

ESEMPIO: le variabili

LETTURA/SCRITTURA DI VARIABILI

Specifica

- Leggere da tastiera un intero, un carattere, un float e un double.
- Stamparli a video.

LETTURA/SCRITTURA DI VARIABILI

Codice

```
#include <stdio.h>
void main() {
    int i;
    char c;
    float f;
    double d;
```

```
printf("Inserisci un intero: ");
scanf("\n%d",&i);
```

```
printf("Inserisci un carattere: ");
scanf("\n%c",&c);
```

Inserire `\n` prima della variabile da leggere equivale a svuotare il buffer di I/O. (vedi appunti su precisazioni scanf)

LETTURA/SCRITTURA DI VARIABILI

```
printf("Inserisci un numero in virgola mobile: ");  
scanf("\n%f",&f);
```

```
printf("Inserisci un numero in doppia precisione: ");  
scanf("\n%lf",&d);
```

```
printf("\n\nINTERO: %d\nCARATTERE: %c\nFLOAT:  
%f\nDOUBLE: %lf\n",i,c,f,d);
```

```
}
```

ESEMPIO: la selezione

CALCOLO DEL VALORE ASSOLUTO

Specifica

- Dato un valore letto da tastiera, calcolarne il valore assoluto.

Progetto

Definizione formale:

$$|x| = \begin{cases} x & \text{se } x \geq 0 \\ -x & \text{altrimenti} \end{cases}$$

- Leggi un valore da tastiera (quale tipo di dato?).
- Se il valore è maggiore o uguale a 0 stampalo, altrimenti stampa (-valore).

CALCOLO DEL VALORE ASSOLUTO

Codice soluzione 1

```
#include <stdio.h>
void main() {

    int x, temp;
    printf("Inserisci un intero: ");
    scanf("\n%d", &x);

    if ( x >= 0) {
        temp = x;
    } else {
        temp = -x;
    }

    printf("Il valore assoluto di %d e' %d\n\n",x, temp);
}
```

CALCOLO DEL VALORE ASSOLUTO

Codice soluzione 2

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

```
void main() {
```

```
    int x, temp;
```

```
    printf("Inserisci un intero: ");
```

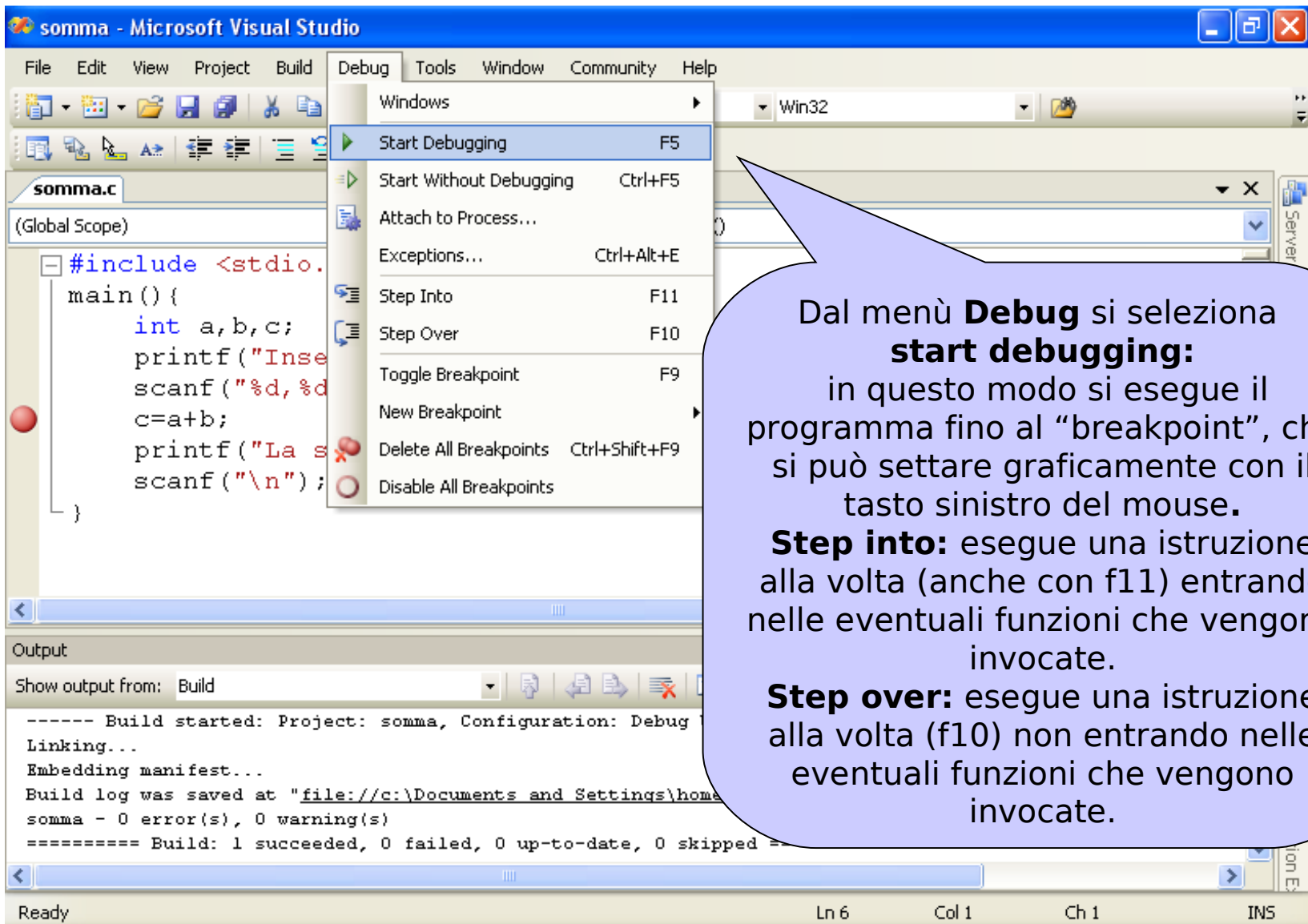
```
    scanf("\n%d", &x);
```

```
    temp = ( x >= 0 ) ? x : -x;
```

```
    printf("Il valore assoluto di %d e' %d\n\n", x, temp);
```

```
}
```

Debug



Dal menù **Debug** si seleziona **start debugging**: in questo modo si esegue il programma fino al "breakpoint", che si può settare graficamente con il tasto sinistro del mouse.

Step into: esegue una istruzione alla volta (anche con f11) entrando nelle eventuali funzioni che vengono invocate.

Step over: esegue una istruzione alla volta (f10) non entrando nelle eventuali funzioni che vengono invocate.

Watch

The screenshot shows the Microsoft Visual Studio IDE in Debug mode. The main window displays the source code for `somma.c` with the following content:

```
#include <stdio.h>
main() {
    int a,b,c;
    printf("Inserire i due addendi, separati da uno spazio");
    scanf("%d,%d", &a, &b);
    c=a+b;
    printf("La somma di %d e %d è uguale a %d", a, b, c);
    scanf("\n");
}
```

The **Autos** window at the bottom left shows the current state of local variables:

| Name | Value | Type |
|------|------------|-------|
| &a | 0x0012ff60 | int * |
| &b | 0x0012ff54 | int * |
| a | 2 | int |
| b | 34 | int |
| c | -858993460 | int |

The **Watch** window at the bottom right shows expressions being monitored:

| Name | Value | Type |
|------|------------|-------|
| c | -858993460 | int |
| c*3 | 1717986916 | int |
| &c | 0x0012ff48 | int * |
| | -858993460 | int |

The status bar at the bottom indicates the current position: `Ready`, `Ln 8`, `Col 17`, `Ch 14`, `INS`.

Nel riquadro **Autos** si può osservare il valore delle variabili del programma durante l'esecuzione.

Nel riquadro **Watch** si possono inserire delle espressioni ed osservare il loro valore durante l'esecuzione (si compie cioè l'operazione di "watching" di variabili ed espressioni).

Il simbolo & davanti ad una variabile indica il suo indirizzo in memoria.

Call Stack

The screenshot shows the Microsoft Visual Studio interface during a debug session. The main window displays the source code for `somma.c` with the `main()` function. The `Call Stack` window is open at the bottom, showing the current function and its callers. A speech bubble points to the `Call Stack` window.

Code in `somma.c`:

```
#include <stdio.h>
main() {
    int a, b, c;
    printf("Inserire i due addendi, separati da uno spazio\n");
    scanf("%d, %d", &a, &b);
    c=a+b;
    printf("La somma di %d e %d è uguale a %d\n", a, b, c);
    scanf("\n");
}
```

Call Stack:

| Name | Lang |
|--|------|
| somma.exe!main() Line 6 | C |
| somma.exe!__tmainCRTStartup() Line 586 + 0x19 | C |
| somma.exe!mainCRTStartup() Line 403 | C |
| kernel32.dll!7c816fd7() | |
| [Frames below may be incorrect and/or missing, no symbols loaded for kernel32.dll] | |

Autos:

| Name | Value | Type |
|------|------------|-------|
| &a | 0x0012ff60 | int * |
| &b | 0x0012ff54 | int * |
| a | 2 | int |
| b | 34 | int |
| c | -858993460 | int |

Speech Bubble:

Nel riquadro **Call Stack** viene visualizzato lo stack delle chiamate di funzione. Vediamo che il nostro programma (la cui funzione principale è il "main") viene lanciato a sua volta da altri "processi". La funzione in alto è quella ora in esecuzione.

ESERCIZIO 1

- Aprire l'ambiente integrato Visual C++.
- Creare un nuovo progetto di nome **media** e all'interno di questo inserire un file **voto.c**
- Progettare un algoritmo che dati tre numeri (voti) a , b , c che possono assumere i valori interi da 18 a 30, calcoli la loro media (il controllo sui dati in input è facoltativo)
- Codificare il programma in C e scriverlo in **voto.c**
- Procedere alle operazioni di Compile, Build e correggere eventuali errori. *I warning non sono errori.*
- Seguire l'esecuzione del programma con l'uso del debugger (mettere sotto osservazione la variabile *media*).

ESERCIZIO 2

- Creare un nuovo progetto di nome **media1** con un file **voto1.c**
- Progettare e codificare in C (scrivendolo in **voto1.c**) un programma che permetta il calcolo della media di tre voti dati in trentesimi. La media deve essere fornita in centesimi o trentesimi a seconda del valore assunto da una variabile *scala* di tipo carattere
- Seguire l'esecuzione con il debugger (mettere sotto osservazione la variabile *media*)
- Cosa succede se definite le variabili a cui sono assegnati i voti (*a,b,c*) di tipo *int*?
- E cosa invece se le definite di tipo *float*?
- In cosa differiscono e perché?

ESERCIZIO 3

Specifica

- Dati tre valori a , b , c , rappresentanti i coefficienti di un'equazione di secondo grado $ax^2 + bx + c = 0$, calcolarne le radici (reali).

Progetto

Occorre calcolare il valore $\text{delta} = b^2 - 4ac$.

Se $\text{delta} \geq 0$ allora ci sono radici reali (si può esprimere assegnando ad un boolean *Reali* il valore 1) altrimenti le radici sono complesse. (*Reali*=0)

Calcolo delle radici reali:

calcolare $d = \sqrt{\text{delta}}$

calcolare le due radici $x_1, x_2 = - (b \pm d) / 2a$

ESERCIZIO 3

Algoritmo (pseudo codice)

Delta = $b*b - 4*a*c$;

Reali = (Delta >= 0);

se Reali=1

{

Rad=radice(delta);

X1 = - (b+Rad)/2*a;

X2 = - (b- Rad)/2*a;

}

ESERCIZIO 3

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

```
main (){
    int reali=0;
    float a, b, c;
    float delta, d, x1, x2;
    printf("inserisci i valori dei coefficienti a,b,c\n");
    scanf("%f %f %f",&a,&b,&c);
    delta = b*b-4*a*c;
    reali=(delta>=0);
    if (reali){
        d = sqrt(delta);
        x1 = -(b+d)/(2*a);
        x2 = -(b-d)/(2*a);
        printf("le radici dell' equazione sono %f, %f\n",
            x1,x2);
    } else printf("l' equazione non ha radici reali\n");
}
```

Direttiva al preprocessore:
include la libreria matematica
(fornisce la funzione **sqrt**)

ESERCIZIO 4

Specifica

Progettare e codificare in C un programma che, data una quantità di soldi in lire, restituisca la quantità corrispondente in euro (/2000) o in dollari (/2500), a seconda del valore assunto da una variabile *moneta* di tipo char.

ESERCIZIO 5

Specifica

Progettare e codificare in C un programma che prende da tastiera, due variabili (tipo int da trattare come booleani [0,1]).

Stampa a video il valore logico delle due variabili (se x è uguale a 1 allora x è TRUE altrimenti è FALSE), e ne fa le seguenti operazioni:

- **AND**
- **OR**
- **XOR** $a \text{ XOR } b = (a \text{ OR } b) \text{ AND } (\text{NOT}(a \text{ AND } b))$
- **=>** $(a \text{ => } b) = \text{NOT}(a) \text{ OR } (a \text{ AND } b)$

Stampare a video i risultati, sia come intero ([0,1]), sia come booleano ([TRUE, FALSE]).

ESERCIZIO 6

Specifica

Progettare e codificare in C un programma che prende in ingresso il valore di un anno (es.: 1492, 2003...).

Stabilisce se l'anno è bisestile o no.

Un anno è bisestile se è divisibile per 4 e, qualora sia l'anno d'inizio di un secolo, solo se è divisibile per 400

- es: 1900 non bisestile, 2000 bisestile, 1984 bisestile

ESERCIZIO 7

Specifica

Progettare e codificare in C un programma che prende in ingresso tre numeri (reali); quindi stampa un piccolo menu del tipo:

1. Calcolo della media
2. Calcolo del max
3. Calcolo del min
4. Uscita

A seconda della scelta dell'utente, il programma esegue l'operazione richiesta, stampandone a video il risultato

ESERCIZIO 8

Calcolo del successivo di una data

Si scriva un programma C che

- prende in ingresso una data (giorno, mese, anno forniti come numeri interi)
- calcola la data successiva e la visualizza.
- Ricordiamo che
 - il mese di Febbraio ha 29 giorni negli anni bisestili
 - un anno è bisestile se è divisibile per 4 e, qualora sia l'anno d'inizio di un secolo, solo se è divisibile per 400
 - es: 1900 non bisestile, 2000 bisestile, 1984 bisestile