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 0xYY sono numeri esadecimali. Per convertirli in numeri decimali utilizzate pure un convertitore online, oppure la calcolatrice del vostro computer (per programmatori). 0x00001141 :

1) 0x00001141 <+8>: mov EAX,0x20

2) 0x00001148 <+15>: mov EDX,0x38

3) 0x00001155 <+28>: add EAX,EDX

4) 0x00001157 <+30>: mov EBP, EAX

5) 0x0000115a <+33>: cmp EBP,0xa

6) 0x0000115e <+37>: jge 0x1176 <main+61>

7) 0x0000116a <+49>: mov eax,0x0

8) 0x0000116f <+54>: call 0x1030 <printf@plt>

1) l’istruzione mov eax,0x20 🡪 coincide in decimale al valore 32 , copia il valore 32 nel registro eax

2) mov edx.0x38 🡪 coincide al valore 56, copia il valore 56 nel registro edx

3) add EAX,EDX 🡪 add indica l’istruzione di somma , vengono sommati i valori dei registri di aex e edx , il quale valore verrà inserito nel registro di eax

4) mov ebp,asx 🡪 in questo caso l’istruzione mov , sta ad indicare che il valore contenuto nel registro di asx verrà copiato nel registro di ebp

5) cmp ebp,0xa 🡪 l’istruzione cmp modifica i flag (zf) (cf) , l’istruzione cmp lavora su destinazione/sorgente. Quindi traduciamo il valore esadecimale in decimale , cmp ebp,10. Sappiamo che il valore di ebp è di 88 , quindi sottrae 88-10 , vediamo che la destinazione è maggiore della sorgente , pertanto i flag menzionati prima hanno entrambi valori 0

6) jpe 0x1176 <main+61> 🡪 jump if greater or equal , Salta all’indirizzo specificato (0x1176) se la destinazione è maggiore o uguale della sorgente

7) mov eax,0x0 🡪 copia il valore 0 ( dove non sono andato ad usare un convertitore , testimone marco ) nel registro di memoria di eax

8) call 0x1030 🡪 chiamata all’istruzione printf@plt, la quale visualizza a schermo i valori degli argomenti secondo il formato specificato