

---

## Nested if-statements

---

- **Observation**

- The **if-statement**:

```

if ( C )
{
    s1;
    s2;
    ...
}

```

give rise to an **assembler program** with the following **structure**:

```

instructions to perform a compare
specified by the condition C

branch of FALSE of the condition C to label IfEnd

instructions to perform s1
instructions to perform s2
....
IfEnd:

```

- The **if-else-statement**:

```

if ( C )
{
    s1;
    s2;
    ...
}
else
{
    t1;
    t2;
    ...
}

```

give rise to an **assembler program** with the following **structure**:

```

instructions to perform a compare
specified by the condition C

branch of FALSE of the condition C to label Else

instructions to perform s1
instructions to perform s2
....
bra IfEnd

```

```

Else:
    instructions to perform t1
    instructions to perform t2
    ....
IfEnd:

```

○ Note:

- The **statements** (s1, s2, ... t1, t2, ...) inside the **then/else parts** can be **"translated"** into **assembler code** *independently* from the **if-statement**

- Use this **property** to handle:

- **Nesting** of an **if/if-else** statement inside another **if/if-else** statement

• Nesting of if-statements

○ Example:

```

// x = max of a and b when c > 0
//   otherwise, x = min of a and b

int x, a, b, c;

if ( c > 0 )
{
    // Find maximum
    if ( a > b )
    {
        x = a;
    }
    else
    {
        x = b;
    }
}
else
{
    // Find minimum
    if ( a < b )
    {
        x = a;
    }
    else
    {
        x = b;
    }
}

```

• Modular approach

○ Method:

- Work from the **outside inwards**

- Ignore the **inner section (body of the then and else)** of the **if-statement at first**
- After coding the **outer if-statement**, write the code of the statements inside the **then part** and **else part**

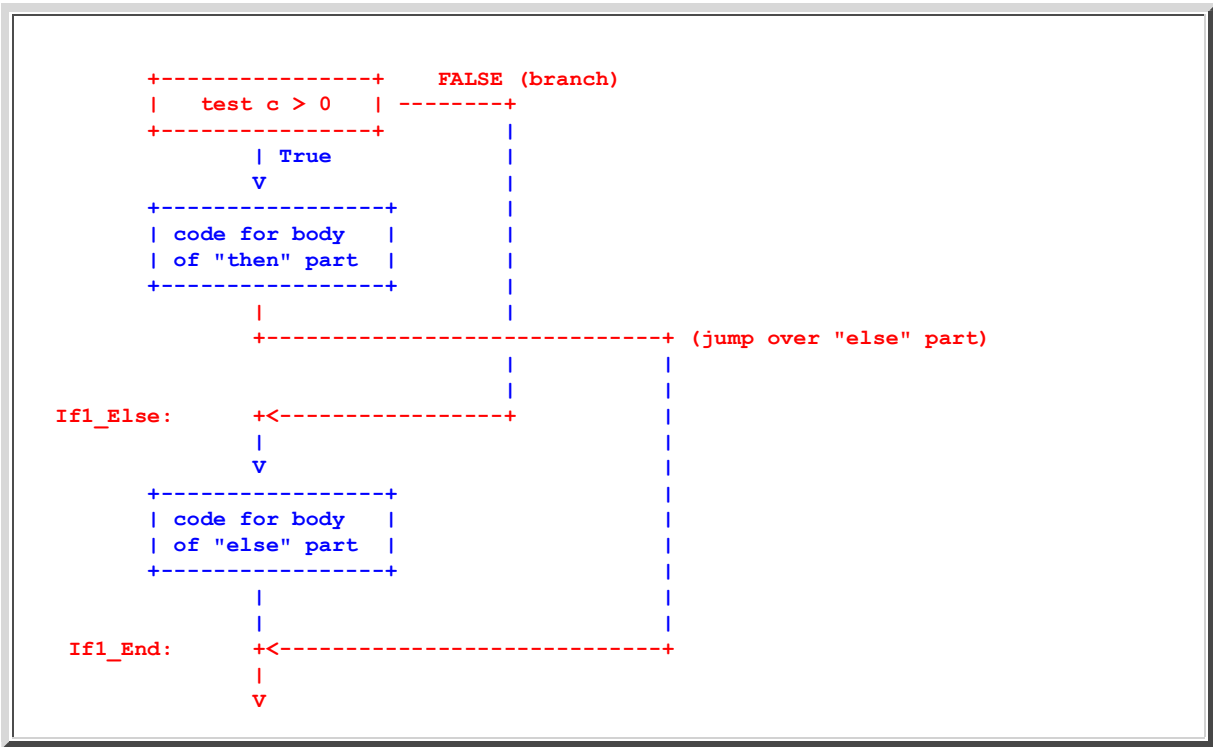
• **Coding the outer if-statement**

- **Example with the then (and else) body stripped:**

```

if ( c > 0 )
{
    "then" body stripped
}
else
{
    "else" body stripped
}
    
```

- **Corresponding program flow chart:**



- **Corresponding assembler code:**

```

move.l c,d0
cmp.l #0,d0      Tests: c ?? 0

ble    Ifl_Else  Branch when !(c < 0) <=> c <= 0
    
```

```

.....
code for body of "then" part
.....

bra    If1_End

If1_Else:
.....
code for body of "else" part
.....

If1_End:

```

- Next: code the body of the "then" (and "else") part (individually)

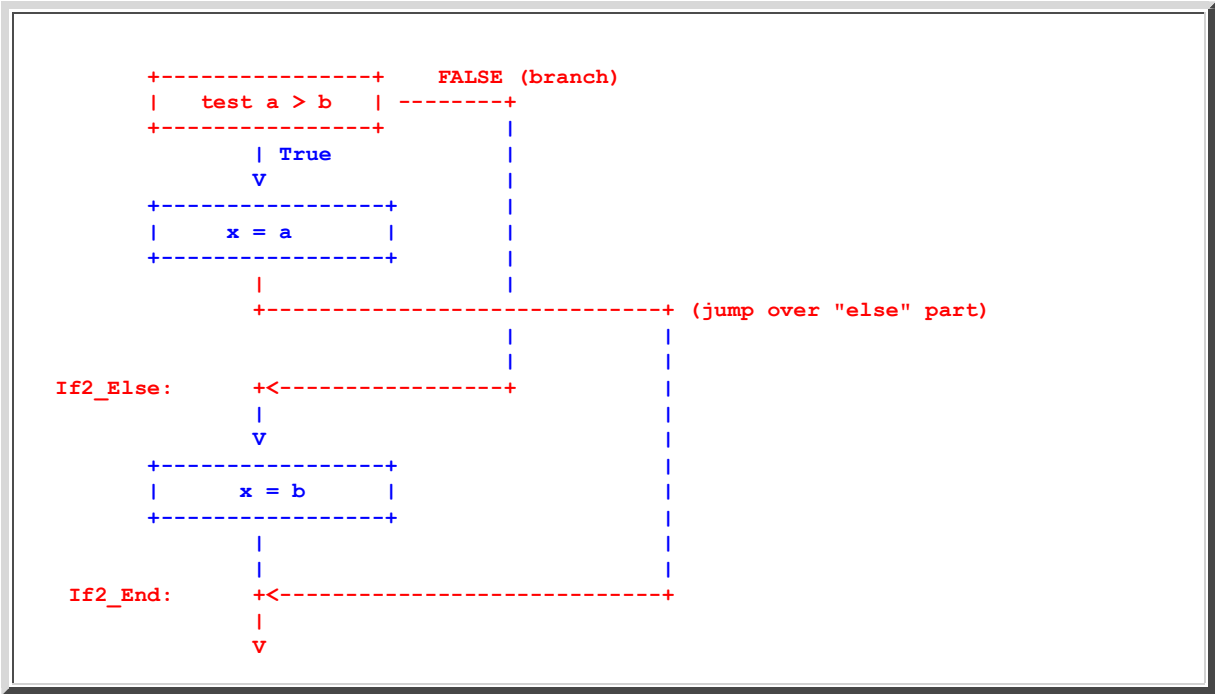
- Body of the "then" part (ignore the *outer* if-statement !):

```

if ( a > b )
{
    x = a;
}
else
{
    x = b;
}

```

- Corresponding program flow chart:



- Corresponding assembler code:

```

move.l a,d0
cmp.l b,d0      Tests: a ?? b

```

```

        ble    If2_Else    Branch when !(a > b) <=> a <= b

        move.l a,d0
        move.l d0,x

        bra    If2_End

If2_Else:

        move.l b,d0
        move.l d0,x

If2_End:
    
```

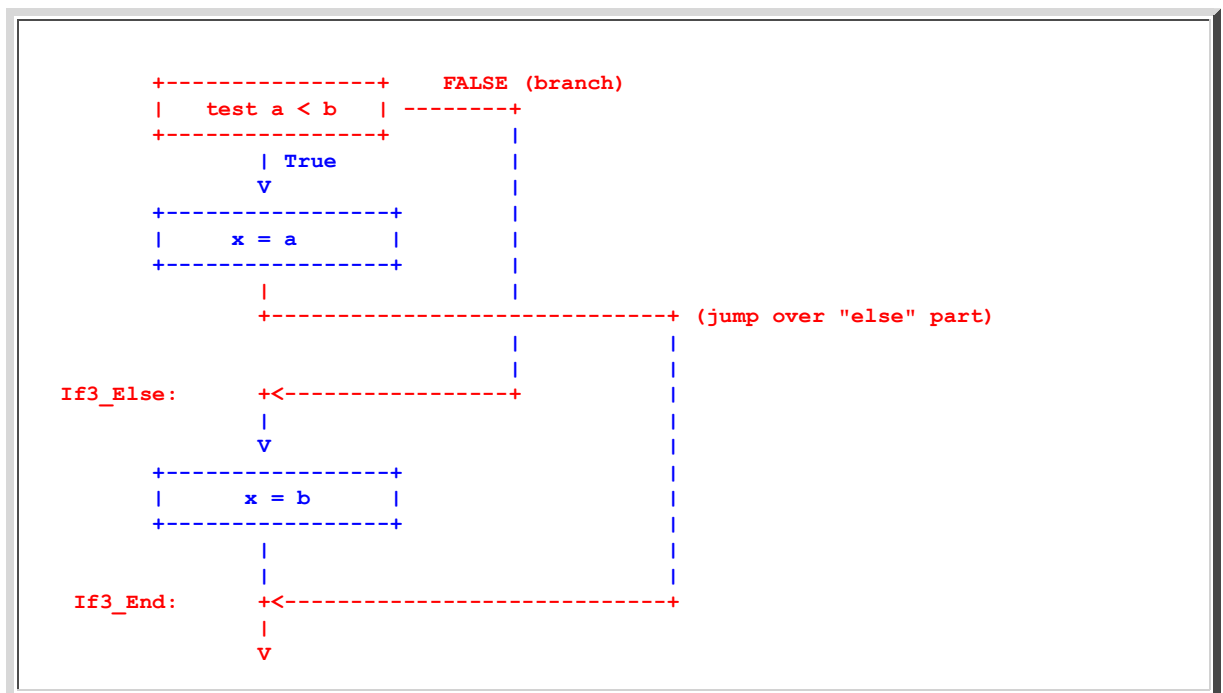
• Continued....: code the body of the "else" part (individually)

- Body of the "else" part (ignore the *outer* if-statement !):

```

if ( a < b )
{
    x = a;
}
else
{
    x = b;
}
    
```

- Corresponding program flow chart:



- Corresponding assembler code:

```

    
```

```

move.l a,d0
cmp.l b,d0      Tests: a ?? b

bge   If3_Else  Branch when !(a < b) <=> a >= b

move.l a,d0
move.l d0,x

bra   If3_End

If3_Else:

    move.l b,d0
    move.l d0,x

If3_End:

```

- Finally, put the "then" and "else" parts into their places

- Originally:

```

move.l c,d0
cmp.l #0,d0     Tests: c ?? 0

ble   If1_Else  Branch when c <= 0

.....
code for body of "then" part
.....

bra   If1_End

If1_Else:

.....
code for body of "else" part
.....

If1_End:

```

- After inserting the code for the "then" body:

```

move.l c,d0
cmp.l #0,d0     Tests: c ?? 0

ble   If1_Else  Branch when c <= 0

-----

move.l a,d0
cmp.l b,d0     Tests: a ?? b

ble   If2_Else  Branch when a <= b

move.l a,d0
move.l d0,x

bra   If2_End

```

```

If2_Else:

    move.l b,d0
    move.l d0,x

If2_End:
-----

    bra    If1_End

If1_Else:

    .....
    code for body of "else" part
    .....

If1_End:

```

- Finally, after inserting the code for the "else" body:

```

    move.l c,d0
    cmp.l #0,d0      Tests: c ?? 0

    ble   If1_Else   Branch when c <= 0
-----

    move.l a,d0
    cmp.l b,d0      Tests: a ?? b

    ble   If2_Else   Branch when a <= b

    move.l a,d0
    move.l d0,x

    bra   If2_End

If2_Else:

    move.l b,d0
    move.l d0,x

If2_End:
-----

    bra   If1_End

If1_Else:
-----

    move.l a,d0
    cmp.l b,d0      Tests: a ?? b

    bge   If3_Else   Branch when a >= b

    move.l a,d0
    move.l d0,x

```

```
bra    If3_End

If3_Else:

    move.l b,d0
    move.l d0,x

If3_End:

-----

If1_End:
```

○ **Note:**

- you may have **multiple labels** marking the **same location**  
E.g.: **If1\_End** and **If3\_End** mark the same location in the program....
- That's OK, it **will not compromise** the **correctness** of the code
- (Besides, compilers do that all the time)