Esercizi di programmazione in Assembly

Alessandro Pellegrini

Esercizio 1 : Dato un vettore di n unsigned word, individuare il minimo. Il vettore deve essere esplicitamente dichiarato come variabile globale; il minimo deve essere salvato all'interno di un'altra variabile globale esplicitamente dichiarata.

Esercizio 2 : Dato un vettore di n signed word, individuare il massimo. Il vettore deve essere esplicitamente dichiarato come variabile globale. Il codice deve essere organizzato come una funzione il cui passaggio di parametri (indirizzo del vettore) ed il valore di ritorno (il massimo) devono rispettare le calling conventions z64.

Esercizio 3 : Scrivere un programma che, dato un vettore di n longword, salvi in un secondo vettore gli elementi memorizzati nel primo in ordine inverso.

Esercizio 4 : All'indirizzo `0x1280` è presente un vettore di 64 signed longword. Implementare in Assembly l'algoritmo bubble sort per ordinare gli elementi di questo vettore.

Esercizio 5 : All'inidirizzo `0x5555` è presente un vettore di n unsigned word. Scrivere una subroutine Assembly che scriva all'interno del registro %rax il numero di elementi pari presenti in posizione dispari all'interno del vettore (ossia, se in posizione 1 nel vettore è presente un numero pari, %rax viene incrementato, se in posizione 3 è presente un numero pari %rax viene incrementato, e così via). Il numero n di elementi del vettore può essere sia pari che dispari. Si implementi il programma principale che permette di passare alla subroutine implementata l'indirizzo e la taglia di un vettore per effettuare il calcolo.

Esercizio 6 : Dato un vettore di 32 unsigned byte, scandire il vettore e creare una maschera di bit all'interno del registro %eax in cui un bit viene impostato ad 1 se l'elemento corrispondente del vettore è pari, a 0 altrimenti.

Esercizio 7 : Scrivere una subroutine `divisione` che implementi via software l'operazione di divisione tramite il metodo delle sottrazioni successive. La funzione opera su interi non segnati e rispetta le calling convention dello z64. Scrivere un programma che utilizzi questa funzione per calcolare il risultato della divisione tra due numeri specificati all'interno di due variabili globali.

Esercizio 8 : Utilizzare la funzione `divisione` realizzata per l'Esercizio 7 per implementare un programma che calcoli il massimo comune divisore tra due numeri utilizzando il metodo delle divsioni successive.

Esercizio 9 : Data una variabile globale `a` di tipo word, memorizzare all'interno di un'altra variabile globale `b` di dimensione longword la rappresentazione dello stesso dato contenuto in `a` rappresentato mediante un codice correttore d'errore a distanza di Hamming 3.

Esercizio 10 : Data una variabile globle `a` di tipo longword che memorizza un dato rappresentato con codice di correzione d'errore a distanza di Hamming 3, implementare: 1) una subroutine Assembly che restituisca il valore booleano `true` se il dato passato non ha alcun errore, `false` altrimenti; 2) un subroutine Assembly che restituisca la rappresentazione originale del dato passato (eliminando, quindi, i bit di parità). Scrivere un programma Assembly che utilizzi queste due subroutine per verificare la correttezza di un dato contenuto in una variabile globale e (in caso positivo) decodifichi lo stesso.

Esercizio 11 : Dato un numero intero rappresentato in aritmetica segnata, scrivere un programma assembly che converta questo numero nella rappresentazione in virgola mobile IEEE 754.