



Politecnico di Milano

Facoltà di Ingegneria Civile, Ambientale e Territoriale

Informatica (ICA-LC) [091456] – Informatica A (ICA-LC) [091461]

Informatica [097256]

Prof. G. Boracchi

Allievi Ingegneria Civile e Ambientale

RECUPERO PRIMA PROVA IN ITINERE (19 Febbraio 2016)

<i>Cognome e nome</i>	
<i>Matricola</i>	
<i>Firma</i>	

Domanda	1	2	3	4	5	TOT
Punteggio max	4	4	12	6	6	32
Punteggio						

La **durata** della **prova** è di **1h30**. 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. La Macchina di Von Neumann (4 punti).

Si elenchino alcuni registri della CPU e si descriva brevemente le informazioni contengono

Domanda 2. Codifica Binaria (4 punti)

Si dica quale dei seguenti numeri (espressi in differenti basi) è maggiore e quale è minore. Si giustifichi la risposta

- (101 0111) in CP2
- (123) in base 10
- (123) in base 16
- AA in base 16
- (123) in base 8

si consideri quindi il numero maggiore ed il numero minore e li converta in CP2. Si esegua la loro somma indicando l'eventuale presenza di carry e offset.

SOLUZIONE

Il numero più piccolo è (1010111) perchè è l'unico negativo
il numero più grande è invece 123 in base 16 perchè ha un numero di cifre maggiore o uguale agli altri e ha base maggiore di quelli con lo stesso numero di cifre

(123)₁₆ diventa 0001 0010 0011 in base 2 (non è necessario usare l'algoritmo delle divisioni successive ma basta convertire ogni cifra in binario a 4 cifre)

in CP2 questo sarà 01 0010 0011 ed usa 10 cifre.

Occorre quindi codificare 101 0111 a 10 cifre. Convertiamo quindi 101 0111 in base 10 ottenendo $-2^6 + 2^4 + 2^2 + 2^1 + 2^0 = -41$ e riscriviamo -41 in CP2 a 10 bit.

Per fare questo codifico 41 in base 2 e ottengo 10 1001,

lo metto a 10 cifre e ottengo	00 0010 1001
complemento	11 1101 0110
e sommo 1	11 1101 0111

quindi il calcolo richiesto è	01 0010 0011
	11 1101 0111

(1)	00 1111 1010

che non ha alcun overflow (somma di numeri di segno opposto) e ha un bit di carry

Domanda 3. Matlab (12 punti).

Si implementi una funzione denominata *proiezioneMatrice* che riceve in ingresso una matrice **M** di numeri interi e che restituisce un vettore colonna **R** ed un vettore colonna **C** così definiti:

- **R**(i) contiene la somma di tutti i valori che compaiono sull'i-sima riga di **M** ad esclusione dell'elemento che sta sulla diagonale (se la diagonale interseca la riga i-sima).
- **C**(i) contiene la somma di tutti i valori che compaiono sull'i-sima colonna di **M** ad esclusione dell'elemento che sta sulla diagonale (se la diagonale interseca la colonna i-sima).

Si scriva quindi uno script che richiede all'utente di inserire una matrice **M** elemento per elemento e quindi

- chiama alla funzione *proiezioneMatrice*,
- stampa a schermo la matrice **M** e dei vettori restituiti dalla funzione.

Esempio:

se la funzione *proiezioneMatrice* riceve in ingresso la seguente matrice

2	2	13	2
22	31	2	3
32	22	23	1

restituirà

R = [17; 27; 55]

C = [54, 24, 15, 6]

SOLUZIONE

```
function [R,C] = proiezioneMatrice(M)
```

```
R = zeros(size(M, 1), 1);
```

```
C = zeros(1, size(M, 2));
```

```
for ii = 1 : size(M, 1)
```

```
    for jj = 1 : size(M, 2)
```

```
        if ii ~= jj
```

```
            R(ii) = R(ii) + M(ii, jj);
```

```
            C(jj) = C(jj) + M(ii, jj);
```

```
        end
```

```
    end
```

```
end
```

Domanda 5. Interpretazione del codice (6 punti)

```
p = [8: -2 : 0];
disp(['p = ', num2str(p)])

n = 2.^p;
disp(['n = ', num2str(n)])

r = sqrt(n);
disp(['r = ', num2str(r)])

m = [r , n]';
disp('m = ')
m

for ii = 1 : length(m)
    if (mod(ii, 2) == 0)
        disp([' ii = ', num2str(ii)])
        disp([' m(ii-1) = ', num2str(m(ii - 1))])
    end
end
```

SOLUZIONE

```
p = 8 6 4 2 0
n = 256 64 16 4 1
r = 16 8 4 2 1
m =
```

```
m =
```

```
16
8
4
2
1
256
64
16
4
1
```

```
ii = 2
m(ii-1) = 16
ii = 4
m(ii-1) = 4
ii = 6
m(ii-1) = 1
```

$$ii = 8$$

$$m(ii-1) = 64$$

$$ii = 10$$

$$m(ii-1) = 4$$

Domanda 5. Algebra di Boole (6 punti)

Un programma deve controllare se un numero N inserito dall'utente sia contemporaneamente:

- primo,
- dispari,
- compreso tra 3 e 100, estremi inclusi.

Quando il numero N inserito non soddisfa tutte le condizioni citate, l'inserimento deve essere ripetuto.

Per realizzare tale programma, un programmatore ha scritto il seguente frammento di codice Matlab, sfruttando la funzione `primo` che restituisce *true* se il suo argomento è un numero primo, *false* altrimenti:

```
while(primo(N) && (3 <= N) && (N <= 100) && (mod(N,2)))  
    N = input('Inserire un numero: ');  
end
```

Si risponda alle seguenti domande:

1. Si dica se il frammento di codice soddisfa i requisiti del programma. Nel caso in cui si ritenga che il frammento di codice non sia corretto, se ne indichi la motivazione e se ne fornisca un controesempio (ovvero un valore di N che ne mostri la non correttezza) spiegandolo.
2. Nel caso in cui si ritenga che il frammento di codice non sia corretto, se ne fornisca una versione in grado di soddisfare i requisiti indicati.
3. Si indichi se, e nel caso come, sia possibile semplificare la condizione riportata nel ciclo `while` per evitare ridondanze.
4. Si mostri la correttezza, o non correttezza, della condizione riportata nel ciclo `while` compilando una tabella di verità, ove:

A: `primo(N)`
B: `(3 <= N)`
C: `(N <= 100)`
D: `(mod(N,2))`

Soluzione

1. Un ciclo `while` viene ripetuto fino a quando la condizione che viene valutata è vera!

Requisito del programma è di **TERMINARE** l'inserimento (quindi avere la condizione `== 0` del ciclo `while`) quando sono vere tutte le tre condizioni da verificare sul valore di N . La

condizione che regola il ciclo while nel frammento di codice restituisce esattamente l'opposto. Pertanto il frammento di codice non è corretto.

2. La condizione corretta del ciclo while richiesto è esattamente la negazione di quella riportata nel frammento di codice. Un controesempio è il numero $N = 5$, per il quale il ciclo viene eseguito nuovamente nonostante 5 sia numero primo, dispari e compreso tra 3 e 100.
3. La condizione $(3 \leq N)$ implica che il numero N sia maggiore di 3, la condizione $\text{primo}(N)$ richiede che il numero sia primo, la condizione $(\text{mod}(N,2))$ e' invece inutile; infatti un numero primo e' sicuramente dispari, a meno che non sia pari a 2, ma questo valore viene escluso dalla condizione $(3 \leq N)$.

L'espressione da valutare e' quindi: $\neg(\text{primo}(N) \wedge (3 \leq N) \wedge (N \leq 100))$.

4.

A: $\text{primo}(N)$

B: $(3 \leq N)$

C: $(N \leq 100)$

D: $(\text{mod}(N,2))$

A	B	C	D	$(B \wedge C)$	$(B \wedge C \wedge D)$	$(A \wedge B \wedge C \wedge D)$
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	0	0	0
0	0	1	1	0	0	0
0	1	0	0	0	0	0
0	1	0	1	0	0	0
0	1	1	0	1	0	0
0	1	1	1	1	1	0
1	0	0	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	0	0	0
1	0	1	1	0	0	0
1	1	0	0	0	0	0
1	1	0	1	0	0	0
1	1	1	0	1	0	0
1	1	1	1	1	1	1