



Politecnico di Milano

Facoltà di Ingegneria Civile, Ambientale e Territoriale

Informatica [097256] [091461] [091456]

Prof. G. Boracchi

Allievi Ingegneria Civile e Ambientale

Appello di Recupero del 17 Febbraio 2017

Cognome e Nome	
Matricola	
Firma	

DOMANDE PRIMO COMPITINO	1	2	3	4	5	TOT
Punteggio max	2	6	6	8	10	32
Punteggio						

DOMANDE SECONDO COMPITINO	5	6	7	8	9	TOT
Punteggio max	10	6	6	6	6	32
Punteggio						

La **durata** della **prova** è di **1h40 per una prova singola, 3h per entrambe le prove**. 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. Tabella di Verità (2 punti, Primo Compitino)**

Si dica se le seguenti espressioni sono equivalenti

$$\sim(A \& B) \parallel C$$

$$\sim((C \& B) \& A)$$

**SOLUZIONE**

Le due espressioni non sono equivalenti, lo si può dimostrare facendo entrambe le tabelle di verità oppure sviluppando con le leggi di de Morgan

$$\sim(A \& B) \parallel C = (\sim A \parallel \sim B) \parallel C = \sim A \parallel \sim B \parallel C$$

$$\sim((C \& B) \& A) = \sim(C \& B) \parallel \sim A = (\sim C \parallel \sim B) \parallel \sim A = \sim A \parallel \sim B \parallel \sim C$$

e quindi identificando una tripletta di valori in cui differiscono. Ora chiamiamo  $X = \sim A \parallel \sim B$  che compare in entrambe le espressioni. Basta trovare una combinazione per cui  $X \parallel C$  è diverso da  $X \parallel \sim C$ . Guardiamo quindi quando  $X$  è falsa (altrimenti l'espressione è in entrambi i casi vera).

La tripletta di valori in cui differiscono è  $A = 1, B = 1, C = 0$  per cui la prima è falsa e la seconda è vera (oppure per  $A=1, B=1, C=1$  per cui la prima è vera, la seconda falsa).

## Domanda 2 Codifica Binaria (6 punti, Primo Compitino)

Indicare il numero di bit necessario per rappresentare in complemento a 2 ciascuno dei seguenti numeri. Si motivi la risposta

$N1 = -281$

$N2 = -256$

Indicare il numero di bit minimo necessario per la loro rappresentazione in modulo-segno, motivando la risposta

$N1$

$N2$

Codificare entrambi i numeri in complemento a due utilizzando il numero di bit che permette di codificare entrambi i numeri (si riportino tutti i passaggi necessari).

Si svolgano le seguenti operazioni in complemento a due evidenziando i bit del registro di flag attivi (si riportino tutti i passaggi necessari).

$N1 + N2$

$N1 - N2$

### SOL

$N1 = -281$  richiede 10 bit in CP2 per coprire l'intervallo  $[-2^9, 2^9 - 1] = [-512, 511]$

$N2 = -256$  richiede 9 bit in CP2 per coprire l'intervallo  $[-2^8, 2^8 - 1] = [-256, 255]$

$N1 = -281$  richiede 10 bit in MS per coprire l'intervallo  $[-2^9 + 1, 2^9 - 1] = [-511, 511]$

$N2 = -256$  richiede 10 bit in MS per coprire l'intervallo  $[-2^9 + 1, 2^8 - 1] = [-255, 255]$

Codifico quindi i numeri in CP2 a 10 bit

per codificare  $N1 = -281$  inizio a scrivere 281 in base 2 con 10 bit

01 0001 1001

Complemento e sommo 1 per ottenere  $(-281)_{CP2} = 10 1110 0111$

$N2 = -256$  inizio a scrivere 256 a 10 bit

01 0000 0000

Complemento e sommo 1 per ottenere 11 0000 0000

N1 + N2

10 1110 0111

11 0000 0000

-----

[1](1) 01 1110 0111

c'è overflow perchè la somma di due numeri negativi ha dato un numero positivo. Metto il bit di overflow uguale a 1, i.e. [1]. Ignoro il bit di carry (1)

N1-N2 (recupero la rappresentazione di -N2 = 256 = 01 0000 0000

10 1110 0111

01 0000 0000

-----

[0](0) 11 1110 0111

Non c'è overflow perchè sto sommando due addendi di segno disconcorde.

### Domanda 3. Matlab Intepretazione (6 punti, Primo Compitino)

Si consideri la seguente funzione

```
function outPut = mistero(p, s)
outPut = [];
for c = p
    if(c >= 'a' & c <= 'z')
        c = c + s;
        if c > 'z'
            c = c - ('z' - 'a') - 1;
        end
        if c < 'a'
            c = c + ('z' - 'a') + 1;
        end
    end
    outPut = [outPut, char(c)];
end
```

- 1) Si invochi la funzione mistero passando come primo argomento il proprio nome **in lettere minuscole** e come secondo argomento 2. Si visualizzi a schermo l'output.
- 2) Si descriva brevemente il funzionamento della funzione
- 3) Si scriva l'output visualizzato a schermo dalle seguenti invocazioni

```
>> mistero(mistero('mamma', 3), -3)
```

```
>> mistero(mistero('zucca', 3), -3)
```

### SOLUZIONE

```
>> mistero('giacomo',3)
```

```
ans =
```

```
jldfrpr
```

La funzione trasforma ogni lettera che compone una parola di un determinato numero di posizioni (s). Altri caratteri rimangono immutati.

Nel caso in cui una lettera venisse portata oltre la 'z', il restante 'a'. Allo stesso modo, in caso di spostamenti negativi ( $s < 0$ ), se una lettera viene portata prima della 'a', si ricomincia dalla 'z'.

La funzione può essere invertita chiamando l'output uno spostamento di segno opposto.

```
>> mistero(mistero('mamma', 3), -3)
```

```
ans =
```

```
mamma
```

```
>> mistero(mistero('zucca', 3), -3)
```

```
ans =
```

```
zucca
```

#### Domanda 4. Matlab (8 punti, Primo Compitino).

Si scriva una funzione *analizzaMatrice* che prende in ingresso una matrice rettangolare  $M$ , di numeri interi ed un vettore colonna  $V$  avente lo stesso numero di righe di  $M$  e contenente anch'esso numeri interi.

La funzione *analizzaMatrice* restituisce in output:

- una matrice  $S$  che vale 0 nelle posizioni in cui  $M$  è inferiore alla media di  $V$  e 1 altrimenti.
- un vettore colonna  $P$  avente lo stesso numero di righe di  $M$  e che contiene nella posizione  $i$ -sima il numero di elementi che nell' $i$ -sima riga di  $M$  sono superiori all'elemento  $i$ -simo in  $V$ .
- un vettore riga  $K$ , avente lo stesso numero di colonne di  $M$ , che riporta in  $K(i)$  la posizione del primo elemento dell' $i$ -sima colonna che è maggiore della media della colonna. Si calcoli tale posizione a partire dalla prima riga.

Si scriva quindi un frammento di script in cui si

- richiede all'utente di inserire una matrice ed un vettore
- si controlli se le dimensioni di questi sono compatibili con le richieste della funzione *analizzaMatrice*.
- In caso affermativo si invochi la funzione *analizzaMatrice* sulla matrice ed il vettore inseriti dall'utente
- si stampi a schermo il risultato.

Ad esempio, se si inserisse

```
M =  
    12    23     4  
    33     4     5
```

```
V =  
    12  
     3
```

si otterrebbe

```
S =  
     1     1     0  
     1     0     0
```

```
P =  
     1  
     3
```

```
Q =  
     2     1     2
```

## SOLUZIONE

```
function [S, P, Q] = analizzaMatrice(M,V)

S = M > mean(V);
P = zeros(size(M, 1), 1 );

for ii = 1 : size(M, 1)
    P(ii) = nnz(M(ii,:) > V(ii));
end
Q = zeros(1, size(M, 2));
for ii = 1 : size(M, 2)
    indx = find(M(:, ii) > mean(M(:, ii)));
    Q(ii) = indx(1);
end


matrice = input('inserire matrice: ')
vettore = input('inserire vettore: ')

if size(vettore, 1) == size(matrice, 1) && size(vettore, 2) ==
1
    [S, P, Q] = analizzaMatrice(matrice, vettore)
end
```



### Domanda 5. Matlab (8 punti, Primo e Secondo Compitino)

Sia  $\{x_t\}$ ,  $t = \{1, 2, \dots, T\}$  una serie storica, cioè una sequenza di valori che descrive, ad esempio, l'andamento del prezzo giornaliero di un titolo in borsa. Definiamo la **media mobile di  $\{x_t\}$  con finestra  $n$**  la serie storica  $\{m_t\}$ ,  $t = \{1, 2, \dots, T\}$  il cui valore nel punto  $t$  corrisponde alla media di  $m_t$  e degli  $n-1$  valori che precedono  $m_t$ .

Nel caso in cui, prima di  $t$ , non vi siano almeno  $n$  valori, non viene eseguita alcuna media e la media mobile corrisponde al valore della serie  $x$

Ad esempio:

la media mobile con finestra 3 della serie storica

[10 2 3 13 101]

è:

[10 2 5 6 39]

in quanto

- 10 non ha almeno 2 numeri che lo precedono nella serie in input
- 2 non ha almeno 2 numeri che lo precedono nella serie in input
- 5 è la media di 3, 2, 10
- 6 è la media di 13, 3, 2
- 39 è la media di 101, 13, 3

- A. Implementare una funzione MATLAB *controllaSeIncluso* che riceve in ingresso un indice *indx* ed una dimensione di una finestra  $n$  e restituisce true/false se esiste una finestra contenente almeno  $n-1$  valori che precedono *indx*.
- B. Implementare una funzione MATLAB *mediaMobile* che riceve in ingresso un vettore  $x$  contenente una serie storica e un intero  $n$  che rappresenta la dimensione della finestra da considerare, e restituisce un vettore contenente la media mobile di  $x$  con finestra  $n$ , con  $t = \{1, 2, \dots, \text{length}(x)\}$ . Si richiami la funzione *controllaSeIncluso*.
- C. Implementare uno script MATLAB che:
- genera una serie storica, come un vettore  $x$  di 500 numeri casuali compresi fra 0 e 10;
  - legge da tastiera due valori interi  $a$  e  $b$ ;
  - memorizza nel vettore  $mma$  la media mobile di  $x$  con finestra  $a$  e nel vettore  $mmb$  la media mobile di  $x$  con finestra  $b$ ;
  - visualizza su un unico grafico la serie rappresentata dal vettore  $x$ , la media mobile  $mma$ , e la media mobile  $mmb$  (riportando sull'ascissa i valori fra 1 e 500);
  - memorizza in un vettore  $t$  tutti gli indici di  $x$  in cui le due medie mobili  $mma$  e  $mmb$  **si incrociano** (ovvero tutte le posizioni  $i$  in cui si ha  $mma(i) \geq mmb(i)$  e  $mma(i-1) < mmb(i-1)$  o viceversa)

## SOL

```
function res = controllaSeIncluso (indx, n)
res = indx >= n-1
```

```
function mm = mediaMobile(x, n)
```

```
for i = 1:length(x)
    if(controllaSeIncluso(i, n))
        mm(i) = mean(x(i - n + 1 : i));
    else
        mm(i) = x(i);
    end
end
```

B.

```
x = rand(1, 500) * 10;
a = input('a = ');
b = input('b = ');
```

```
mma = mediaMobile(x, a);
mmb = mediaMobile(x, b);
```

```
plot(1:500, x, 1:500, mma, 1:500, mmb);
```

```
t = [find( (mma(2:end) >= mmb(2:end) & mma(1:end-1) < mmb(1:end-1)) |
(mma(2:end) < mmb(2:end) & mma(1:end-1) >= mmb(1:end-1)) )]
```

**Domanda 6. Sistema Operativo (6 punti, Secondo Compitino)**

- 1) Dire cos'è il kernel di un sistema operativo e di quali componenti si compone.
  
- 2) I processi P1 e P2 sono in stato di pronto in un sistema che adotta Round-Robin come algoritmo di scheduling. P1 ha maggior priorità di P2 e impiega un minor tempo di esecuzione essere eseguito.  
Chi entrerà per primo in esecuzione? Siamo certi che P1 terminerà prima di P2? Si motivi adeguatamente la risposta.
  
- 3) Si è progettato un sistema dotato di indirizzo virtuale a 16bit e memoria fisica di 8MB. Si indichi se le scelte progettuali sono state valide.

## SOLUZIONE

- 1) vedi slides
- 2) P1 va in esecuzione prima di P1. P1 potrebbe non terminare prima di P2 perchè potrebbe richiedere una supervisor call e finire i stato di attesa
- 3) 16bit per l'indirizzo virtuale permettono di indirizzare  $2^{16} = 65\text{KB}$  di memoria virtuale che è quindi significativamente minore di quella fisica. Il sistema non è stato ben progettato perchè tipicamente vale il contrario. Si può arrivare alla stessa conclusione guardando lo spazio di indirizzamento: la memoria fisica di 8MB richiede 23bit di indirizzo, mentre la memoria virtuale ha solo 16bit.

**Domanda 7. Matlab: interpretazione del codice (6 punti, Secondo Compitino)**

Si riporti sotto ad ogni istruzione l'output visualizzato da Matlab a schermo

```
>> a = [1 : 1 : 10]
```

```
>> b = a([1 3 5])
```

```
>> c = (a > 7)
```

```
>> d = a(a>7 | a <=2 )
```

```
>> e = find(d < 0)
```

```
>> f = min(find(a > 5))
```

```
>> g = a(a)
```

```
f = @(x,y) (x - y);
```

```
g = @(z) z^2;
```

```
risultato = f(g(3), g(2) - f(2,1))
```

## SOLUZIONE

a =

1 2 3 4 5 6 7 8 9 10

b =

1 3 5

c =

0 0 0 0 0 0 0 0 1 1 1

d =

1 2 8 9 10

e =

Empty matrix: 1-by-0

f =

6

g =

1 2 3 4 5 6 7 8 9 10

risultato =

6

**Domanda 8. Progettazione DB: Schema Logico + Schema Concettuale (6 punti Secondo Compitino)**

Il centralino del radio-taxi riceve le chiamate di richiesta di un taxi; al centralino lavorano le telefoniste, caratterizzate da un codice, cognome, nome, data di nascita. Le chiamate vengono tutte registrate e sono caratterizzate da un numero di chiamata, la data e ora della chiamata, il luogo della partenza e il luogo di destinazione (che le telefoniste richiedono sempre, non accettando chiamate se questo non viene fornito); per ogni chiamata si registra quale telefonista l'ha ricevuta. Le chiamate sono poi suddivise in chiamate urgenti e chiamate per appuntamento; per queste ultime si vuole conoscere anche la data e ora dell'appuntamento richiesto dal cliente. Quando un cliente esegue una chiamata urgente, specifica il tempo massimo che è disposto ad aspettare il taxi.

I taxi sono caratterizzati da un codice, il nome e cognome del proprietario (che è anche il conducente); i taxi si dividono in disponibili (che possono quindi servire le chiamate) e indisponibili, e di questi ultimi si vuole sapere fino a che data e ora sono indisponibili.

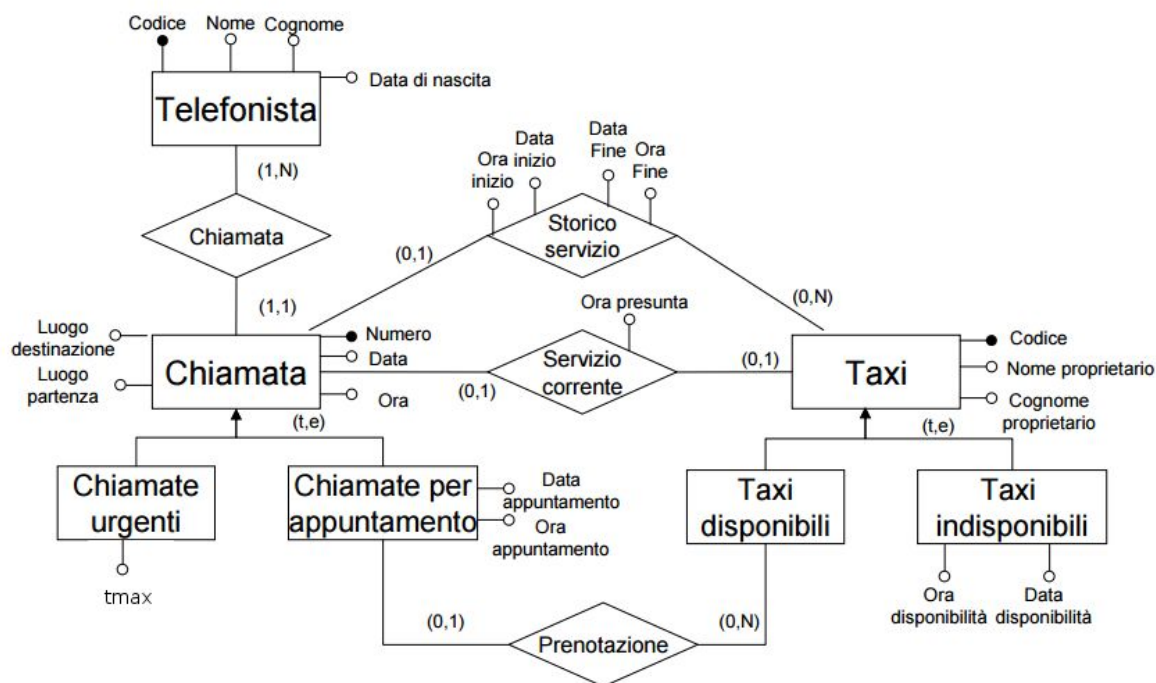
Le chiamate vengono servite dai taxi; si vuole sapere in ogni momento chi sta servendo una certa chiamata (una chiamata può essere servita al più da un taxi e un taxi può servire al più una chiamata), e l'ora presunta alla quale ciascun taxi terminerà il servizio. In più si vuole registrare lo storico di tutte le chiamate servite, indicando data e ora di inizio e data e ora di fine servizio del taxi che è intervenuto. Infine, ad ogni chiamata per appuntamento si associa uno o più taxi disponibili.

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

### PROGETTO CONCETTUALE



### PROGETTO LOGICO

**Telefonista** (cod\_telefonista, nome, cognome, data\_nascita)

**Chiamata** (numero, data, ora, luogo\_partenza, luogo\_destinazione, cod\_telefonista, tipo, data\_appuntamento, ora\_appuntamento, cod\_taxi\_prenotato, t\_max)

**Taxi** (cod\_taxi, nome\_proprietario, cognome\_proprietario, tipo, data\_disponibilità, ora\_disponibilità)

**Servizio corrente** (numero\_chiamata, cod\_taxi, ora\_presunta)

**Storico servizio** (numero\_chiamata, cod\_taxi, ora\_inizio, data\_inizio, ora\_fine, data\_fine)



**Domanda 9. Interrogazione DB: SQL (6 punti Secondo Compitino)**

Si consideri il seguente schema logico che descrive il DB di una compagnia di assicurazioni:

**AUTO** (Targa, Marca, Cilindrata, Potenza, CodF, CodAss)

**PROPRIETARI** (CodF, Nome, Residenza)

**ASSICURAZIONI** (CodAss, Nome, Sede)

**SINISTRO** (CodS, Località, Data)

**AUTOCOINVOLTE** (CodS, Targa, ImportoDelDanno)

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

Definire le seguenti interrogazione in SQL :

1. Trovare Targa e Marca delle Auto di cilindrata superiore a 2000 cc o di potenza superiore a 120 CV
2. Trovare Nome del proprietario e Targa delle Auto di cilindrata superiore a 2000 cc oppure di potenza superiore a 120 CV
3. Trovare Targa e Nome del proprietario delle Auto di cilindrata superiore a 2000 cc oppure di potenza superiore a 120 CV, assicurate presso la "SARA" .
4. Trovare Targa e Nome del proprietario delle Auto assicurate presso la "SARA" e coinvolte in sinistri il 20/01/2012
5. Per ciascuna Assicurazione, il nome, la sede ed il numero di auto assicurate

**SOL.**

**AUTO** (Targa, Marca, Cilindrata, Potenza, CodF, CodAss)

**PROPRIETARI** (CodF, Nome, Residenza)

**ASSICURAZIONI** (CodAss, Nome, Sede)

**SINISTRO** (CodS, Località, Data)

**AUTOCOINVOLTE** (CodS, Targa, ImportoDelDanno)

Vincoli integrità referenziale:

AUTO.CodF -> PROPRIETARI.CodF

AUTO.CodAss -> ASSICURAZIONI.CodAss

AUTOCOINVOLTE.CodS -> SINISTRO.CodS

AUTOCOINVOLTE.Targa -> AUTO.Targa

```
1- SELECT Targa, Marca
   FROM Auto
   WHERE Cilindrata > 2000 OR Potenza > 120
```

```
2 - SELECT Nome,Targa
   FROM Auto JOIN Proprietari ON Auto.CodF = Proprietari.CodF
   WHERE Cilindrata > 2000 OR Potenza > 120
```

```
3 - SELECT Nome,Targa
```

```
FROM ( Auto JOIN Proprietari ON Auto.CodF = Proprietari.CodF) JOIN Assicurazioni ON
    Assicurazioni.CodAss = Auto.CodAss
WHERE Cilindrata > 2000 OR Poenza > 120 AND Assicurazioni.Nome = 'SARA'
```

4 - SELECT Nome,Targa

```
FROM ((( Auto JOIN Proprietari ON Auto.CodF = Proprietari.CodF) JOIN Assicurazioni
ON    Assicurazioni.CodAss = Auto.CodAss) JOIN AUTOCOINVOLTE ON
AUTOCOINVOLTE.Targa = Auto.Targa ) JOIN SINISTRO ON SINISTRO.CodS =
AUTOVCOINVOLTE.CodS
WHERE Assicurazioni.Nome = 'SARA' AND SINISTRO.Data ="20/01/2012";
```

5- SELECT Nome,Sede,COUNT(\*)

```
FROM ASSICURAZIONI JOIN AUTO ON AUTO.CodAss = ASSICURAZIONI.CodAss
GROUP BY Nome,Sede;
```