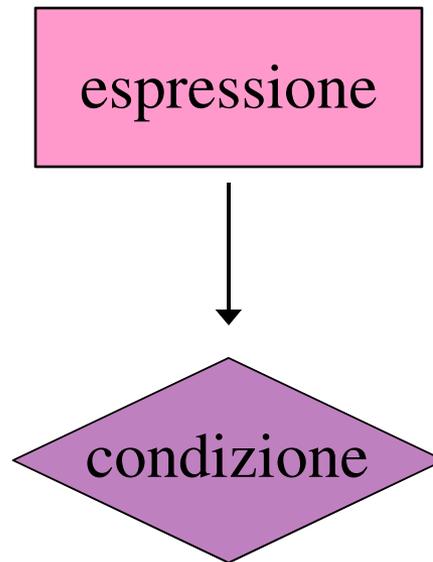
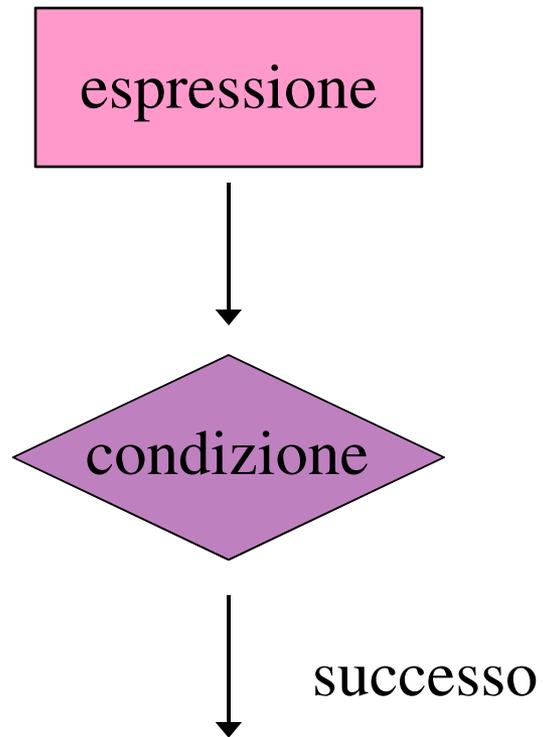


Istruzioni di selezione (if/else/switch)

espressione





Sintassi di if

if (**espressione**) **istruzione**

dove **espressione** è una qualsiasi espressione C e **istruzione** può essere una singola istruzione o una sequenza di istruzioni racchiusa tra { e }.

Semantica di if

- Nell'esecuzione di un'istruzione **if** viene valutata l'espressione tra parentesi
 - **Se** il suo valore è non nullo allora viene eseguita l'istruzione
 - **Se** il suo valore è nullo l'istruzione viene ignorata.
 - In entrambi i casi l'esecuzione procede con l'istruzione successiva all'**if**

Esempi di uso di if

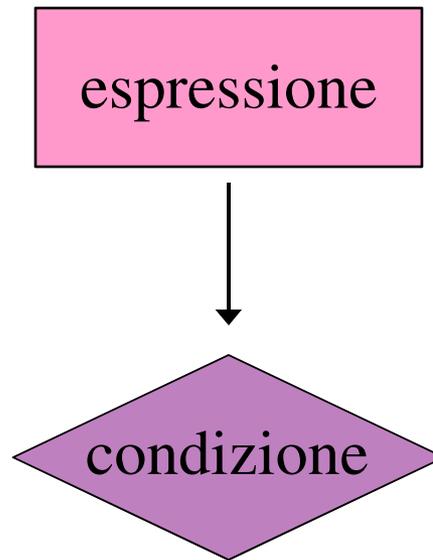
```
⋮  
⋮  
if ( x < y ) x = x + 5;
```

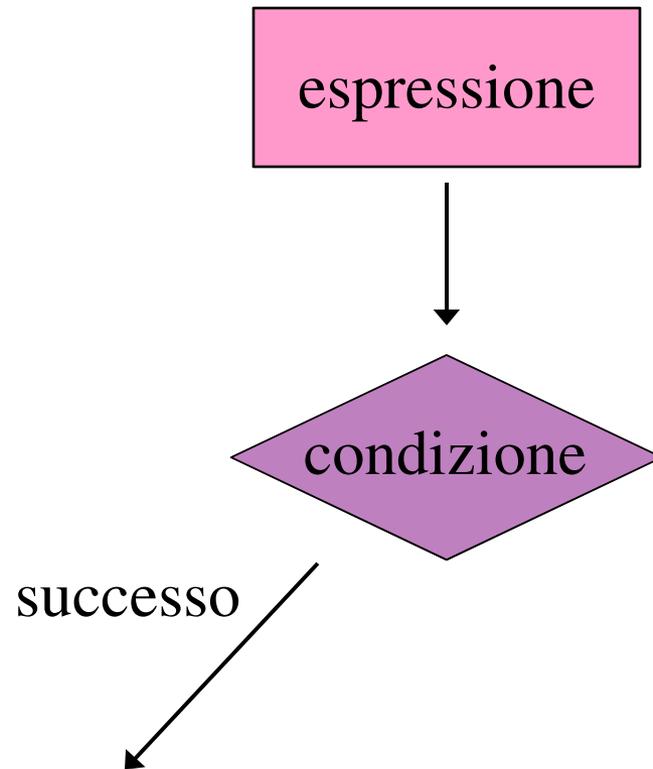
```
⋮  
⋮  
if ( x == 0 ) {  
    x = -1;  
    y = y + 25;
```

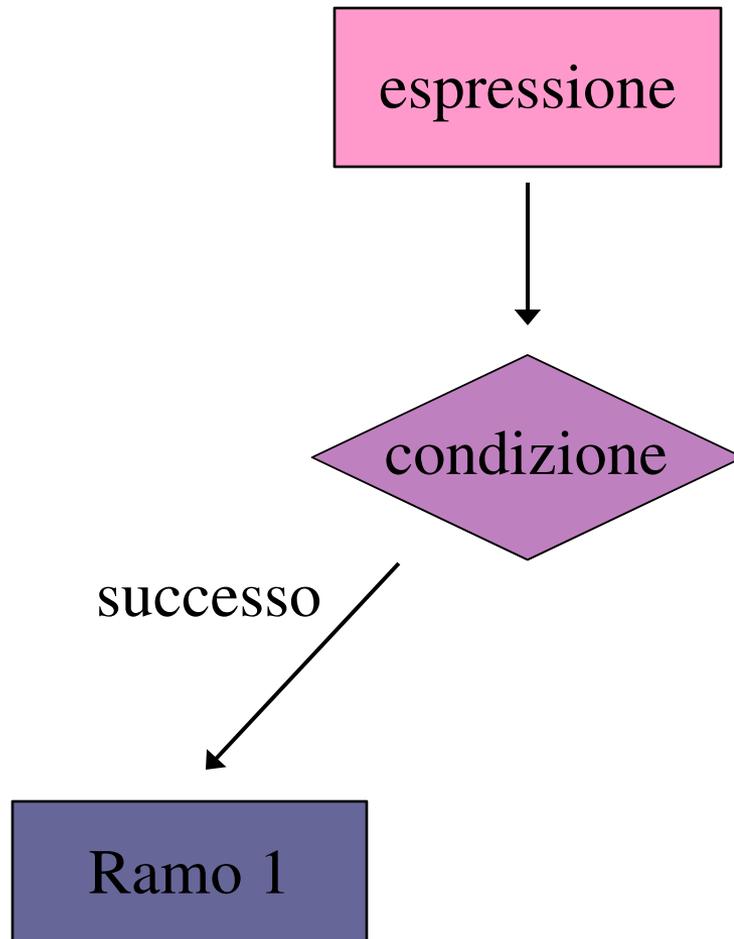
```
}  
⋮
```

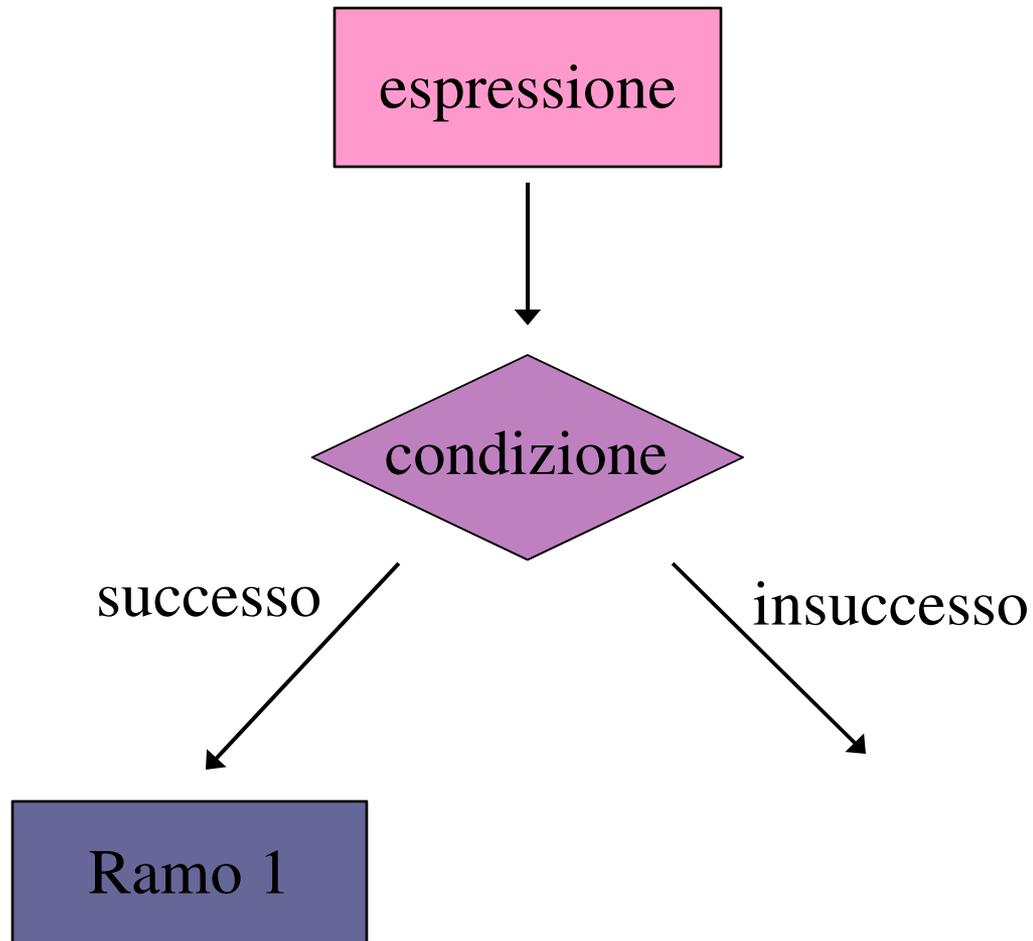
Esempio di programma

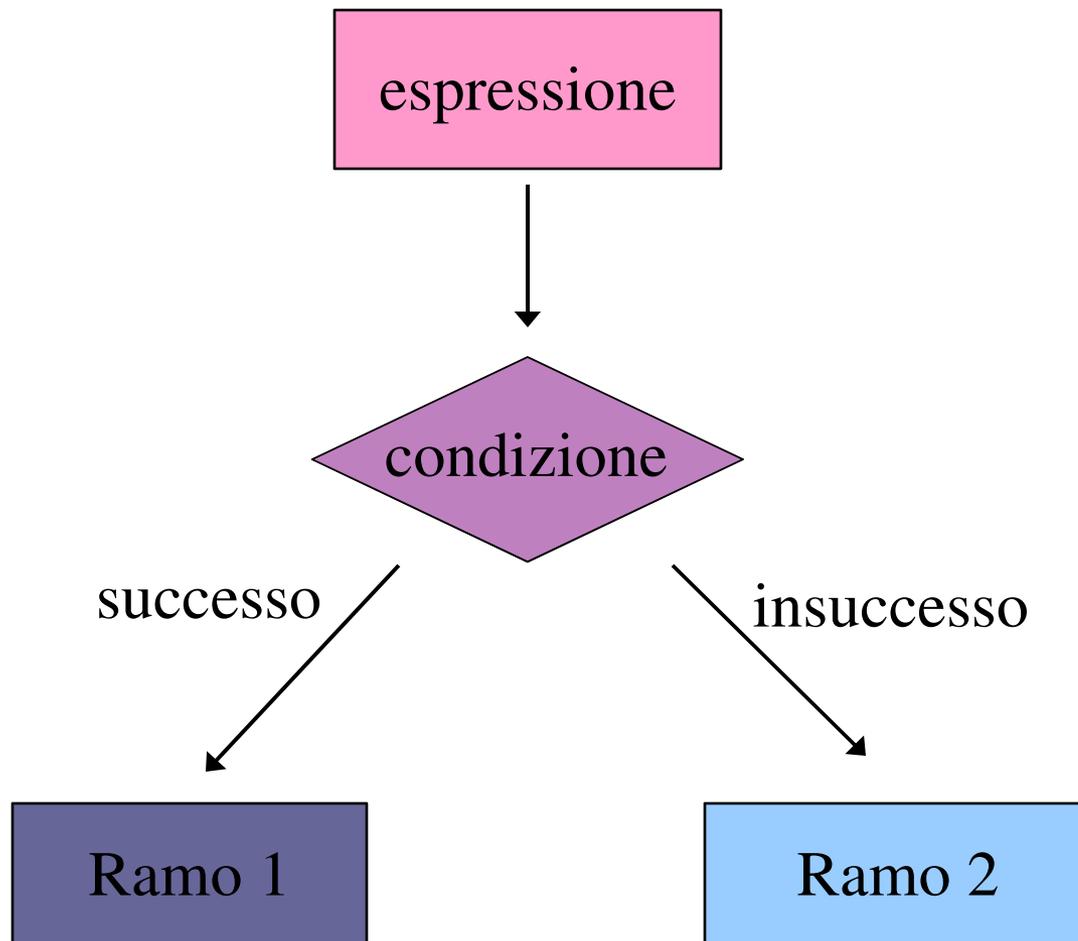
```
#include <stdio.h>
main ( )
{
    int  x, y;
    printf(" Inserire due numeri interi separati da invio\n");
    scanf("%d %d",&x,&y);
    printf(" valori inseriti:%d,%d",x,y);
    if(x > y) printf(" il primo e' piu' grande\n");
    if(x == y) printf(" sono uguali\n");
    if(x < y)printf(" il secondo e' piu' grande");
}
```











Sintassi di if...else

```
if (espressione) istruzione1 else istruzione2
```

dove **espressione** è una qualsiasi espressione C e **istruzione1(2)** sono una istruzione singola o sequenze di istruzioni racchiuse tra { e }.

Semantica di if...else

- Viene valutata l'**espressione** tra parentesi:
 - **Se** il valore è non nullo viene eseguita l'**istruzione1** e non viene eseguita l'**istruzione2**
 - **Se invece** il valore è nullo l'**istruzione1** non viene eseguita e viene eseguita l'**istruzione2**.

Esempi

```
:  
if ( x != 0 ) {  
    y = y / x;  
    x = x - 1;  
} else {  
    x = 25;  
    y = y / x;  
}  
:
```

```

:
if ( x != 0 ) {
    y = y / x;
    x = x - 1;
}

else {
    x = 25;
    y = y / x;
}

:
```

```
:  
:  
if ( x >y ) x = y - 6 ;  
else y = x + 6 ;  
:
```

Corretto ma di difficile lettura: quando si legge la riga dell'if non ci si aspetta che ci sia un seguito...

```
⋮  
⋮  
if ( x > 0 ) {  
    x = x + 25;  
    y = y / x;  
}  
else x = x + 1;  
⋮  
⋮
```

```
:  
:  
if ( x != 0 ) y = y / x;  
  
else {  
    x = 25;  
    y = y / x;  
}  
:
```

Esempio di programma

```
#include <stdio.h>
main( ){
    int  x, y, small, large;
    printf(" Inserire due numeri interi separati da invio\n");
    scanf("%d %d",&x,&y);

    if (x > y){
        large = x;
        small = y;
    } else {
        large = y;
        small = x;
    }
    printf(" %d >= %d \n",large,small);
}
```

if... else annidati

- Nei costrutti più complessi, in cui si vogliono porre delle condizioni all'interno di un `if...else`, si pone il problema di associare correttamente l'`else` ad un `if`.
- La regola è che ogni `else` è associato al più vicino `if` che lo preceda, sempre che questo non sia già stato associato ad un altro `else`.

Esempio:

Quanto vale y dopo l'esecuzione di questo blocco di istruzioni, dato $x = 4.5$?

```
y = 3;  
if (x > 4)  
if (x > 5)  
if (x > 6) y = 3;  
else y = 4;  
else y = 5;
```

Utilizziamo l'incolonnamento per visualizzare i diversi blocchi logici:

```
y = 3;  
if (x > 4)  
    if (x > 5)  
        if (x > 6) y = 3;  
        else y = 4;  
    else y = 5;
```

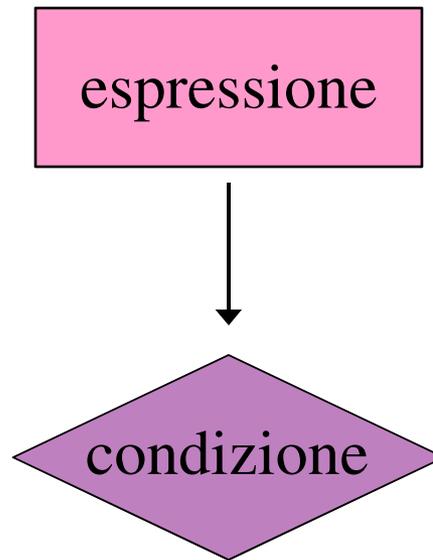
sono tutte istruzioni singole e quindi le parentesi graffe non sono necessarie, ma aiutano!

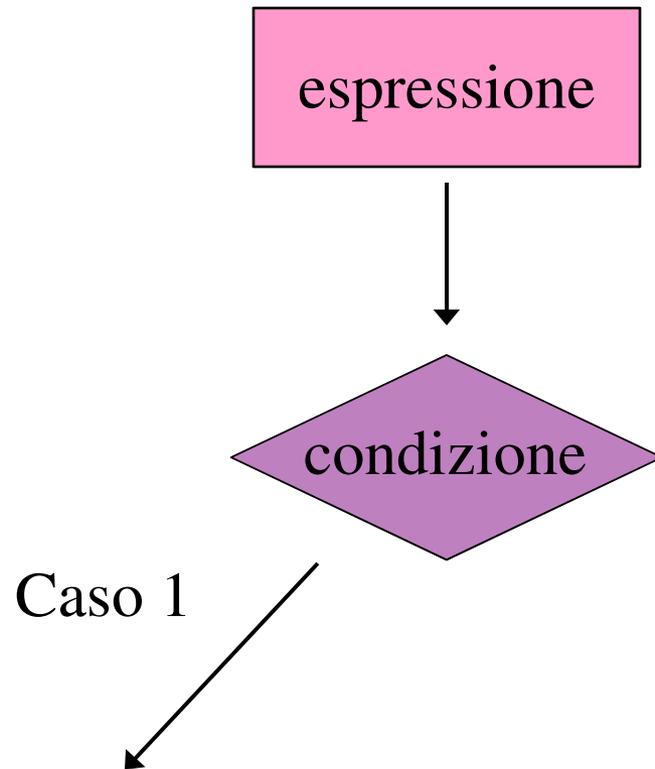
```

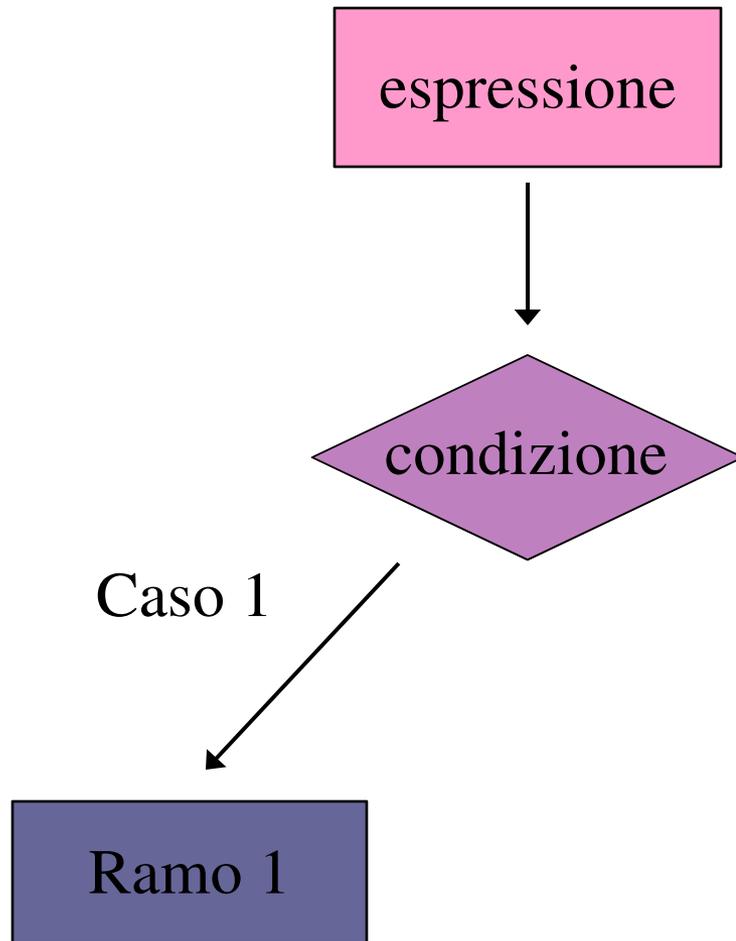
y=3;
if (x > 4) {
    if (x > 5) {
        if (x > 6) {
            y = 3;
        } else {
            y = 4;
        }
    } else {
        y = 5;
    }
}

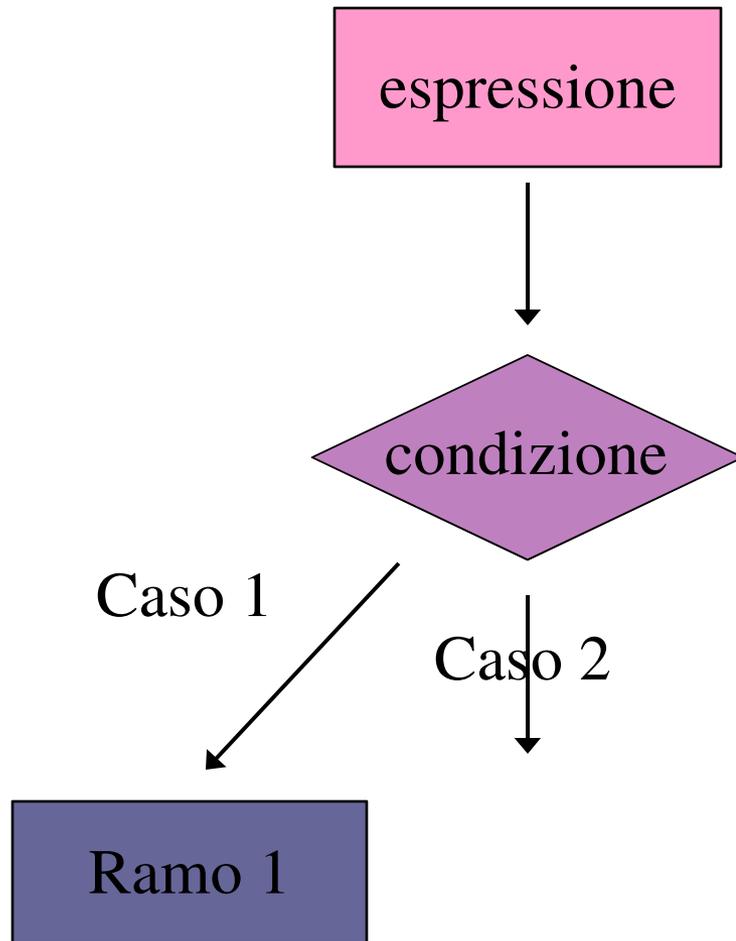
```

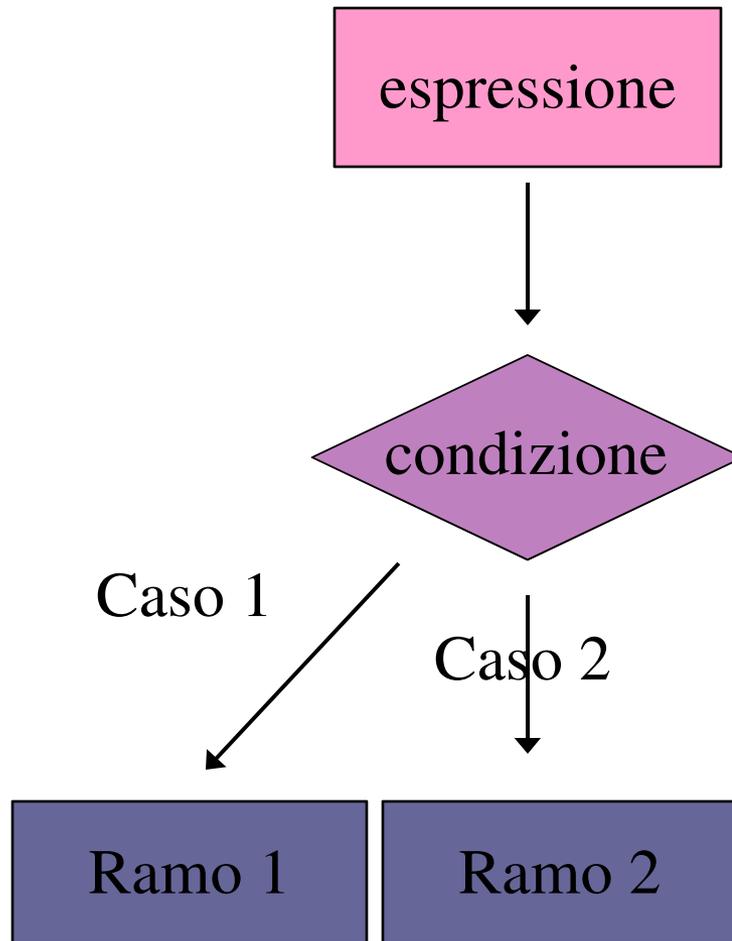
/* vero per x=4.5 */
 /* falso per x=4.5 */
 /* eseguito se x <= 5*/
 /* per x=4.5 y=5 ! */

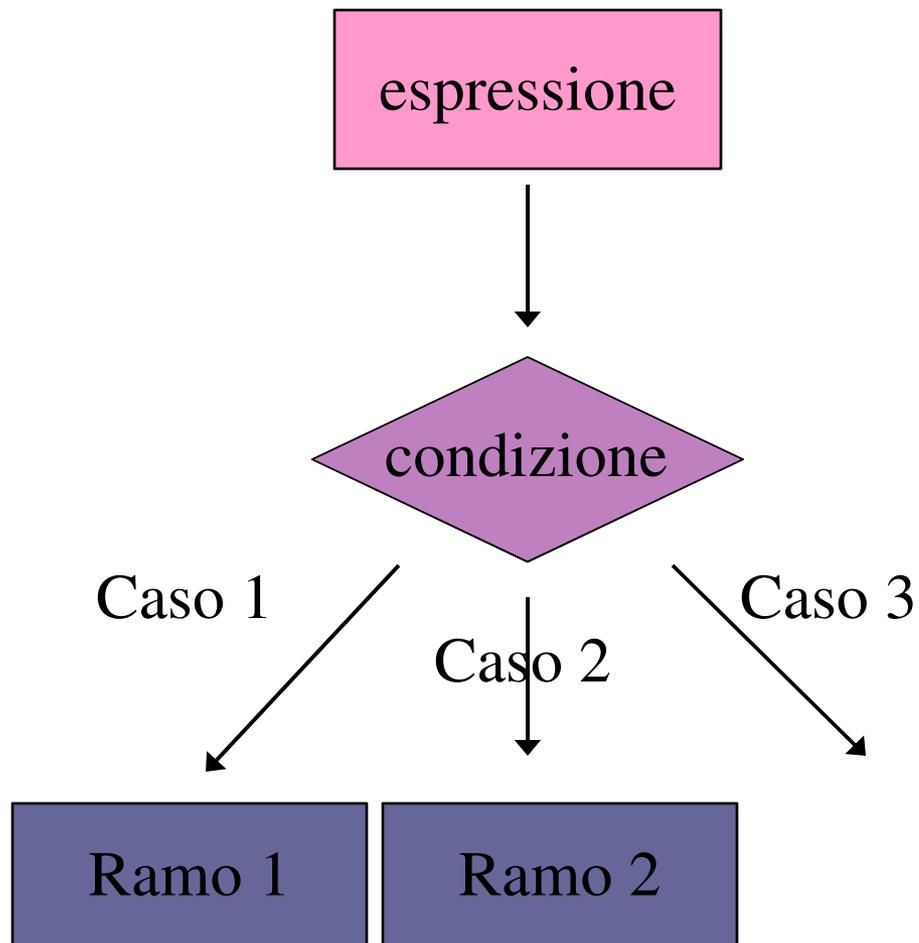


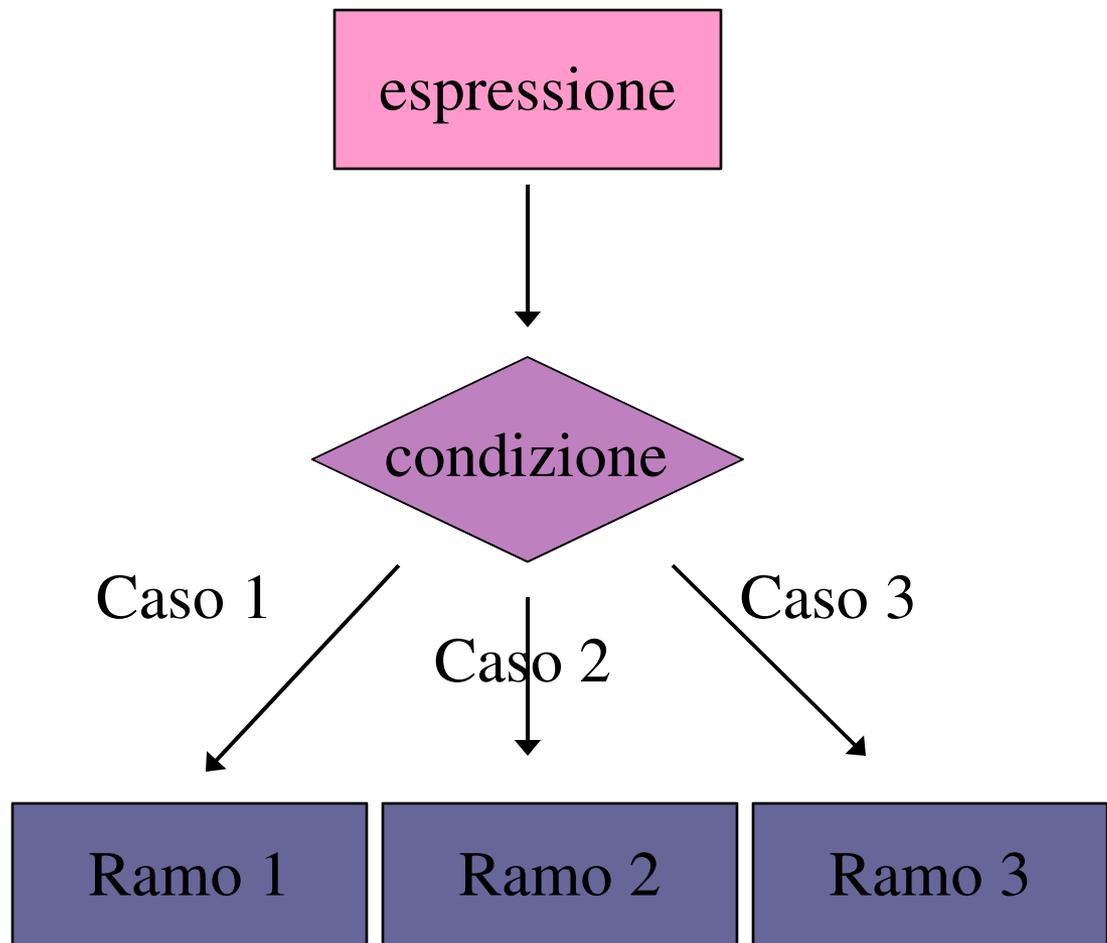












Sintassi di switch

```
switch(espressione) {  
    case val1: istruzione1  
    case val2: istruzione2  
    :  
    case valn: istruzionen  
default: istruzione  
}
```

Dove l'**espressione** assume **valori interi** (val1,... valn) e ogni **istruzione** può essere singola o composta da una sequenza di istruzioni.

Semantica di switch

Nell'esecuzione dell'istruzione **switch**

2. Viene valutata l'**espressione**
3. Il valore dell'espressione viene confrontato con i diversi valori dei **case**. Se è uguale ad uno di questi viene eseguito il blocco istruzione ad esso associato.
4. Se il valore dell'espressione non coincide con nessuno dei valori previsti si esegue il blocco di istruzioni associato a **default**.

NB: una volta trovato il caso giusto il programma **esegue anche** tutti i casi successivi!

Esempio

```
int dato;  
scanf("%d",&dato);  
switch(dato) {  
    case 5: printf("dato == 5\n");  
    case 4: printf("dato == 4\n");  
    default:  
        printf("dato diverso da 4 o 5 \n");  
}
```

Se `dato==6` viene eseguita solo la stampa di default, se è 4 vengono eseguite quelle del case 4 e del default, se è 5 vengono eseguite tutte!

break

Se si inserisce un **break** alla fine di ogni blocco istruzioni si ovvia al problema precedente:

```
int dato;
scanf("%d",&dato);
switch(dato) {
    case 5: printf("dato == 5\n");break;
    case 4: printf("dato == 4\n");break;
    default:
        printf("dato diverso da 4 o 5 \n");
}
```

else if

L'uso di switch con break corrisponde in definitiva a un uso di `if...else if...else`:

```
if(dato ==5) {
    printf("dato == 5\n");
} else if(dato==4) {
    printf("dato == 4\n");
} else {
    printf("dato diverso da 4 o 5 \n");
}
```