

La selezione binaria

Andrea Marin

Università Ca' Foscari Venezia
Laurea in Informatica
Corso di Programmazione

a.a. 2012/2013

Introduzione

- ▶ L'esecuzione di tutte le istruzioni in sequenza può non essere sufficiente per risolvere anche problemi molto semplici
- ▶ Supponiamo di voler scrivere un programma che acquisisca da standard input due numeri interi (dividendo e divisore) e scriva su standard output il loro quoziente e resto



Divisione (sol. errata)

```
#include <stdio.h>

int divisore , dividendo , quoziente , resto;

int main() {
    scanf("%d", &dividendo);
    scanf("%d", &divisore);
    quoziente = dividendo / divisore;
    resto = dividendo % divisore;
    printf("Il quoziente e %d mentre il
           resto e %d\n", quoziente , resto);
    return 0;
}
```

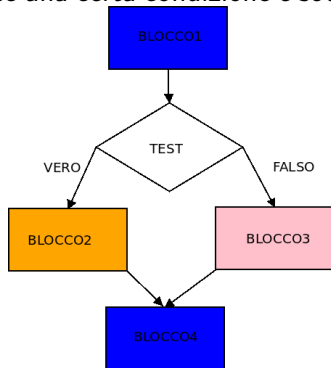


- ▶ Supponiamo che nell'input stream troviamo i numeri 19 e 6
- ▶ dividendo avrà valore 19 e divisore 6
- ▶ quoziente assume valore 3 dopo l'assegnamento
- ▶ resto assume valore 1 dopo l'assegnamento
- ▶ Ma il programma funziona sempre?
 - ▶ Quando la variabile divisore assume valore 0 si ha un errore *run-time* di divisione per zero
 - ▶ Per evitare questo dobbiamo svolgere le divisioni solo **se** il divisore ha valore diverso da 0



La struttura grafica

- La struttura del condizionale consente di eseguire un **blocco** di istruzioni se una certa condizione è soddisfatta, un altro **blocco** altrimenti



Possibili esecuzione:

- BLOCCO1 → BLOCCO2 → BLOCCO4 (se il test è soddisfatto)
- BLOCCO1 → BLOCCO3 → BLOCCO4 (se il test non è soddisfatto)

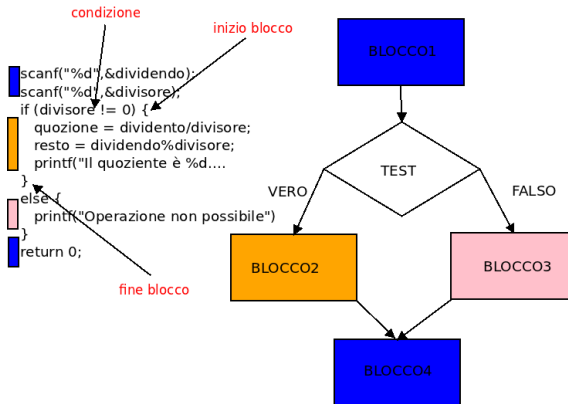


La condizione (o test)

- ▶ Idealmente la condizione dovrebbe essere un'*espressione* di tipo *boolean*
- ▶ In tal caso, un ramo viene scelto se l'espressione ha valore *true*, l'altro se l'espressione ha valore *false*
- ▶ In C l'espressione che dà la condizione è di tipo `int` con la consueta codifica dei valori booleani
 - ▶ Valore $= 0 \Rightarrow$ condizione non soddisfatta
 - ▶ Valore $\neq 0 \Rightarrow$ condizione soddisfatta



Ancora sull'esempio della divisione



Struttura sintattica del condizionale in C

```
if (exp)
    BLOCCO1;
else
    BLOCCO2;
```

dove:

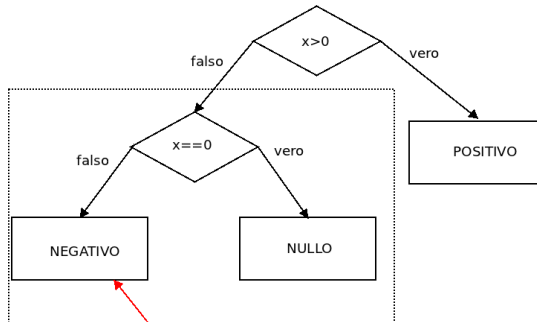
- ▶ exp è un'espressione di tipo intero
- ▶ BLOCCO1 è una o più istruzioni (nel qual caso racchiuse tra graffe) eseguite solo se exp ha un valore diverso da 0 (**true**)
- ▶ BLOCCO2 è una o più istruzioni (ne qual caso racchiuse tra graffe) eseguite solo se exp ha valore 0 (**false**)
 - ▶ Attenzione: la parte **else** del condizionale può essere assente

Nidificazione dei condizionali

- ▶ Supponiamo di voler scrivere un programma che acquisito un input intero stampi se esso è positivo, negativo o nullo
- ▶ Una sola selezione a due vie non basta!
- ▶ In questi casi si sfrutta il fatto che un **blocco** che compare nei rami del condizionale può a sua volta contenere altri condizionali



Esempio



Per concludere questo
tengo presente che $x > 0$ è falso (quindi $x \leq 0$) e che $x \neq 0$



Codice

```
#include <stdio.h>

int x;
int main() {
    scanf("%d", &x);
    if (x > 0) {
        printf("Positivo\n");
    }
    else {
        if (x == 0) {
            printf("Nullo\n");
        }
        else {
            printf("Negativo\n");
        }
    }
    return 0;
}
```

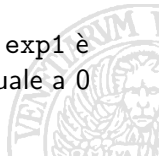


If aritmetico

- ▶ C mette a disposizione un costrutto chiamato if-aritmetico
- ▶ A differenza del condizionale appena visto, l'if-artimetrico è un'espressione
- ▶ Sintassi dell'espressione:

$$(exp1) \ ? \ exp2 \ : \ exp3 ;$$

- ▶ $exp1$ ha tipo `int`, $exp2$ e $exp3$ sono espressioni con lo stesso tipo
- ▶ il valore dell'espressione è quello di $exp2$ se il valore di $exp1$ è diverso da 0, è quello di $exp3$ se il valore di $exp1$ è uguale a 0



Calcolo del modulo

```
#include <stdio.h>
```

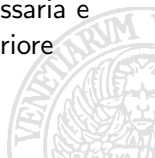
```
float x;
```

```
int main() {  
    scanf("%f", &x);  
    x = (x > 0) ? x : -x;  
    printf("%f\n", x);  
    return 0;  
}
```



Esercizi

1. Leggere due valori float da standard input e memorizzarli in variabili distinte. Quindi scambiare il valore delle due variabili.
2. Letti due valori int da standard input stamparli sullo standard output in ordine crescente
3. Letti tre valori float da standard input stampare su standard output il maggiore
4. Letti tre valori double da standard input dire se possono essere tre lunghezze dei lati di un triangolo. (Condizione necessaria e sufficiente è che la lunghezza del lato maggiore sia inferiore alla somma delle lunghezze dei due lati più piccoli).



Scambio di variabili

```
#include <stdio.h>
```

```
float x,y,supporto;
```

```
int main(){  
    scanf("%f",&x);  
    scanf("%f",&y);
```

```
    /*eseguo lo scambio*/  
    supporto = x;  
    x = y;  
    y = supporto;  
    ...  
}
```



Stampare il più grande di tre valori (alternative?)

```
#include <stdio.h>
float x, y, z;
int main() {
    /* leggi x,y,z*/
    ...
    if (x > y) {
        if (x > z) {
            printf("%f", x);
        }
        else {
            printf("%f", z);
        }
    }
    else {
        if (y > z) {
            printf("%f", y);
        }
        else {
            printf("%f", z);
        }
    }
    return 0;
}
```

