Pratica S10/L3 - Assembly x86

L'esercizio di oggi consiste nel prendere 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.

0x00001141 <+8>: mov EAX,0x20 si traduce nel caricare il valore esadecimale 0x20 (32 in decimale) nel registro EAX.

0x00001148 <+15>: mov EDX,0x38 si traduce nel caricare il valore esadecimale 0x38 (56 in decimale) nel registro EDX.

0x00001155 <+28>: add EAX,EDX si traduce nell'aggiungere il contenuto del registro EDX al contenuto del registro EAX, quindi se prendiamo gli esempio sopra avremmo che AEX = 56+32 =88

0x00001157 <+30>: mov EBP, EAX si traduce nel copiare il contenuto del registro EAX nel registro EBP=88.

0x0000115a <+33>: cmp EBP,0xa si traduce in un confronto tra il valore contenuto nel registro EBP e il valore immediato 0xa (10 in decimale). La differenza tra i due valori viene utilizzata per impostare i flag del registro di stato.

0x0000115e <+37>: jge 0x1176 <main+61> L'istruzione assembly jge 0x1176 rappresenta un salto condizionato. In particolare, "jge" sta per "jump if greater than or equal" (salta se maggiore o uguale). Il destino del salto è l'indirizzo di memoria 0x1176, che indica l'inizio di un'istruzione successiva nel programma.

0x0000116a <+49>: mov eax,0x0 si traduce nel caricare il valore zero (0) nel registro EAX e sovrascrivere il valore presente in essa.

0x0000116f <+54>: call 0x1030 <printf@plt> rappresenta una chiamata a una funzione, in questo caso, alla funzione 'printf' situata all'indirizzo di memoria 0x1030. La funzione 'printf' è comunemente utilizzata per stampare output formattato sullo schermo.