

M6 D5

ESERCIZIO 2

ASSEMBLY - C

Ricostruzione delle istruzioni originali da linguaggio Assembly a linguaggio C

CODICE ASSEMBLY

```
push %ebp          ; Salva l'indirizzo precedente del puntatore base
mov %esp, %ebp     ; Imposta il puntatore base all'indirizzo corrente dello stack
sub $0x8, %esp      ; Assegna 8 byte allo stack
call 80483E9 <bar>   ; Chiama la funzione bar
leave              ; Ripristina l'indirizzo precedente del puntatore base
ret                ; Torna alla funzione chiamante
```

; La funzione bar non chiama altre funzioni.

```
push %ebp          ; Salva l'indirizzo precedente del puntatore base
mov %esp, %ebp     ; Imposta il puntatore base all'indirizzo corrente dello stack
sub $0x8, %esp      ; Assegna 8 byte allo stack
call 80483FB <baz>   ; Chiama la funzione baz
call 8048400 <quux>  ; Chiama la funzione quux
leave              ; Ripristina l'indirizzo precedente del puntatore base
ret                ; Torna alla funzione chiamante
```

; La funzione baz chiama la funzione quux.

; La funzione quux non chiama altre funzioni.

TRASPOSIZIONE IN LINGUAGGIO C

```
int bar(int x) {

    // Questa funzione prende in input un intero e restituisce l'intero sommato di 1.

    return x + 1;

}

int baz(int x, int y) {

    // Questa funzione prende in input due interi e restituisce la somma di entrambi.

    return x + y;

}
```

CODICE ASSEMBLY

```
push %ebp          ; Salva l'indirizzo precedente del puntatore base
mov %esp, %ebp     ; Imposta il puntatore base all'indirizzo corrente dello stack
pop %ebp           ; Ripristina l'indirizzo precedente del puntatore base
ret                ; Torna alla funzione chiamante
```

; La funzione `foo` è una funzione vuota.

```
push %ebp          ; Salva l'indirizzo precedente del puntatore base
mov %esp, %ebp     ; Imposta il puntatore base all'indirizzo corrente dello stack
mov $0x0, %eax     ; Imposta eax a 0
movl $0x1, (%eax)  ; Salva il valore 1 nella posizione di memoria puntata da eax
pop %ebp           ; Ripristina l'indirizzo precedente del puntatore base
ret                ; Torna alla funzione chiamante
```

; La funzione bar salva il valore del puntatore base sullo stack e imposta il puntatore base all'indirizzo corrente dello stack.

; Esegue poi la funzione foo, ripristina il puntatore base e torna alla funzione chiamante.

; Esegue poi la funzione foo, ripristina il puntatore base e torna alla funzione chiamante.

```
push %ebp          ; Salva l'indirizzo precedente del puntatore base
mov %esp, %ebp     ; Imposta il puntatore base all'indirizzo corrente dello stack
and $0xffffffff0, %esp ; Allinea il stack a 16 byte
call 8043DC <foo>   ; Esegue la funzione foo
mov $0x0, %eax     ; Imposta eax a 0
leave              ; Ripristina l'indirizzo precedente del puntatore base
ret                ; Torna alla funzione chiamante
```

TRASPOSIZIONE IN LINGUAGGIO C

// Questo codice dichiara una funzione chiamata foo e una funzione principale.

// La funzione foo inizializza una variabile x a 0 e quindi la imposta a 1.

// La funzione principale chiama la funzione foo e quindi restituisce 0.

```
void foo() {
```

```
    // Inizializza la variabile x a 0
```

```
    int x = 0;
```

```
    // Imposta la variabile x a 1
```

```
    x = 1;
```

```
}
```

```
int main() {
```

```
    // Chiama la funzione foo
```

```
    foo();
```

```
    // Restituisce 0.
```

```
    return 0;
```

```
}
```