Esercizi di introduzione alla programmazione

Federico Reghenzani

Informatica ed Elementi di Informatica Medica 2017-18

1 Esercizi di codifica

1.1 Conversione da base 10 a un'altra base

Trovare x:

- $203_{10} = x_2$
- $19_{10} = x_2$
- $52_{10} = x_8$
- $71_{10} = x_5$
- $43_{10} = x_{16}$ (La base 16 viene codificata con le cifre $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F$)

1.1.1 Soluzione

Si utilizza il metodo delle divisioni successive.

- $203_{10} = 11001011_2$
- $19_{10} = 10011_2$
- $52_{10} = 64_8$
- $71_{10} = 241_5$
- $43_{10} = 2B_{16}$

1.2 Conversione da un'altra base a base 10

- $10101_2 = x_{10}$
- $100_5 = x_{10}$
- $73_8 = x_{10}$
- $FA_{16} = x_{10}$

- $10101_2 = 2^4 + 2^2 + 1 = 21_{10}$
- $400_5 = 4 * 5^2 = 100_{10}$
- $73_8 = 7 * 8 + 3 = 59_{10}$
- $1FA_{16} = 1 * 16^2 + 15 * 16 + 10 = 506_{10}$

1.3 Conversione in complemento a 2

Convertire i seguenti numeri in complemento a 2 su 8 bit:

- $+18_{10}$
- $+127_{10}$
- -55_{10}
- -90_{10}

1.3.1 Soluzione

- $+18_{10} = 00010010_2$
- $+127_{10} = 01111111_2$
- $-55_{10} = 11001001_2$
- $-90_{10} = 10100110_2$

1.4 Operazioni in complemento a 2

Eseguire le seguenti operazioni in complemento a 2 su 8 bit:

- $+18_{10} + 32_{10}$
- $+18_{10} + 120_{10}$
- $-18_{10} + 22_{10}$
- $-52_{10} - 100_{10}$

1.4.1 Soluzione

- $+18_{10} + 34_{10} = 00010010_2 + 00100010_2 = 00110100_2 = 52_{10}$
- $+18_{10}+120_{10} = 00010010_2+01111000_2 = 10001010_2 = -118_{10}$ `(OVERFLOW!)`
- $-18_{10} + 22_{10} = 11101110_2 + 00010110_2 = 00000100_2 = 4_{10}$
- $-52_{10}-101_{10} = 11001100_2+10011011_2 = 01100111_2 = 103_{10}$ `(OVERFLOW!)`

1.5 Operazioni bit-a-bit

Il simbolo $\sim$ indica l'operazione di NOT.
Il simbolo & indica l'operazione di AND.
Il simbolo | indica l'operazione di OR.
Il simbolo ^ indica l'operazione di XOR.

- $\sim 01001101_2$
- 11110000_2 & 01010101_2
- 11110000_2 | 01010101_2
- 11110000_2 ^ 01010101_2

1.5.1 Soluzione

- $\sim 01001101_2 = 10110010_2$
- 11110000_2 & $01010101_2 = 01010000_2$
- 11110000_2 | $01010101_2 = 11110101_2$
- 11110000_2 ^ $01010101_2 = 10100101_2$

2 Esercizi di dimensionamento

2.1 Video

Si consideri un flusso video ad alta risoluzione "4K" come una sequenza di immagini (o "frame") della dimensione di $4096x3112$ pixels. Il colore di ogni pixel è codificato come una tripletta RGB (Red-Green-Blue) della dimensione totale di 32-bit. La velocità del flusso video è di $60fps$ (frame-per-second). Calcolare la dimensione della memoria necessaria a contenere 24 ore di video.

Si considerino le immagini "RAW" ovvero non applicare alcun algoritmo di compressione.

2.1.1 Soluzione

Ogni frame è composto da $4096x3112 = 12746752$ pixels, ciascuno dei quali grande $3B$, per una dimensione totale di $\sim 36MiB$. Ogni secondo vengono catturati 60 frames che richiedono una velocità di salvataggio complessiva in memoria di $36MiB * 60 = 2.1GB/s$. Infine, per registrare l'intero video di 24 ore, cioè 86400 secondi, è necessaria una capacità di $86400s * 2.1GB/s = 177TiB$.

Ad oggi, una velocità di scrittura e una quantità di memoria così elevate sono irrealistiche. Per questo si rende necessario utilizzare gli **algoritmi di compressione** per i video e immagini ad alta risoluzione.

2.2 ECG

La registrazione di un elettrocardiogramma (ECG) in genere ha una durata di $10s$. Al termine dell'elaborazione vengono salvate in un database le onde P, Q, R, S e T, ognuna delle quali registrate con una frequenza di $250Hz$ e una risoluzione di $16 - bit$. Calcolare lo spazio occupato da un database contenente 10.000 ECG considerando inoltre che ogni 20 ECG il database deve registrare $2048B$ di metadati.

2.2.1 Soluzione

Il numero totale di campioni per ogni ECG è $10s \cdot 250Hz = 2500$ per un totale di $2500 * 2B * 5 = 25KB =\sim 24.4KiB$. Il database occuperà $25KB \cdot 10000 + 10000/20 \cdot 2048B =\sim 251MB =\sim 239.4MiB$

3 Esercizi Flowchart

Le soluzioni sono presenti nell'archivio compresso.

3.1 Verifica se un numero è un quadrato perfetto

Dato in input un numero intero $A > 0$, verificare se A è un quadrato perfetto, cioè se esiste un numero B tale che $A = B^2$.

3.2 Calcolo del MCD di un numero

Calcolare il Massimo Comune Divisore di due numeri interi positivi A e B.
Esempio input: $90, 27$
Esempio output: 9

3.3 Serie di Fibonacci

Scrivere un programma che ricevuto un numero intero positivo N come input, stampi i primi N numeri della serie di Fibonacci.
I primi due numeri della serie di fibonacci sono sempre 0 e 1. I successivi numeri sono $x_n = x_{n-1} + x_{n-2}$.
I primi 10 numeri della serie di Fibonacci sono quindi: $0, 1, 1, 2, 3, 5, 8, 13, 21, 34$.

3.4 Sequenza di coppie

Leggere una sequenza di coppie di interi (A,B) e calcolarne le differenze $B - A$. Per ogni coppia stampare la differenza solo se essa è positiva. Quando la differenza è nulla, il programma termina.
Esempio input: $2, 5, 6, 7, 5, 1, 2, 1, 0, 3, 1, 1$
Esempio output: $3, 1, 3$

3.5 Numeri primi di Mersenne

Dato in input un numero primo P, verificare che il numero di Mersenne $M = 2^P - 1$ sia anch'esso primo e nel caso stamparlo.
Esempi:

- $P = 2$ produce un numero primo di Mersenne perchè $M = 2^2 - 1 = 3$ è primo;
- $P = 3$ produce un numero primo di Mersenne perchè $M = 2^3 - 1 = 7$ è primo;
- $P = 11$ **non** produce un numero primo di Mersenne perchè $M = 2^{11} - 1 = 2047$ **non** è primo.

3.6 Il prodotto delle sequenze

Leggere da input due numeri N e M, successivamente leggere da input N sequenze di M numeri. Stampare la somma dei N prodotti degli M interi di ciascuna sequenza.

Esempi:

- Input: $3, 2, 0, 1, 2, 3, 4, 5$ Output: 26 (Calcolato come $0 * 1 + 2 * 3 + 4 * 5$)
- Input: $3, 3, -1, 2, 3, 4, 5, 6, 0, 6, 2$ Output: 114 (Calcolato come $1 * 2 * 3 + 4 * 5 * 6 + 0 * 6 * 2$)