S10/L3 Esercizio 29/11/2023

Nell'esercizio odierno ci viene richiesto di identificare lo scopo di ogni istruzione, inserendo una descrizione per ogni riga di codice del seguente c. Assembly:

Il linguaggio Assembly

E' è un linguaggio di programmazione a basso livello (es. appunto Assembly) che rappresenta un'interfaccia tra il linguaggio macchina e il linguaggio di alto livello (es. C e C++), è univoco per una data architettura di un PC, ma cambia da architettura ad architettura. La conoscenza del linguaggio Assembly servirà per "leggere" le istruzioni eseguite dalla CPU in formato leggibile dall'uomo.

La base del linguaggio sono le istruzioni che sono costituite da 2 parti: un codice mnemonico (ovvero una parola che identifica l'istruzione da seguire) e uno o più operandi che identificano le variabili o la memoria oggetto dell'istruzione

0x00001141 <+8>: **mov (move) EAX, 0x20** **= 32 in decimale - muove il valore esadecimale 0x20 nel registro EAX**

0x00001148 <+15>: **mov (move) EDX, 0x38** **= 56 in decimale – muove il valore esadecimale nel registro EDX**

0x00001155 <+28>: **add (addizione) EAX, EDX = 32+56=88 in decimale - aggiunge il contenuto del registro EDX al registro EAX (88 viene salvato in EAX)**

0x00001157 <+30>: **mov (move) EBP, EAX = 88 in decimale (salvato in precedenza) - muove il contenuto di EAX nel registro di base EBP (ora EBP conterrà il valore 88)**

0x0000115a <+33>: **cmp (compara) EBP, 0xa = 10 in decimale – compara il contenuto del registro EBP con il valore esadecimale 0xa**

0x0000115e <+37>: **jge (jump, è una condizione: x>=y) 0x1176 <main+61> = salta all'indirizzo 0x1176 se il risultato della comparazione è maggiore o uguale, quindi se EBP è maggiore o uguale a 10 salterà all'indirizzo 0x1176**

0x0000116a <+49>: **mov (move) eax,0x0 = 0 in decimale - muove il valore 0 nel registro EAX**

0x0000116f <+54>: **call (chiama) 0x1030 <printf@plt> = chiama la funzione "printf" con l'indirizzo 0x1030. Potrebbe servire a stampare il valore presente in EAX**