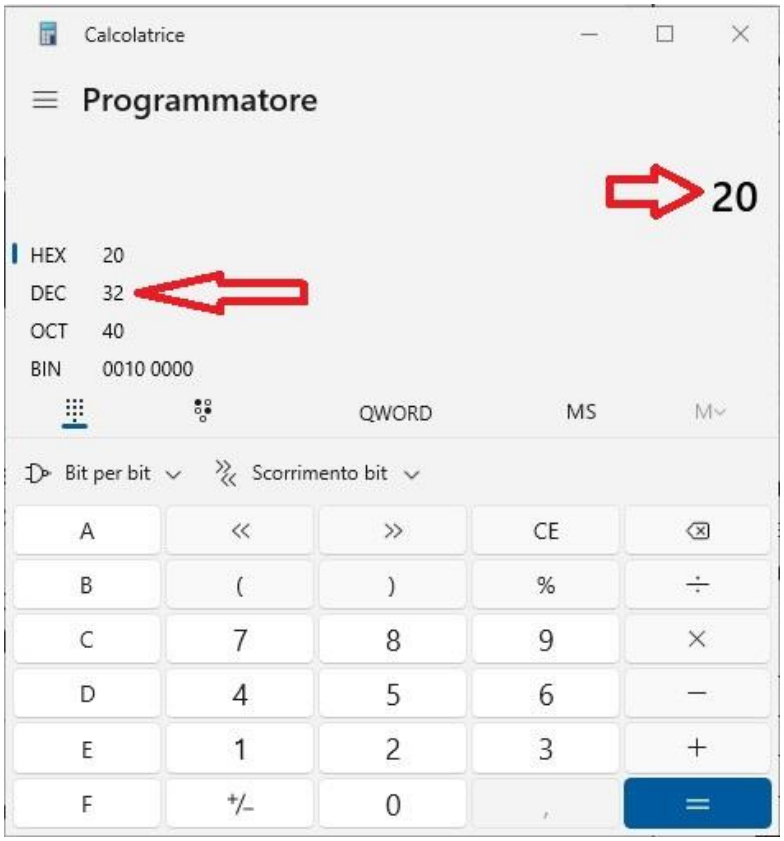
ASSEMBLY x86

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

```
0×00001141 <+8>:     mov    EAX,0×20
0×00001148 <+15>:    mov    EDX,0×38
0×00001155 <+28>:    add    EAX,EDX
0×00001157 <+30>:    mov    EBP, EAX
0×0000115a <+33>:    cmp    EBP,0xa
0×0000115e <+37>:    jge    0×1176 <main+61>
0×0000116a <+49>:    mov    eax,0×0
0×0000116f <+54>:    call   0×1030 <printf@plt>
```

Andremo come prima cosa a convertire i numeri in esadecimale 0xYY in decimale utilizzando come da traccia la nostra calcolatrice per programmatori. Nel formato esadecimale 0xYY, YY rappresenterà la versione esadecimale del numero decimale.
Come da figura andremo quindi ad inserire i valori YY in HEX (esadecimale) per convertirli in DEC (decimale).



In questo caso:

- 0x20 (HEX) = 32 (DEC);

Così per gli altri numeri sarà:

- 0x38 (HEX) = 56 (DEC)
- 0xa (HEX) = 10 (DEC)
- 0x1176 (HEX) = 4470 (DEC)
- 0x0 (HEX) = 0 (DEC)
- 0x1030 (HEX) = 4144 (DEC)

I registri EAX, EDX sono registri “general purpose” a 32 bit, servono come sorgente o destinazione per qualunque operazione di trasferimento e aritmetico logica.

Il registro EBP è un registro previsto per tenere sotto controllo il flusso e l’esecuzione dello stack durante le chiamate di funzione, è il puntatore alla base dello stack.

ISTRUZIONE	DESCRIZIONE
mov EAX, 32	Copia il valore 32 nel registro EAX
mov EDX, 56	Copia il valore 56 nel registro EDX
add EAX, EDX	Somma il valore contenuto in EDX a EAX e salva il risultato in EAX 56 + 32 = 88
mov EBP, EAX	Copia il contenuto del registro EAX (ora aggiornato ad 88) nel registro EBP
cmp EBP, 10	cmp è un’istruzione condizionale di confronto (compare), esegue un confronto (sottrazione) tra Destinazione e Sorgente impostando di seguito il registro FLAG a 0 o 1. Se la Destinazione (in questo caso EBP = 88) è maggiore della sorgente (0xa = 10) i flag ZF (zero flag) e CF (carry flag) saranno 0. 88 > 10 ⟶ ZF = 0, CF = 0
jge 4470 <main+61>	Questa istruzione è un salto condizionato, se il risultato del confronto precedente è maggiore o uguale a 0 l'esecuzione del programma salta all'indirizzo 4470 <main+61>, altrimenti l'esecuzione prosegue all'istruzione successiva. In questo caso la Destinazione (EBP = 88) > Sorgente (0xa = 10) quindi salterà alla locuzione 4470 <main+61>
mov EAX, 0	Copia il valore 0 nel registro EAX
call 4144 <printf@plt>	L’istruzione call passa l’esecuzione del programma alla funzione chiamata per il quale verrà creato un nuovo stack. Chiama la funzione <printf@plt> che si trova all’indirizzo di memoria 4144 passando il contenuto di EAX come parametro

È un programma scritto in codice Assembly per architettura x86, che calcola la somma di due numeri e memorizza il risultato nel registro EBP, poi confronta il risultato con 10. Se il risultato è maggiore o uguale, il programma salta ad un indirizzo specifico, altrimenti continua ad eseguire le istruzioni successive, in questo caso, sposta 0 nel registro EAX e chiama la printf per stampare il contenuto di EAX.