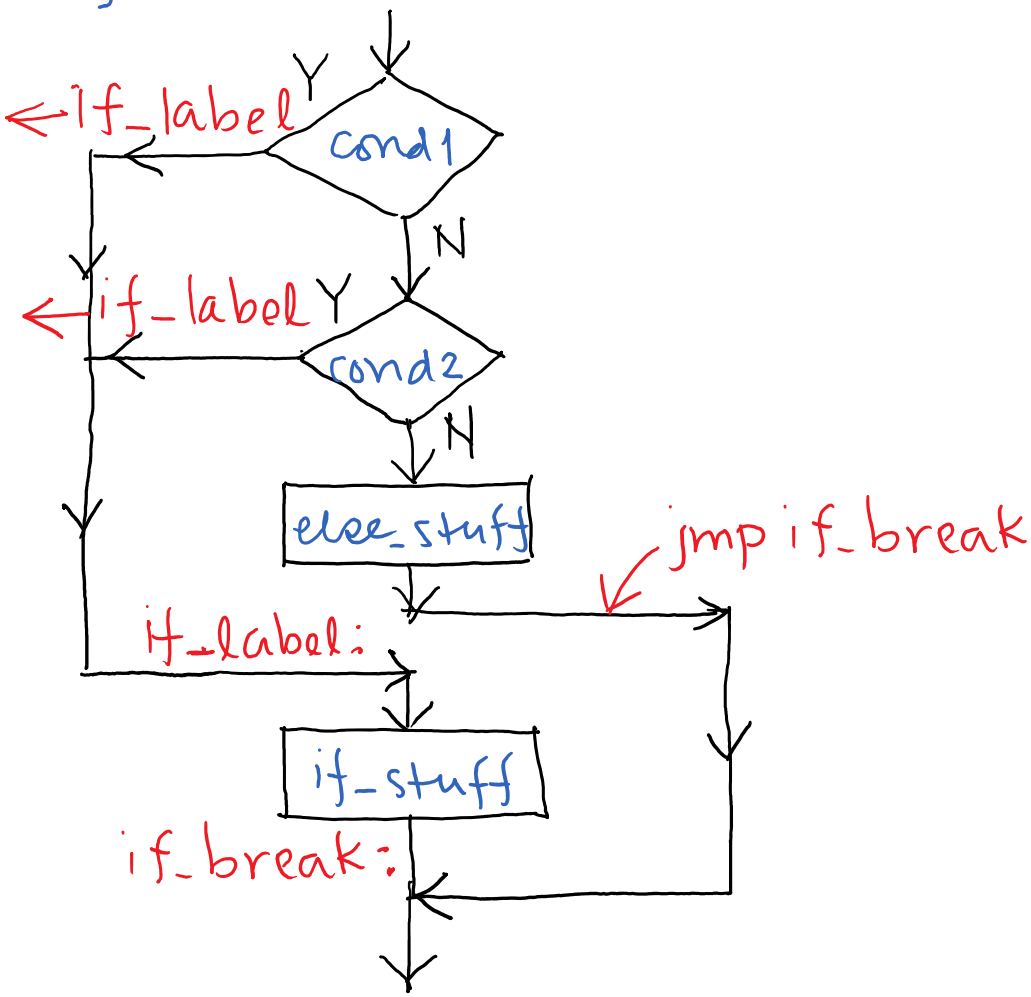
Logical OR

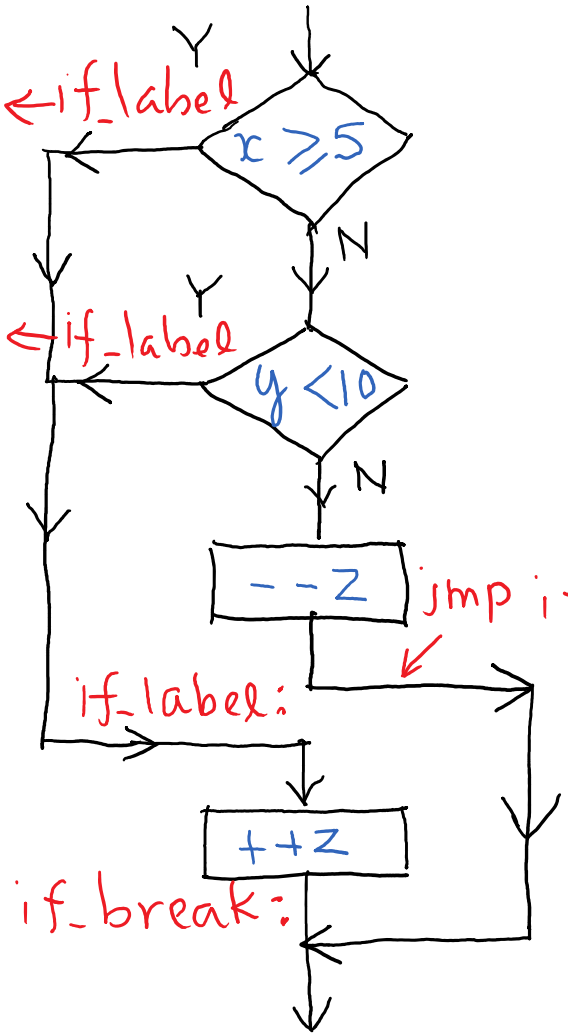
if-structure with Cond1 || cond2

```
if (cond1 || cond2)
{
   if-stuff
}
else
{
   else-stuff
}
```



Example:

```
if( x >=5 || y <10 )
{
    ++z
}
else
{
    --z
}
```

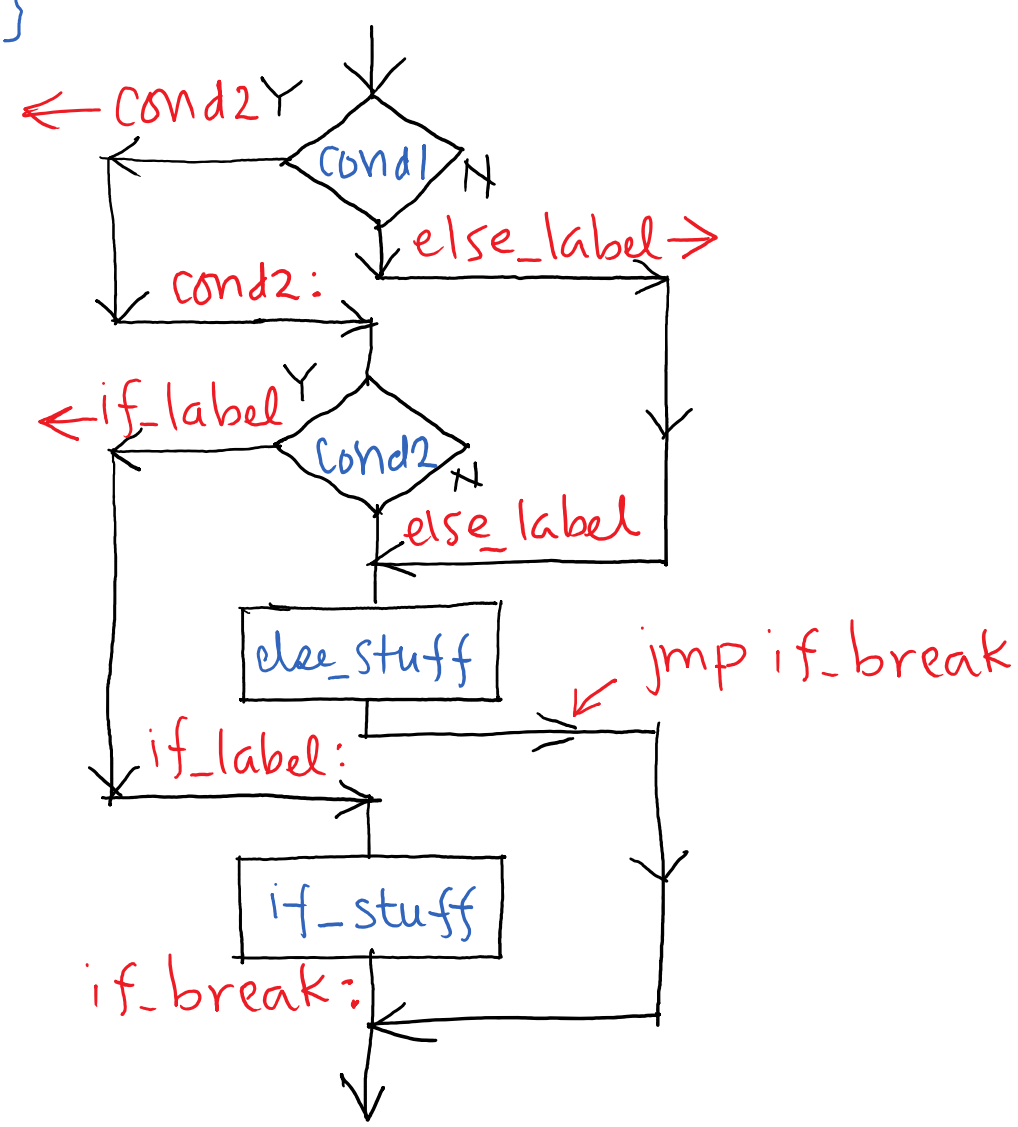



```
    cmp.w #5, &x
    jge if_label

    cmp.w #10, &y
    jl if_label

    inc.w &z

    jmp if_break
if_label:
    dec.w &z
if_break:
```

Logical AND

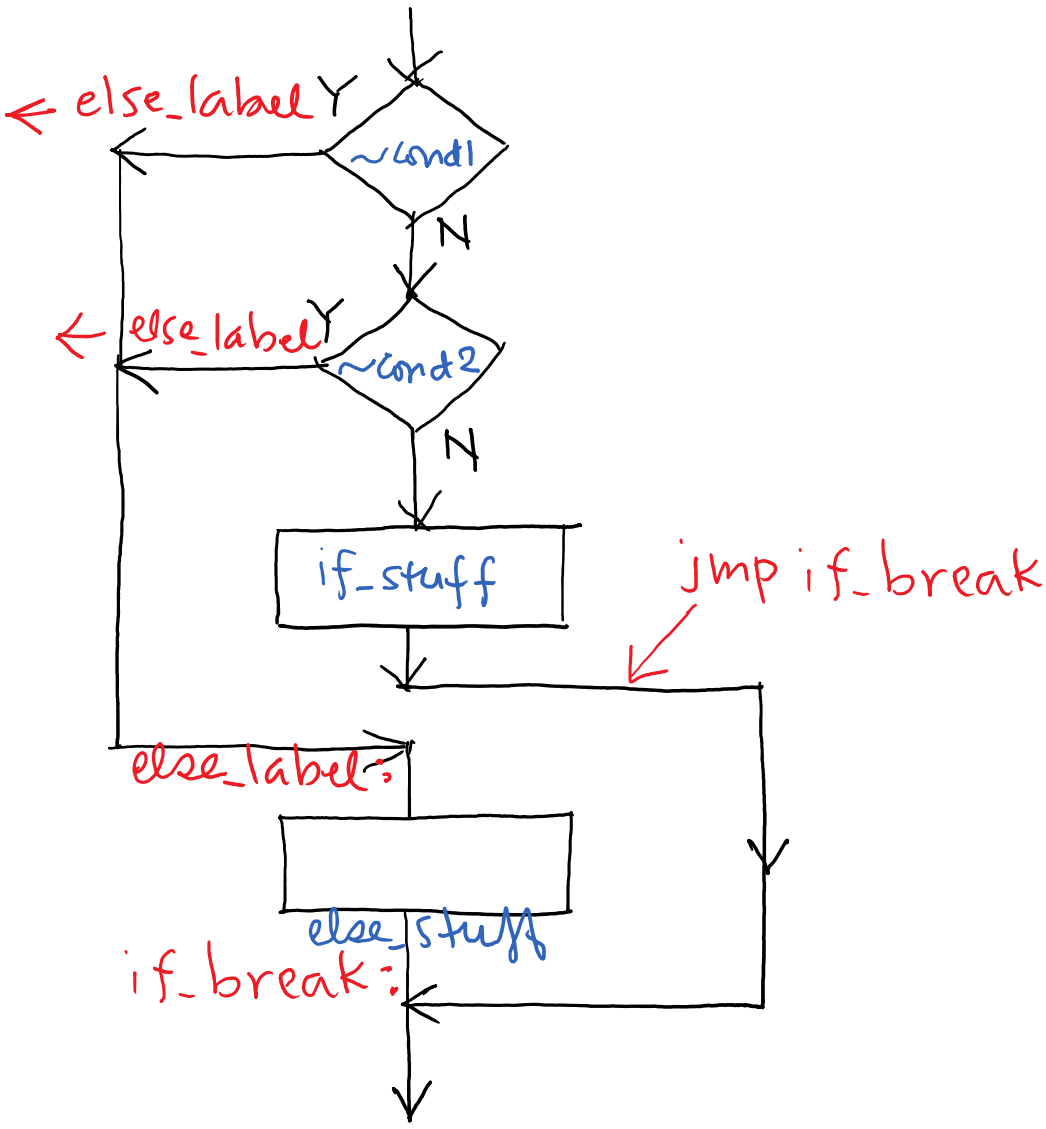
if structure with cond1 && cond2

```
if ( cond1 && cond2)
{
    if_stuff
}
else
{
    else_stuff
}
```



cond1 && cond2 alternate implementation

```
if (cond1 && cond2)
{
    if_stuff
}
else
{
    else_stuff
}
```



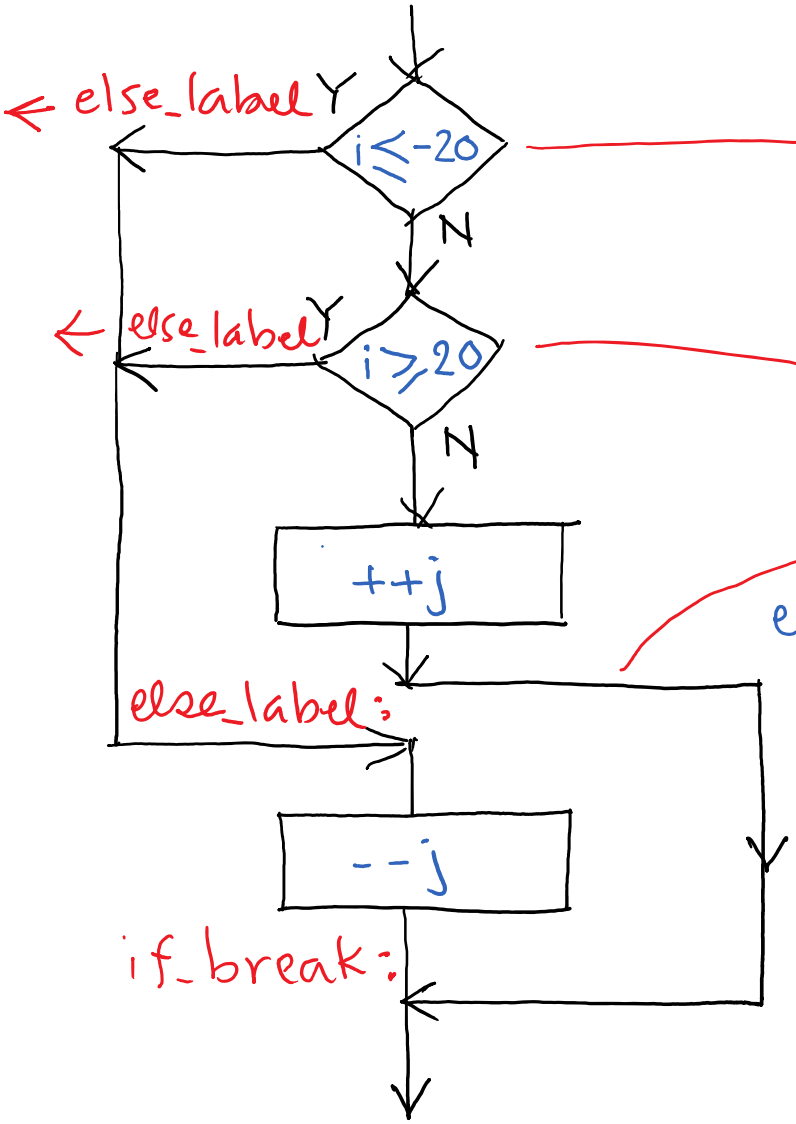
$-20 < i < 20$

Example:

```
if ( i > -20 && i < 20)
{
    ++j
}
else
{
    --j
}
```

$\sim(i > -20) = i \leq -20$

$\sim(i < 20) = i \geq 20$




```
mov.W #-20, R5
cmp.W &i, R5
jge else_label
cmp.W #20, &i
jge else_label
inc.W &j
jmp if_break
else_label:
dec.W &j
if_break:
```

Switch_case structure

```
switch (x)
{
  case 3:
    stuff1
    break
  case 4:
    stuff2
    break
  default:
    default_stuff
    break;
}
```

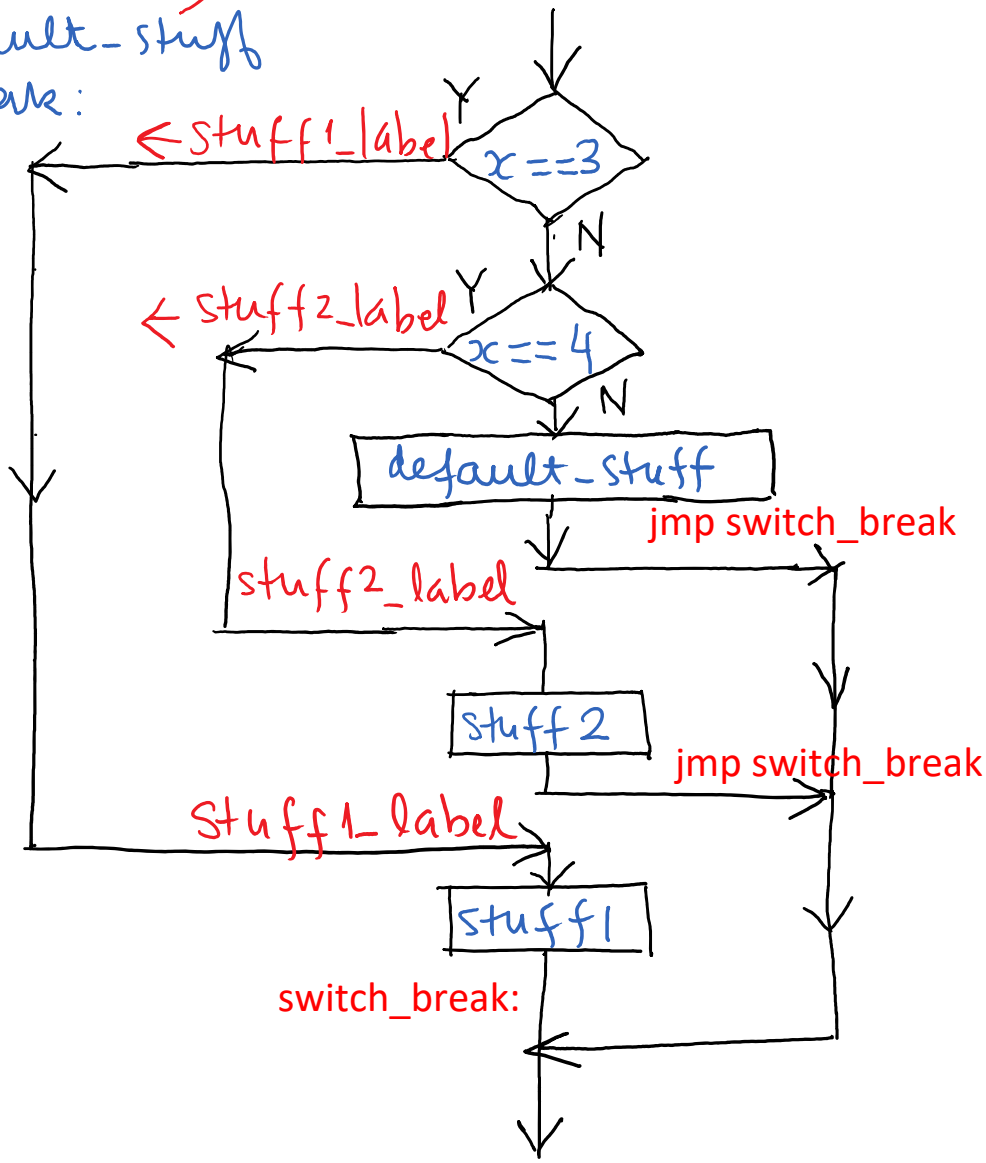
x==3?

do stuff1 if x == 3

x==4?

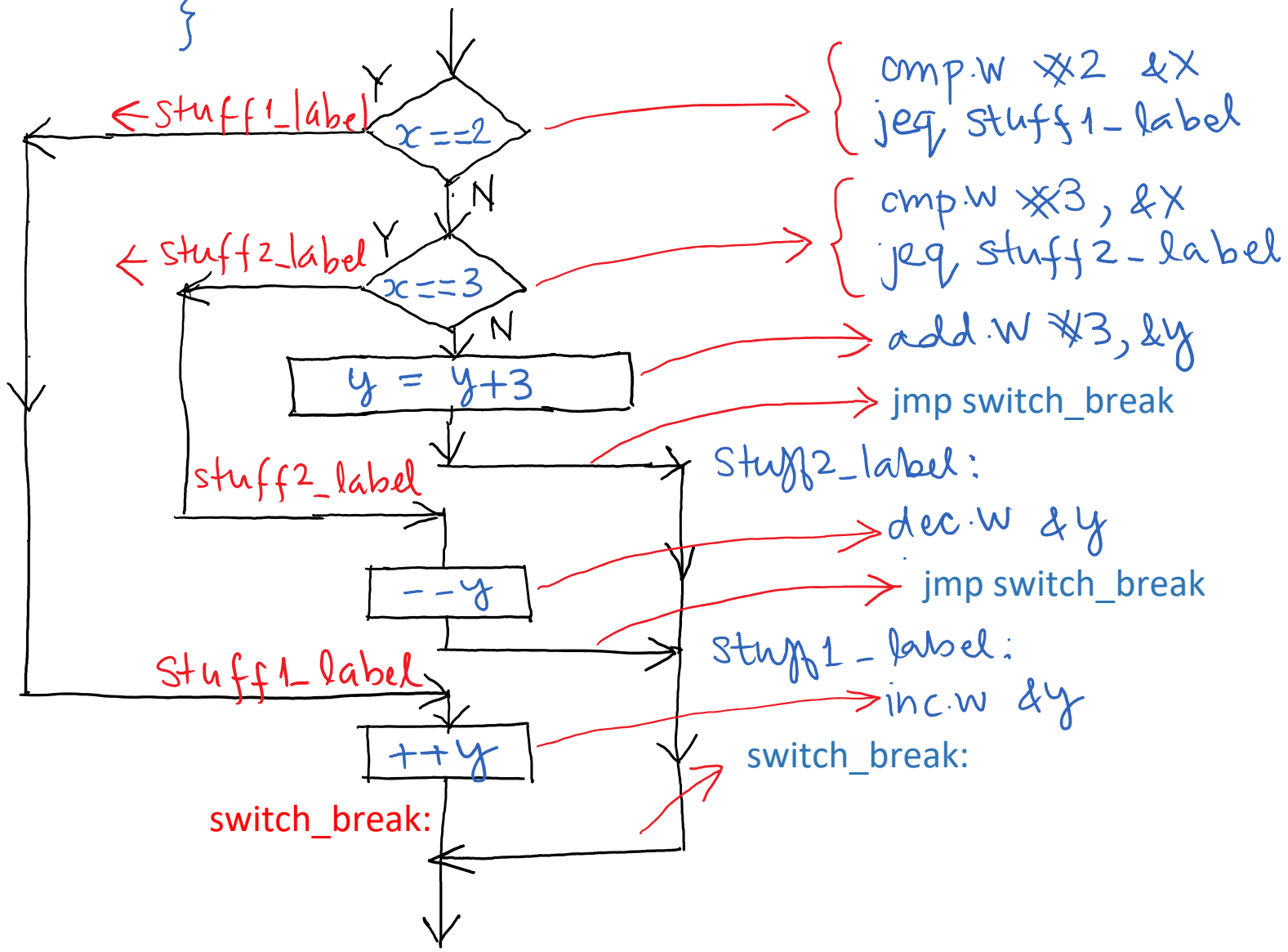
do stuff2 if x == 4

otherwise do default_stuff



example

```
switch (x)
{
  case 2:
     ++y
     break
  case 3:
     --y
     break
  default:
     y = y+3
     break:
}
```

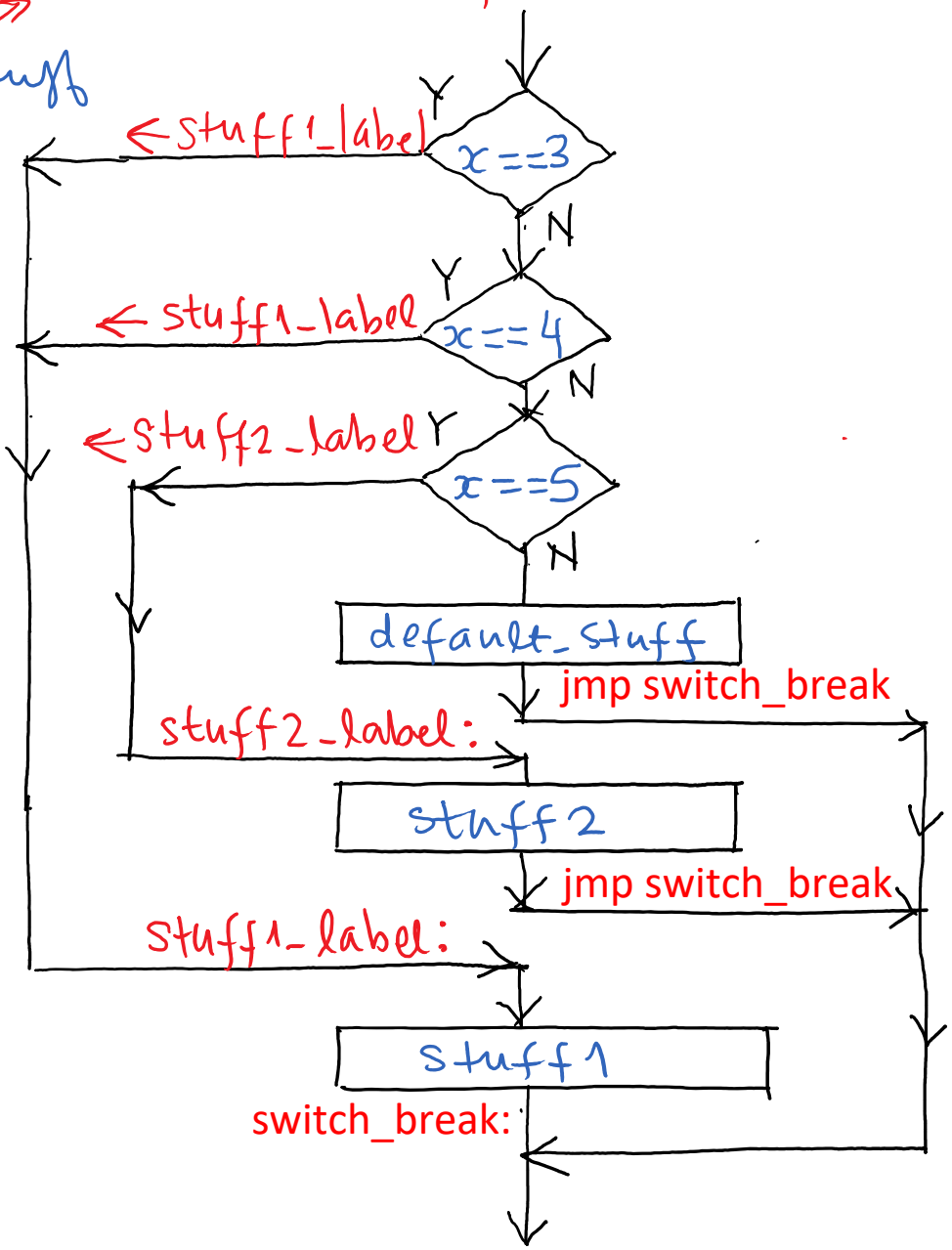


```
switch (x)
{
  case 3:
  case 4:
    stuff1
    break
  case 5:
    stuff2
    break
  default:
    default_stuff
    break:
}
```

do stuff1 if x == 3 || x == 4

do stuff2 if x == 5

otherwise do default_stuff



Example: switch_case inside for_loop

```
        .data
a:              .byte       3, 6, 3, 1, 3, 6, 0, 3, 0, 3
```

Count how many elements of a are equal to 3, how many are equal to 6 and how many are not equal to either 3 or 6.

Pseudo-code:

```
count1 = 0
count2 = 0
count3 = 0

for( i = 0;   i < 10;   ++i)
{
   switch( a[i] )
   {
       case 3:
            ++count1
            break
       case 6:
            ++count2
            break
       default:
            ++count3
            break
     }
}
```

Program: SwitchCase1

```
        .data
a:              .byte  3, 6, 3, 1, 3, 6, 0, 3, 0, 3
count1:         .space  2
count2:         .space  2
count3:         .space  2
;-------------------------------------------------------------------------
        .text
                        …
;-------------- for-loop (alternate implementation)--------------------------
                mov.w       #0, R5              ; i = 0
for_cond:
                cmp.w       #10, R5             ; (i >= 10) = ~(i<10)
                jge         for_break
;-------------- switch-case structure ---------------------------------------

                cmp.b#3, a(R5)
                jeq         stuff1_label

                cmp.b#6, a(R5)
                jeq         stuff2_label

                inc.w  &count3                  ; default stuff
                jmp         switch_break

stuff2_label:
                inc.w  &count2
                jmp         switch_break
stuff1_label:
                inc.w  &count1
switch_break:

;-------------- switch-case structure end -----------------------------------
                inc.w  R5
                jmp    for_cond
for_break:
;-------------- for-loop end------------------------------------------------
loop:           jmp    loop
```