

**Corso di Calcolatori Elettronici I**  
**A.A. 2012-2013**

---

---

# **Strutture di controllo in assembly**

**ing. Alessandro Cilaro**

Accademia Aeronautica di Pozzuoli  
Corso Pegaso V "GArn Elettronici"

---

# Costrutti per il controllo di flusso

---

- Le strutture di controllo normalmente utilizzate nei linguaggi di alto livello (ad esempio il C/C++) sono automaticamente tradotte dal compilatore in sequenze di istruzioni assembler

- Strutture di controllo in C/C++:

```
if ( condizione ) {...}
```

```
if ( condizione ) {...} else {...}
```

```
do{...} while(condizione )
```

```
while( condizione ) {...}
```

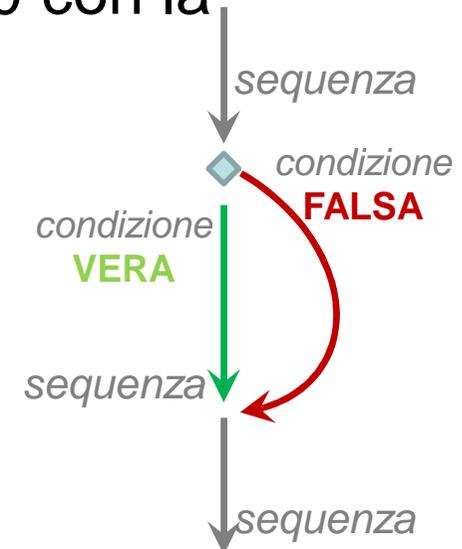
```
for( ...; condizione; ...) {...}
```

# Costrutto di selezione ( *if* )

- `if ( condizione ) { istruzioni }`

può essere resa nell'assembler del 68000 con la sequenza:

```
B(NOT condizione) SKIP
    istruzione
    . . .
SKIP istruzione_successiva
```



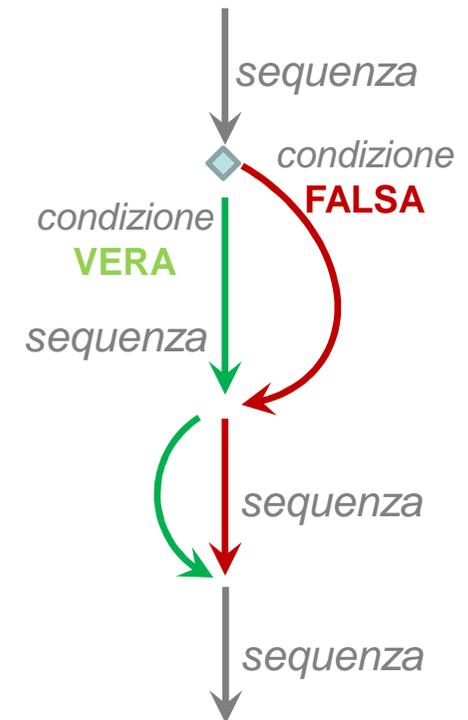
Esempio:

C++	<pre>if (D0 == 5){     D1++; } D2 = D0; . . .</pre>	asm	<pre>CMPI.L #5,D0 BNE     SKIP ADDQ.L #1,D1 SKIP   MOVE.L D0,D2 . . .</pre>
-----	---	-----	---

# Costrutto *if ... else*

- `if ( condizione ) { istruzioni } else { istruzioni }`  
può essere resa nell'assembler del 68000 con la sequenza:

```
B(NOT condizione) SKIP1
    istruzione1
    ...
    BRA SKIP2
SKIP1  istruzione2
    ...
SKIP2  istruzione_successiva
```



# Costrutto *if ... else*

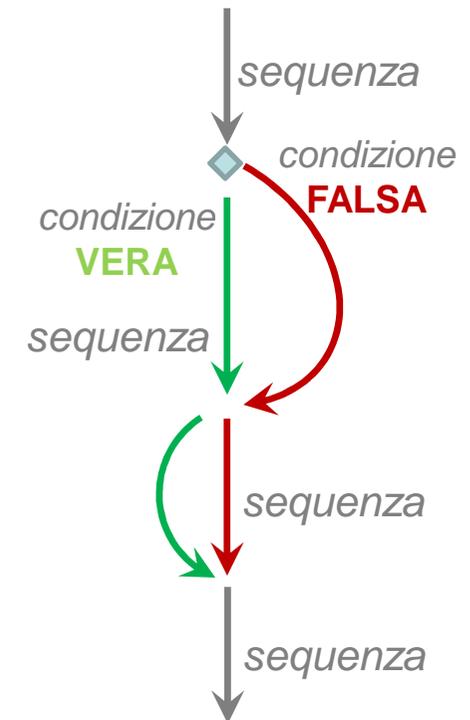
Esempio:

C++

```
if (D0 == 5){  
    D1++;  
}else{  
    D1--;  
}  
D2 = D0;
```

asm

```
. . .  
CMPI.L #5,D0  
BNE    SKIP1  
ADDQ.L #1,D1  
BRA    SKIP2  
SKIP1  ADDQ.L #-1,D1  
SKIP2  MOVE.L D0,D2  
. . .
```

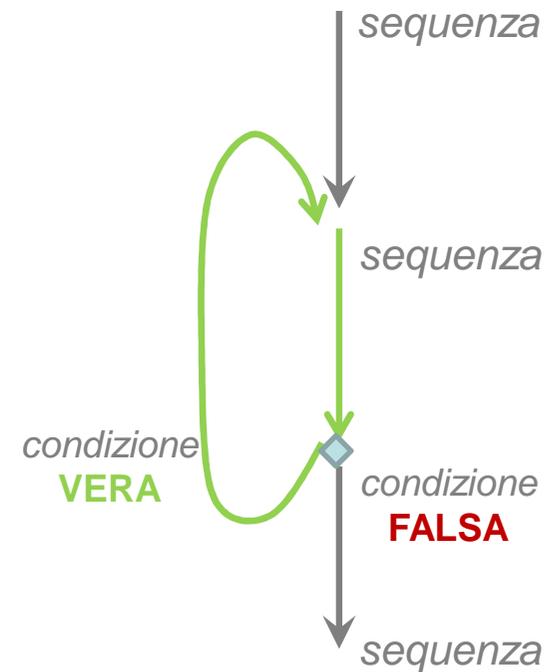


# Costrutto *do ... while*

---

- `do {istruzioni} while( condizione )`  
può essere resa nell'assembler del 68000 con la sequenza:

```
LOOP      istruzione1  
          ...  
          Bcc LOOP  
          istruzione_successiva
```



# Costrutto *do ... while*

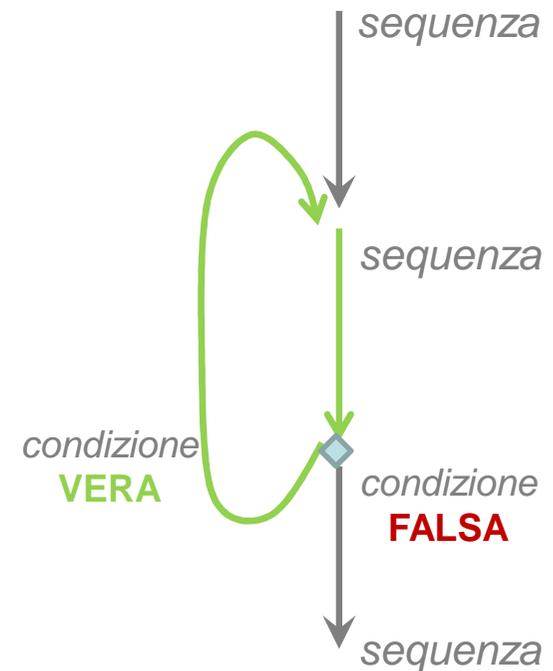
Esempio: calcola  $3^N$  ( $N > 0$ )

C++

```
D0 = 1;  
D1 = 1;  
do{  
    D0 = D0 * 3;  
    D1++;  
}while (D1 <= N);
```

asm

```
MOVE.B #N,D2  
MOVE.B #1,D1  
MOVE.W #1,D0  
LOOP  MULU.W #3,D0  
      ADDQ.B #1,D1  
      CMP.B  D2,D1  
      BLE   LOOP
```



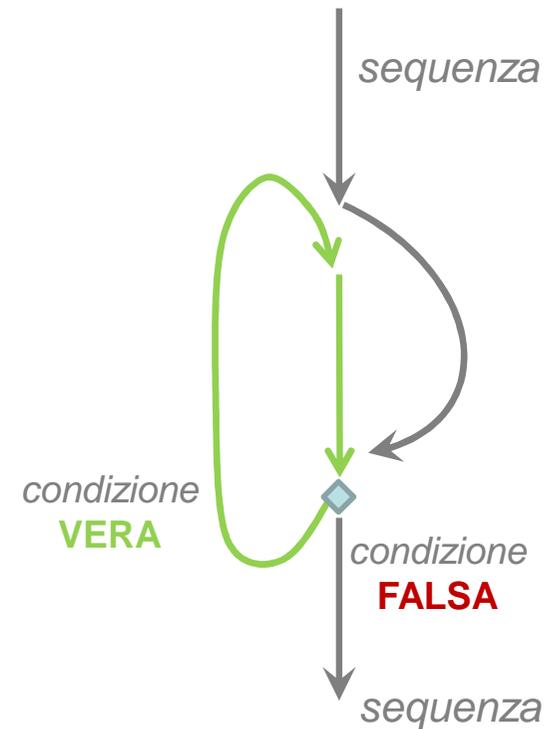
# Costrutto *while*

---

- `while( condizione ) {istruzioni}`

può essere resa nell'assembler del 68000 con la sequenza:

```
                BRA  INIT
LOOP            istruzione
                ...
INIT           Bcc  LOOP
                istruzione_successiva
```



# Costrutto *while*

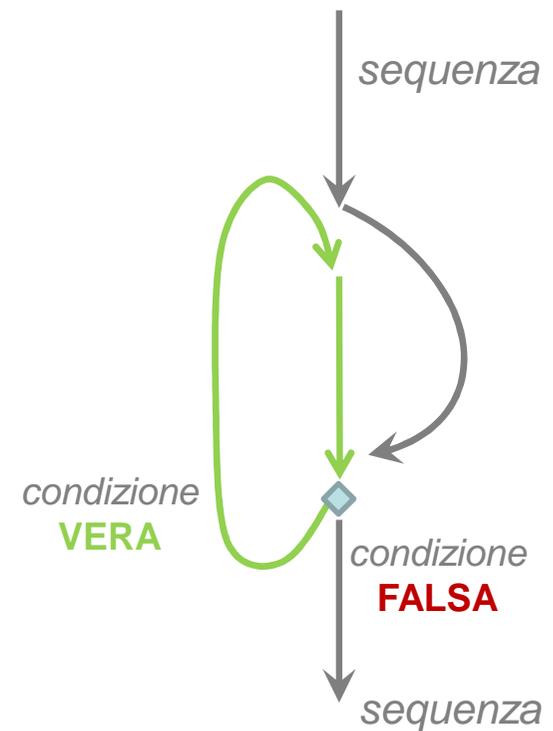
Esempio: calcola  $3^N$  ( $N > 0$ )

C++

```
D0 = 1;
D1 = 1;
while(D1 <= N) {
    D0 = D0 * 3;
    D1++;
}whil;
```

asm

```
MOVE.B #N,D2
MOVE.B #1,D1
MOVE.W #1,D0
BRA     INIT
LOOP   MULU.W #3,D0
      ADDQ.B #1,D1
INIT   CMP.B  D2,D1
      BLE   LOOP
```

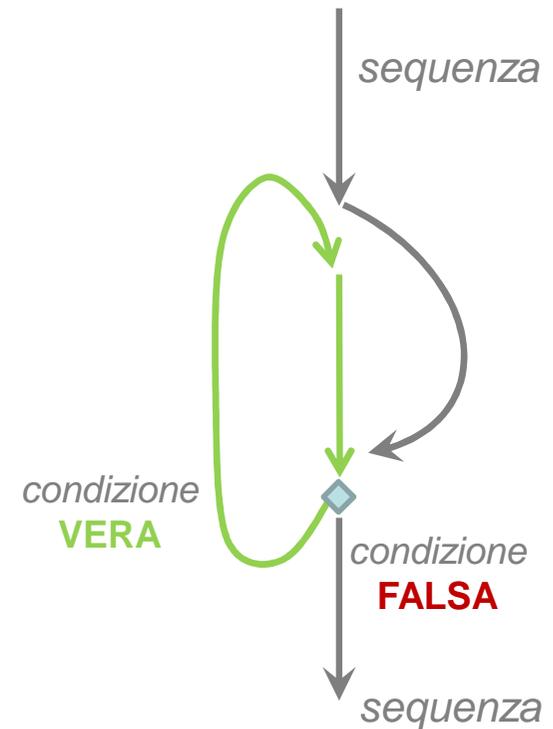


# Costrutto *for*

- `for(int i=0; i<N; i++) {istruzioni}` corrisponde all'uso del costrutto `while` effettuato nel seguente modo:

```
int i=0;
while( i<N){
    istruzioni
    i++;
}
```

e può essere resa nell'assembler del 68000 nello stesso modo in cui si rende il costrutto `while`



# Esercizio

---

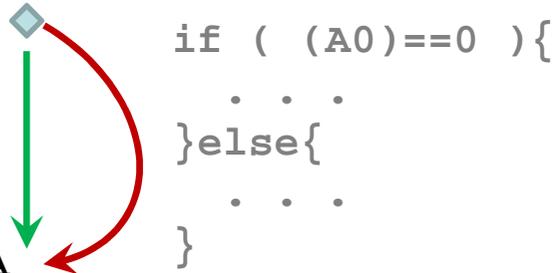
---

- Analizzare tutti i programmi precedentemente sviluppati e riconoscere i costrutti condizionali ed iterativi di volta in volta utilizzati

# Esempio

File: programma018.a68

```
ORG      $8000
START    MOVEA.L  #STRING,A0
         MOVE.B   #TOKEN,D0
LOOP     TST.B    (A0)
         BEQ     DONE
         CMP.B   (A0)+,D0
         BNE    LOOP
FOUND   SUBQ.L   #1,A0
DONE    MOVE.L   A0,TOKENA
        STOP    #$2700
ORG     $8100
TOKEN   EQU     ':'
STRING  DC.B    'QUI QUO:QUA',0
TOKENA  DS.L    1
END     START
```



```
if ( (A0)==0 ) {
    . . .
} else {
    . . .
}
```

