PRATICA S10-L3

SCOPO: Identificare lo scopo di ogni istruzione del codice in seguito, inserendo una descrizione per ogni riga di codice.

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>

Questi codici sono composti in generale dell'indirizzo della memoria ram; degli operatori o istruizione (mov , add , jge , call...) e degli operandi.

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

Questo codice con l'operatore **mov** sposta il valore del sorgente **0x20** nel registro **EAX;** Quindi **EAX** = 0.

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

Questo codice con l'operatore **mov** sposta il valore del sorgente **0x38** nel registro **EDX;** Quindi **EDX = 0 .**

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

Questo codice con l'operatore **add** somma il valore del registro **EDX** con il valore del registro **EAX;** Quindi **add = 0 .**

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

PRATICA S10-L3

Questo codice con l'operatore **mov** sposta il valore del registro sorgente **EAX** nel registro di destinazione **EBP**. Visto che **EAX = 0 -> EPB = 0 .**

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

l'istruzione **cmp** compare il valore di **0xa** con valore del registro **EBP** e otteniamo i risultati seguenti:

Se EBP = 0xa allora ZF= 1 e CF = 0

Se EBP < 0xa allora ZF= 0 e CF = 1

Se EBP > 0xa allora ZF=0 e ZC = 0

Visto che EBP = 0 Quindi **EBP = 0xa = 0 -> ZF =1 e CF = 0**

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

Visto che **0x1176<main+61> = 0 = EBP** allora **jge** salta alla locazione di memoria specifica.

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

Questo codice sposta il valore **0** (sorgente) nel registro **eax** quindi **eax = 0.**

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

In questo caso la funzione chiamante passa l'esecuzione alla funzione chiamata.

PRATICA S10-L3