

long add5 (long b0, long b1, long b2, long b3, long b4) { return b0+b1+b2+b3+b4; }

```

mov %rdi, %rax
add %rsi, %rax
add %rdx, %rax
add %rcx, %rax
add %r8, %rax
ret

```

0

```

add %rdi, %rsi
add %rdx, %rcx
add %rsi, %rcx
add %r8, %rcx
mov %rcx, %rax
ret

```

Se aplica de forma inversa

long add10 (long a0, long a1, long a2, long a3, long a4, long a5, long a6, long a7, long a8, long a9

// Interpreto que a6, a7, a8 y a9 interpretan esos orden

```

push %rbx
mov %rsp, %rbp # guarda rsp inicial
call add5 # tengo los 5 primeros args en donde quiero.

```

```

mov %rax, %rbx
mov %r9, %rdi
mov 8(%rsp), %rsi
mov 16(%rsp), %rdx
mov 24(%rsp), %rcx
mov 32(%rsp), %r8

```

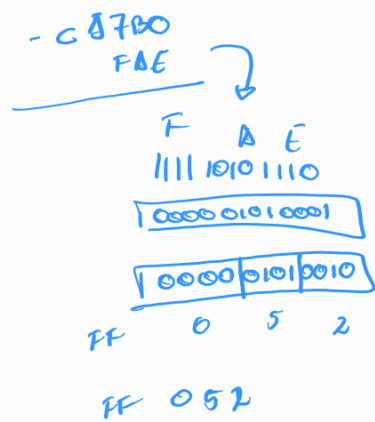
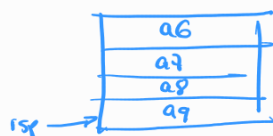
```

call add5
add %rax, %rbx
pop %rbx
ret

```

# argumentos

# llamo a la función  
# sumo resultado  
# retorno mi valor de %rbx  
# termino la función



Si sale 0 tenemos complemento a 1  
1 1  
C87B0  
FF 052  
0C9 802  
positivo