >> b = input(['inserire la penultima cifra del tuo numero
    di matricola']);

3. >> x = max([a, b]);
4. >> y = min([a, b]);

5. >> funzRic(x, y);
```

Si descriva brevemente cosa fa funzRic

Sono necessarie le istruzioni alle righe 3 e 4? Cosa succederebbe se le rimuovessimo?

SOLUZIONE

funzRic stampa su ogni riga l'avanzare (da sx a dx) di un più '+' in una riga di '-'. Ogni riga contiene un numero di caratteri pari al primo argomento e la posizione iniziale del '+' è data dal secondo argomento. La chiamata ricorsiva termina quando '+' raggiunge il fondo della riga.

es

```
>> funzRic(7,1)
```

+-----
-+-----
--+-
---+
----+
-----+
-----+
-----+

Se si rimuovessero le righe 3 e 4 potrebbe capitare che il primo argomento sia minore del secondo, in tal caso la funzione ricorsiva non terminerebbe.

Domanda 3. Matlab (8 punti).

Si scriva una funzione Matlab *chiamaAscensori* per regolare il funzionamento dei 40 ascensori di un palazzo.

- I piani in cui si trovano i 40 ascensori sono salvati in un vettore *PIANI* (es *PIANI(2)* corrisponde al piano in cui si trova l'ascensore 2)
- Lo stato dell'ascensore ('L' = libero, 'O' = occupato, 'R' = rotto) viene salvato in un vettore *STATO* di 40 elementi (*STATO(2)* = 'L' indica che il secondo ascensore è libero)
- Il piano in cui viene chiamato l'ascensore è salvato in una variabile *PIANO_CHIAMATA*

La funzione *chiamaAscensori* restituisce l'ascensore libero che si trova nel piano più vicino.

Nell'implementazione della funzione *chiamaAscensori* si invochi la funzione *closestPos* che prende in ingresso un vettore ed uno scalare e restituisce la posizione dell'elemento del vettore che è più vicino allo scalare.

In particolare si proceda come segue:

- Si implementi la funzione *closestPos*
- Si implementi la funzione *chiamaAscensori*

NB Verranno valutate a punteggio pieno solamente soluzioni che sfruttano le particolarità di Matlab nelle operazioni vettoriali

SOLUZIONE

```
function n = chiamaAscensori(pianiAscensori, statoAscensori, pianoChiamata)
```

```
% vettore che contiene tutti gli indici degli ascensori
indiciAscensori = [1 : numel(pianiAscensori)];
% sottovettore degli ascensori liberi
ascensoriLiberi = pianiAscensori(statoAscensori == 'L');
% posizione (all'interno di ascensoriLiberi) dell'ascensore più vicino
pos = closestPos(ascensoriLiberi, pianoChiamata);
% indice dell'ascensore più vicino tra tutti
n = indiciAscensori(pos);
```

```
function n = closestPos(vet, val)
```

```
dist = abs(vet - val);
[~, n] = min(dist);
```

Domanda 4. Progettazione DB: Schema Logico + Schema Concettuale (6 punti).

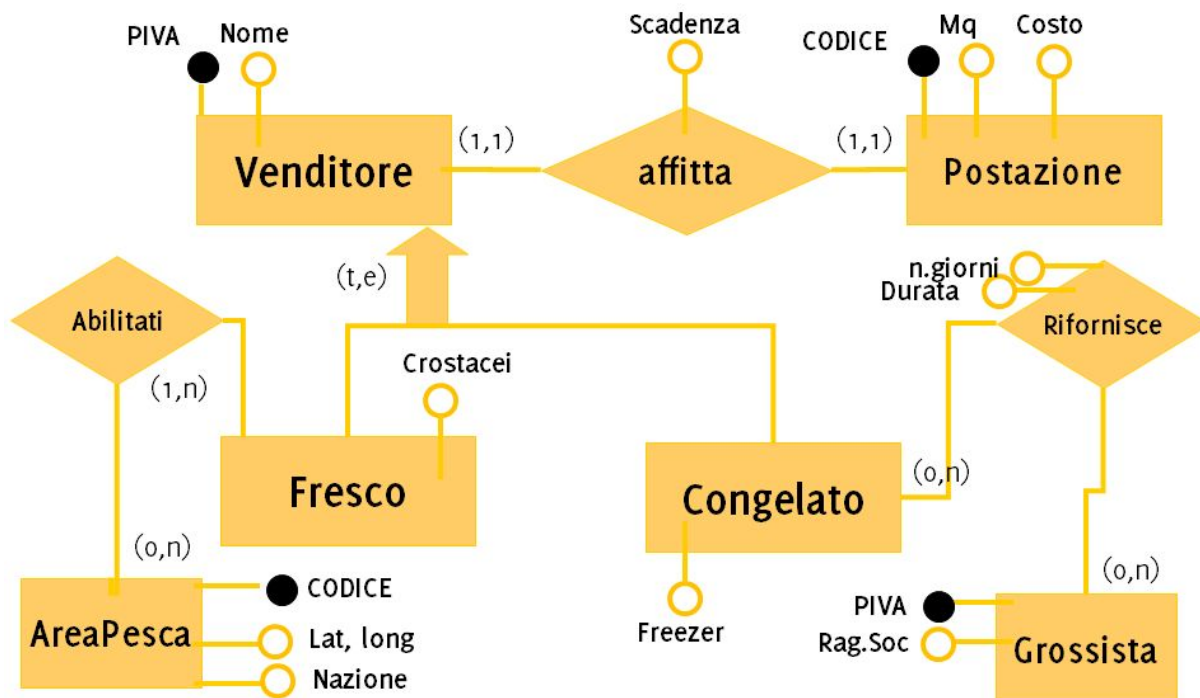
Si progetti la base dati per gestire il mercato ittico di Lecco. Nel mercato esistono diversi venditori, ciascuno identificato dalla P.ta IVA e dal suo nome. Ogni venditore occupa una precisa postazione con la sua bancarella e ha un'autorizzazione per la vendita del pesce fresco o, in alternativa, di pesce congelato. Per tutti i venditori abilitati alla vendita del pesce fresco occorre registrare se può vendere anche crostacei e le aree di pesca dove da dove proviene il pesce in vendita. Per i venditori di pesce congelato occorre registrare i contratti con i grossisti da cui possono rifornirsi e la temperatura minima raggiunta dai freezer. Ogni contratto è caratterizzato dalla durata e da un numero di giorni massimo cui il venditore può tenere in carico la merce.

Per ogni grossista è necessario registrare la P.ta IVA e la ragione sociale, il nome e l'anno di inizio dell'attività. Le aree di pesca sono identificate da un codice, dalle coordinate (lat, long) del suo centro e dalla nazione cui fa' riferimento (eventualmente INTERNAZIONALE). Infine, per le postazioni di vendita, occorre registrare un codice, la scadenza della concessione, i metri quadri ed il costo dell'affitto mensile.

1. Progettare lo schema **Entità-relazione** per la base dati del sistema sopra descritto
2. Progettare lo **schema relazionale** della base dati.

Si inseriscano brevi commenti solo se necessari per giustificare alcune scelte progettuali

SOLUZIONE



VenditoreFresco(PIVA, Crostacei, CodicePostazione, ScadenzaAffitto)

VenditoreCongelato(PIVA, TempFrezer, CodicePostazione, ScadenzaAffitto)

Postazione(Codice, Mq, CostoMensile)

Grossista(PIVA, Rag.Sociale)

RifornimentiGrossista(PIVA_Venditore, PIVA_Grossista, nGiorni, DurataContratto)

AreaPesca(Codice, Lat, Long, Nazione)

AbilitazioneAree(CodiceArea, PIVA_Venditore)

Domanda 5. Interrogazione DB: SQL (6 punti).

Si consideri il seguente schema logico in grado di gestire gli eventi in sale riunioni:

Sala (Nome, Piano, Capienza, Telefono, Videoproiettore)

Evento (ID, Nome, Descrizione, DataInizio, DataFine, NomeSala)

Organizzatore (CF, Nome, Cognome, Ruolo, Telefono, Mail)

Organizza (IDEvento, CFOrganizzatore)

Sottolineare le chiavi primarie ed indicare i vincoli di integrità referenziale (con frecce tratteggiate).

Definire le seguenti interrogazione in SQL :

1. Trovare il nome e la capienza di tutte le sale sprovviste di videoproiettore
2. Trovare il numero di eventi organizzati da DARIO COGLIATI
3. Trovare il nome e la descrizione di tutti gli eventi organizzati nelle sale senza telefono previsti per il mese di dicembre 2015
4. Trovare CF, nome e cognome di tutti gli organizzatori che non hanno mai organizzato eventi in sale da più di 20 persone.
5. Trovare per ogni sala il numero di eventi organizzati nel 2010.

SOL.

```
1- SELECT Nome, Capienza
FROM Sala
WHERE Videoproiettore IS NULL
```

```
2 - SELECT COUNT(*)
FROM Organizzatore, Organizza
WHERE Organizzatore.CF=Organizza.CFOrganizzatore
AND Nome="DARIO" AND Cognome='COGLIATI'
```

```
3- SELECT DISTINCT e.Nome, e.Descrizione
FROM Evento AS e, Sala AS s
WHERE e.NomeSala = s.Nome AND s.Telefono IS NULL AND
e.DataInizio >= 1/12/2015 AND e.DataInizio <= 31/12/2015 AND
e.DataFine >= 1/12/2015 AND e.DataFine <= 31/12/2015
```

```
4-SELECT CF, Nome, Cognome
FROM Organizzatore
WHERE CF NOT IN (SELECT DISTINCT o.CFOrganizzatore
FROM Organizza AS o, Evento AS e, Sala AS s
WHERE o.IDEvento = e.ID AND e.NomeSala = s.Nome
```

AND s.Capienza > 20)

```
5-SELECT NomeSala, COUNT(*)  
FROM Evento  
WHERE DataInizio >= 1/1/2010 AND DataInizio <= 31/12/2010 AND  
DataFine >= 1/1/2010 AND DataFine <= 31/12/2010  
GROUP BY NomeSala
```

