SC 11/3/2026

mercoledì 11 marzo 2026 10:10

ISTRUZIONE TEST

CMP S , D | D−S

Se dobbiamo fare una comparazione con 0

CMP 0 , D | D−0

TEST D , D | D & D

TRADUZIONE IN ASM DI CICLI

WHILE

CODICE C	CODICE C EQ 1	CODICE C EQ 2	CODICE ASM

while (test) {
blocco A
}
blocco B
	L:
	if (test) {
	blocco A
	goto L;
	}
	blocco B
		L:
		if (!Test) goto E;
		blocco A
		goto L;
		E:
		blocco B
			L:
			cmp S , D
			Jcc E
			blocco A
			jmp L
			E:
			blocco B

FOR

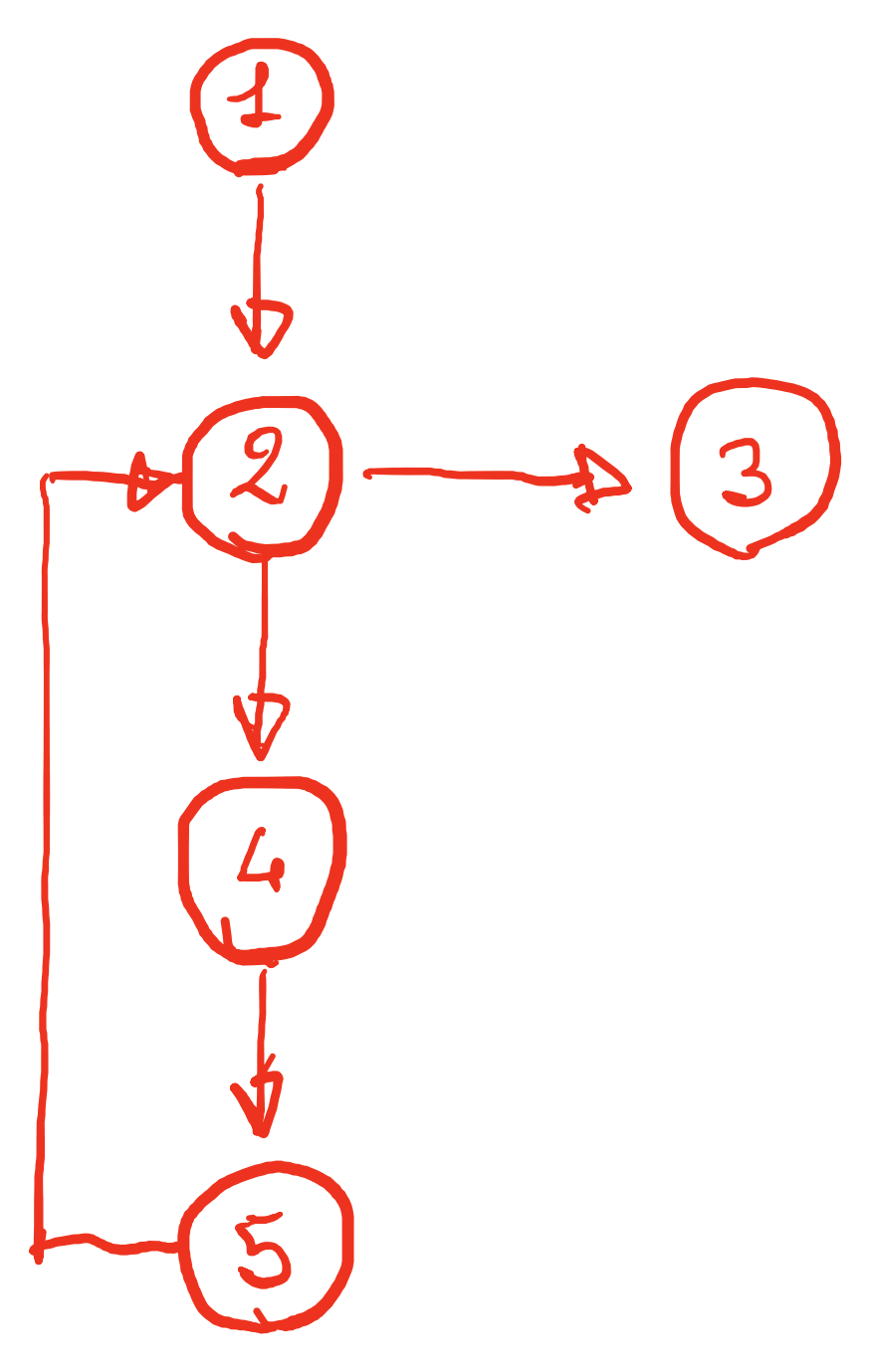
① ② ⑤

for (init ; test ; update) {

④ blocco A

}

③ blocco B



CODICE C	CODICE C EQ 1	- - - -

for (init ; Test ; update) {
blocco A
}
blocco B
	init ;
	while (test) {
	blocco A
	update ;
	}
	blocco B
		- - - -

GESTIONE DEI PUNTATORI IN ASM

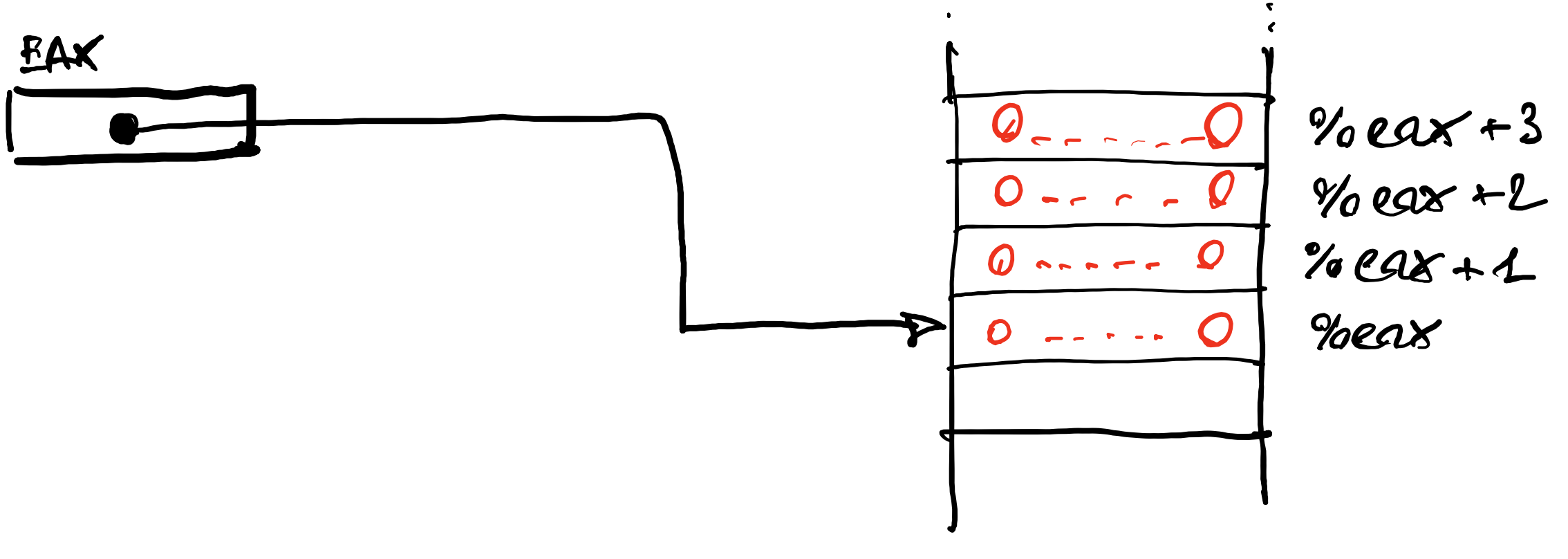
Un puntatore in C di tipo T* contiene l'indirizzo di memoria dove è conservato un valore di Tipo T

int * a; ⇒ # int * a → EAX

*a = 0; ⇒ movl $0 , ~~%eax~~

In ASM per indirizzare una locazione di memoria il cui indirizzo è contenuto in un registro usiamo l'operatore "( )"

movl $0 , (%eax)



int c = *a; ⇒ movl (%eax) , %ecx

questo dipende da sizeof (T)

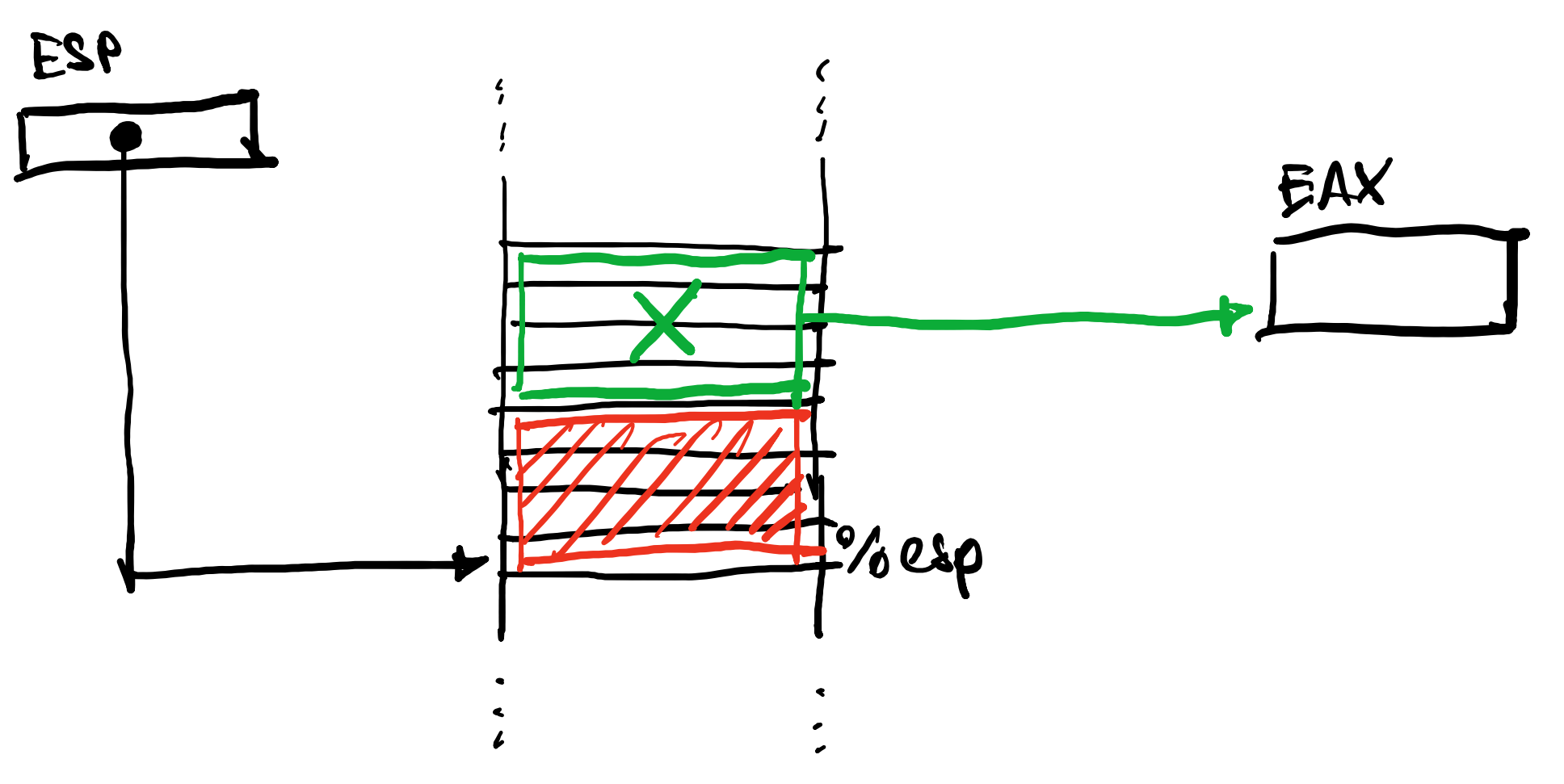
Es.: char * p; # char *p → EDX

char a = * p ; movb (%edx) , %al

Quando accediamo al parametro passato per valore ad una funzione, stiamo in effetti usando un puntatore con "( )"

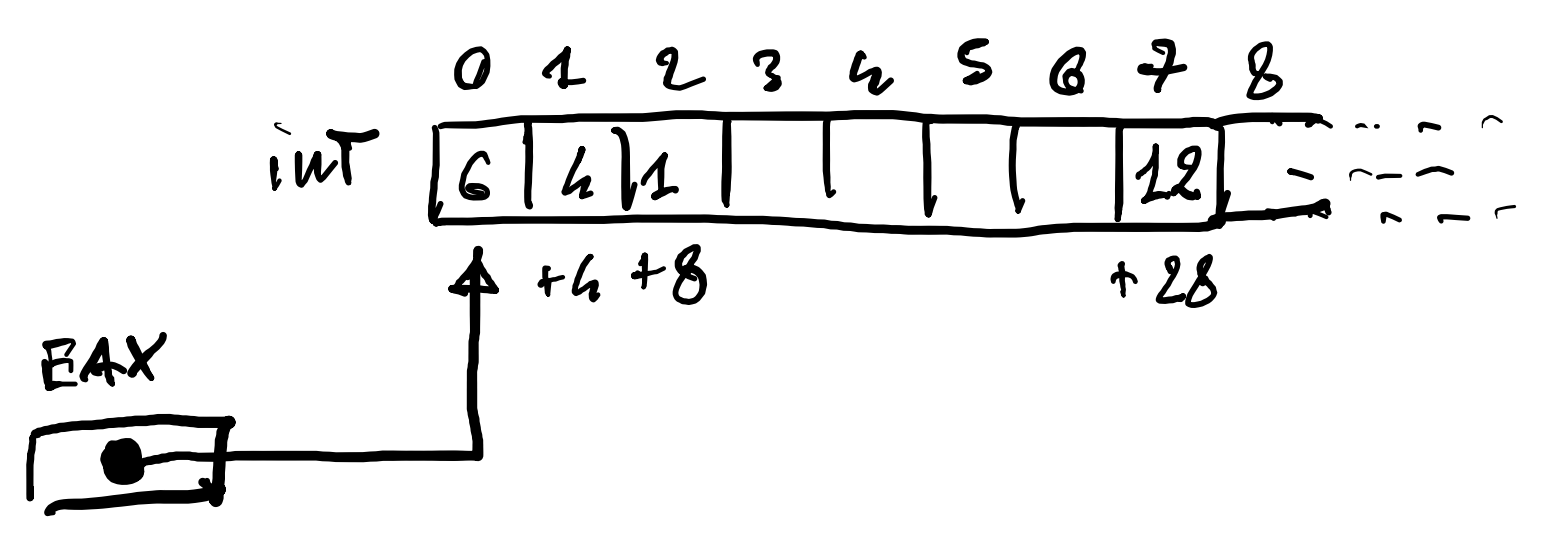
void f(int x) {

int a = x; movl 4(%esp) , %eax



Es.:

int a[7] = 12 ;



movl $12 , 28(%eax)

Es.:

void f(char * x) {

char c = x[1]; ⇒ movb (~~4(%esp)~~) , %cl

IMPORTANTE: In una istruzione IA32 è possibile l'accesso a memoria per al più un operando

movl 4(%esp) , %eax

movb 1(%eax), %cl

IMPORTANTE: l'operatore "( )" può essere usato solo sui registri e questi registri devono contenere indirizzi validi di memoria

ACCESSO IN MEMORIA CON BASE, INDICE E SPIAZZAMENTO

D (X) → accede alla locazione con indirizzo X+D

La sintassi completa dell'operatore "( )" è

IMM (BASE , INDICE , SCALA)

- IMM: spiazzamento definito come valore costante
- BASE: indirizzo di memoria di base a cui accedere. DEVE essere un registro. DEVE essere valido. OBBLIGATORIO
- INDICE: indice dell'elemento a cui si vuole accedere a partire da BASE. DEVE essere un registro. OPZIONALE
- SCALA: dimensione in byte di ogni elemento. DEVE essere 1, 2, 4, 8 (8: solo a 64 bit). OPZIONALE

L'uso dell'operatore permette di accedere alla memoria con indirizzo

indirizzo = BASE + INDICE * SCALA + IMM

Es.:

int a[7] = 12 ⇒

```
movl $7 , %ecx
movl $12, (%eax , %ecx , 4)
```

corrisponde a movl $12, 28(%eax)