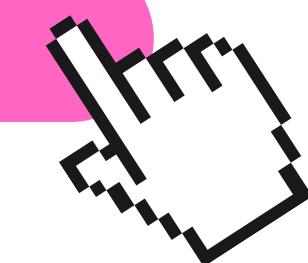


WRITTEN BY NOEMI DE MARTINO

S10vL3

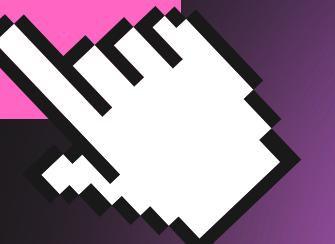
ASSEMBLY X86



TRACCIA

Nella lezione teorica del mattino, abbiamo visto i fondamenti del linguaggio Assembly. Dato il codice in **Assembly per la CPU x86** allegato qui di seguito, identificare lo scopo di ogni istruzione, inserendo una descrizione per ogni riga di codice.

Ricordate che i numeri nel formato OxYY sono numeri esadecimali. Per convertirli in numeri decimali utilizzate pure un convertitore online, oppure la calcolatrice del vostro computer.



0x00001141 <+8>: mov EAX,0x20

0x00001148 <+15>: mov EDX,0x38

0x00001155 <+28>: add EAX,EDX

0x00001157 <+30>: mov EBP,EAX

0x0000115a <+33>: cmp EBP,0xa

0x0000115e <+37>: jge 0x1176 <main+61>

0x0000116a <+49>: mov eax,0x0

0x0000116f <+54>: call 0x1030 <printf@plt>

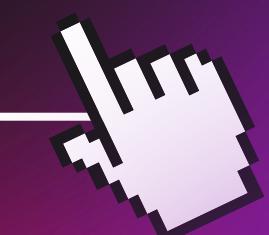
0x00001141 <+8>: mov EAX,0x20

Descrizione

Questa istruzione carica il valore esadecimale 0x20 (che corrisponde a 32 in decimale) nel registro EAX.

Scopo

Inizializzare il registro EAX con il valore 32. Questo valore può rappresentare un dato specifico o un parametro iniziale per un'operazione successiva.



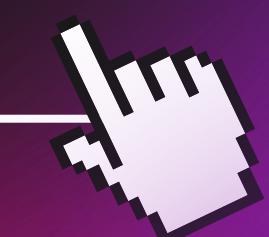
0x00001148 <+15>: mov EDX,0x38

Descrizione

Questa istruzione carica il valore esadecimale Ox38 (che corrisponde a 56 in decimale) nel registro EDX.

Scopo

Inizializzare il registro EDX con il valore 56. Simile alla precedente, questa istruzione prepara un valore che sarà utilizzato successivamente.



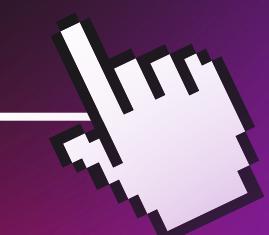
0x00001155 <+28>: add EAX,EDX

Descrizione

Somma il contenuto del registro EDX (56) al contenuto del registro EAX (32). Il risultato della somma (88) viene memorizzato nel registro EAX.

Scopo

Eseguire l'operazione aritmetica di somma tra i due valori precedentemente caricati nei registri. Ora, EAX contiene il risultato della somma (88).



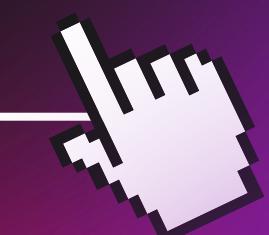
0x00001157 <+30>: mov EBP, EAX

Descrizione

Copia il valore attuale del registro EAX (88) nel registro EBP.

Scopo

Salvare il risultato della somma in un altro registro (EBP). Questo può essere fatto per preservare il risultato per operazioni future o per rispettare convenzioni di chiamata delle funzioni.



0x0000115a <+33>: cmp EBP,Oxa

Descrizione

Confronta il valore del registro EBP (88) con il valore esadecimale Oxa (10 in decimale).

Scopo

Effettuare un confronto per determinare se il valore nel registro EBP è maggiore, minore o uguale a 10. Questo confronto imposta i flag del processore, che verranno utilizzati dalla successiva istruzione condizionale.



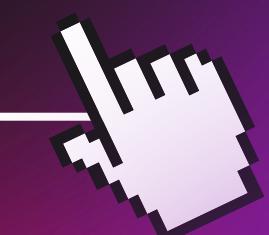
0x0000115e <+37>: jge 0x1176 <main+61>

Descrizione

Salta all'indirizzo 0x1176 (etichettato come <main+61>) se il risultato del confronto precedente (cmp EBP,Oxa) indica che EBP è maggiore o uguale a 10.

Scopo

Realizzare un salto condizionale. In questo caso, poiché 88 è maggiore di 10, il salto verrà effettuato. Questa istruzione consente di alterare il flusso di esecuzione in base al risultato del confronto.



0x0000116a <+49>: mov eax,0x0

Descrizione

Carica il valore 0 nel registro EAX.

Scopo

Inizializzare EAX a zero. Questo può essere usato per preparare un valore di ritorno per una funzione, o come parametro per un'operazione successiva. Questa istruzione sarà eseguita solo se il salto condizionale precedente (jge) non è stato eseguito.



0x0000116f <+54>: call 0x1030 <printf@plt>

Descrizione

Chiama la funzione printf (situata all'indirizzo 0x1030). printf è una funzione di libreria standard che stampa formattato sullo standard output.

Scopo

Eseguire la funzione printf. L'argomento e il formato di stampa dipendono dallo stato del sistema e dei registri, ma tipicamente printf stampa valori sullo schermo.

