



Fondamenti di Informatica
Ingegneria Clinica
Lezione 23/10/2009



Prof. Raffaele Nicolussi
FUB - Fondazione Ugo Bordoni
Via B. Castiglione 59 - 00142 Roma

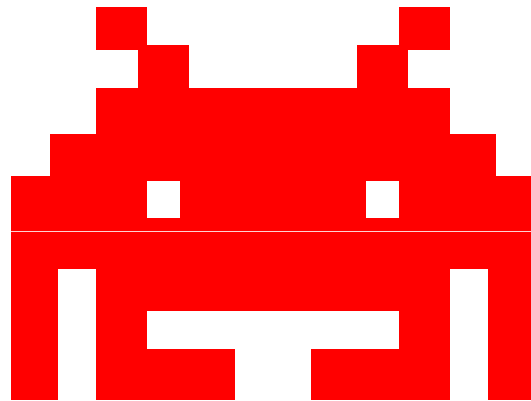


Docente	Raffaele Nicolussi	rnicolussi@fub.it 0654803323
Lezioni Aula 54 (ex aula 4) Via del Castro Laurenziano, 7	Lunedì, Giovedì, Venerdì	12:00 – 13:30
Esercitazioni Aula 15 Via Tiburtina, 205	Giovedì	14:00 – ???
Ricevimento:	Per appuntamento	in FUB, per email, per telefono
Sito web:	http://w3.uniroma1.it/IngClinFondinf	



Correzione esercizi del laboratorio

ERROR



Errori ricorrenti

- ❑ Salvataggio del programma come **.cpp** anziché **.c**



sbagliato.cpp

giusto.c



Errori ricorrenti

- Errata comprensione del testo o definizione dell'algoritmo
 - Se il testo mostra un esempio (come il rettangolo fatto di $=$) non vuol dire che il programma debba fare solo ed esattamente quello riportato nell'esempio!
 - Pensare prima a come risolvere il problema e solo dopo implementarne la soluzione
 1. individuare l'algoritmo
 2. scrivere il codice



scanf e printf

- **scanf** serve solo per chiedere all'utente un dato
- in **printf** devo mettere tanti **%tipodato** (**%d**, **%f**, **%s**, ...) quante sono le variabili che voglio stampare
 - devo poi, alla fine della printf, scrivere tanti nomi di variabile quanti sono i **%tipodato** indicati



varie

- Chiudere sempre l'ultima finestra in cui il programma è stato eseguito
 - Successive compilazioni ed esecuzioni non funzioneranno
 - Questo perché quando compilo viene creato un programma eseguibile (**nome.exe**)
 - Se il “vecchio” **nome.exe** è in esecuzione il “nuovo” **nome.exe** non può essere creato

Separa numero



1. Scrivere un programma che legga da standard input un intero in base 10 di quattro cifre e le visualizzi su standard output una ad una, separandole con un carattere di tabulazione

```
Inserire un numero intero positivo di 4 cifre: 1492
```

```
1         4         9         2
Premere un tasto per continuare . . .
```




Soluzione separa

```
/*
```

Scrivere un programma che legga da standard input un intero in base 10 di quattro cifre e le visualizzi su standard output una ad una, separandole con un carattere di tabulazione

```
*/
```

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

```
#include <stdlib.h>
```

```
int main(void) {
```

```
    int numero;
```

```
    int cifra1, cifra2, cifra3, cifra4;
```

```
    printf("Numero da separare : ");
```

```
    scanf("%d",&numero);
```



Soluzione separa (2)

```
if ( (numero>1000) && (numero<9999) )
{
    /*il C restituisce solamente la parte intera del risultato. Se esiste un resto, questo è perso */

    cifra1 = numero % 10;
    cifra2 = (numero / 10) % 10;
    cifra3 = (numero / 100) % 10;
    cifra4 = (numero / 1000) % 10;

    printf("Il numero %d diviso e\': %d %d %d %d\n", numero, cifra4, cifra3, cifra2, cifra1);

}

system("pause");
return 0;

}
```

Esercizi (2/5)



3. Scrivere un programma che legga da standard input una temperatura espressa in gradi Fahrenheit e visualizzi su standard output il valore corrispondente in gradi Celsius.

La formula per effettuare la conversione da Fahrenheit a Celsius è la seguente:

$$C = \frac{5}{9} \cdot (F - 32)$$

C=temperatura in gradi Celsius
F=temperatura in gradi Fahrenheit

```
Valore in gradi Fahrenheit = 451  
Valore in gradi Celsius = 232.777786  
Premere un tasto per continuare . . .
```

Esercizi (3/5)



4. Estendere il programma precedente in modo tale che l'utente possa scegliere se effettuare la conversione da Fahrenheit a Celsius o viceversa.

Il programma deve leggere da standard input un intero che identifica la scelta dell'utente: nel caso in cui il valore letto sia 1 si deve effettuare la conversione da Fahrenheit a Celsius, nel caso sia 2 si deve effettuare la conversione inversa.

La formula per effettuare la conversione da Celsius a Fahrenheit è la seguente:

$$F = \frac{9}{5} \cdot C + 32$$

C=temperatura in gradi Celsius
F=temperatura in gradi Fahrenheit

```
Premi
'1' per trasformare da F a C
'2' per trasformare da C a F
Scelta: 2

Valore in gradi Celsius = 100
Valore in gradi Fahrenheit = 212.000000
Premere un tasto per continuare . . .
```



Soluzione es. 2 e 3

```
int scelta;
float F,C;

do
{
    printf("1) F->C\n");
    printf("2) C->F\n");
    printf("3) Esci\n");

    scanf("%d",&scelta);

    if (scelta==1)
    {
        printf("F = ");
        scanf("%f",&F);

        C = (5.0/9.0) * (F - 32.0);

        printf("\n%f F = %f C\n",F,C);
    }
}
```



Soluzione es. 2 e 3

```
if (scelta==2)
{
    printf("C = ");
    scanf("%f",&C);

    F = ((9.0/5.0) * C) + 32.0;

    printf("\n%f C = %f F\n",C,F);

}

}
while (scelta!=3);
```

Esercizi (4/5)



5. Scrivere un programma che legga da standard input un carattere, che rappresenta un operatore ('*', '+', '-', '/'), e due numeri reali, e stampi su standard output il risultato dell'operazione corrispondente al carattere.

```
Inserire l'operazione da effettuare ('+', '-', '*', '/'): *  
Inserire due numeri reali: 2.2 3.2  
2.200000 * 3.200000 = 7.040000  
Premere un tasto per continuare . . .
```



Soluzione es. 4

```
float n1, n2;  
char operatore;  
  
printf("\nOperatore : ");  
scanf("%c",&operatore);  
  
printf("\nNumero 1 : ");  
scanf("%f",&n1);  
  
printf("\nNumero 2 : ");  
scanf("%f",&n2);  
  
printf ("\n%f %c %f = ",n1,operatore,n2);  
  
if (operatore=='+') printf ("%f\n",n1+n2);  
if (operatore=='-') printf ("%f\n",n1-n2);  
if (operatore=='*') printf ("%f\n",n1*n2);  
if (operatore=='/') printf ("%f\n",n1/n2);
```


Esercizi (5/5)



7. Scrivere un programma che, letti da standard input tre numeri interi, visualizzi su stdout, a seconda dei casi, uno dei seguenti messaggi:
- "tutti diversi"
 - "due uguali e uno diverso"
 - "tutti uguali"

```
Inserire tre numeri interi: 2 5 1
Tutti diversi
Premere un tasto per continuare . . .
```



Soluzione es. 5

```
int n1, n2, n3;
```

```
printf("\nNumero 1: ");
```

```
scanf("%d",&n1);
```

```
printf("\nNumero 2: ");
```

```
scanf("%d",&n2);
```

```
printf("\nNumero 3: ");
```

```
scanf("%d",&n3);
```

```
if (( n1 == n2 ) && (n2 == n3)) printf("Tutti uguali\n");
```

```
else
```

```
{
```

```
    if ((n1 == n2) || (n2 == n3) || (n1==n3)) printf("Due uguali\n");
```

```
    else printf("Tutti diversi\n");
```

```
}
```

Comportamento dei caratteri



```
#include <stdio.h>
int main (void)
{   int a, b, c, e;
    char d;
    a='Y';  b=89;  c=1;
        printf("\nIntero:%d\t\n", a);
        printf("Carattere: %c\n", a);
    e=a-b;
        printf("Differenza: %d\n", e);
    e=a-c;
        printf("Differenza2: %d\n", e);
        printf("Differenza2 in caratteri: %c\n", e);
    d=a-c;
        printf("Differenza3: %d\n", d);
        printf("Differenza3 in caratteri: %c\n", d);
    return 0; }
```

Intero: 89
Carattere: Y
Differenza: 0
Differenza2:88
Differenza2 in caratteri: X
Differenza3: 88
Differenza3 in caratteri:X



Tipi Floating-point (reali)

float, double, long double

- ❑ introdotti per rappresentare frazioni e numeri molto grandi
 - 0.356 0.000001
- ❑ per dichiarare tali valori si possono usare le parole chiave
 - **float**
 - **double**
 - **double** sta per **doppia precisione**: per rappresentare i numeri con una precisione circa doppia rispetto a float. In genere
 - **float**: 4 byte
 - **double**: 8 byte
 - **long double**: per un range e precisione maggiore di **double**
 - su molte macchine **long** e **double** sono sinonimi
- ❑ La rappresentazione dei floating-point è un aspetto poco standardizzato:
 - per individuare l'intervallo dei valori ammessi e la precisione si veda la documentazione del compilatore (anche nel file header <limits.h>)



Dichiarazioni

□ Variabili

- ogni variabile deve essere dichiarata (nome e tipo) prima di essere usata
 - `int i;`
 - `int j;`
 - `float a, b;`

□ Funzioni

- una funzione viene dichiarata per **il tipo di valore che restituisce**
- in mancanza di dichiarazione di tipo per una funzione, viene associato un tipo di dati di default (`int`)
- se la funzione non restituisce valori, si usa `void`

□ `void`

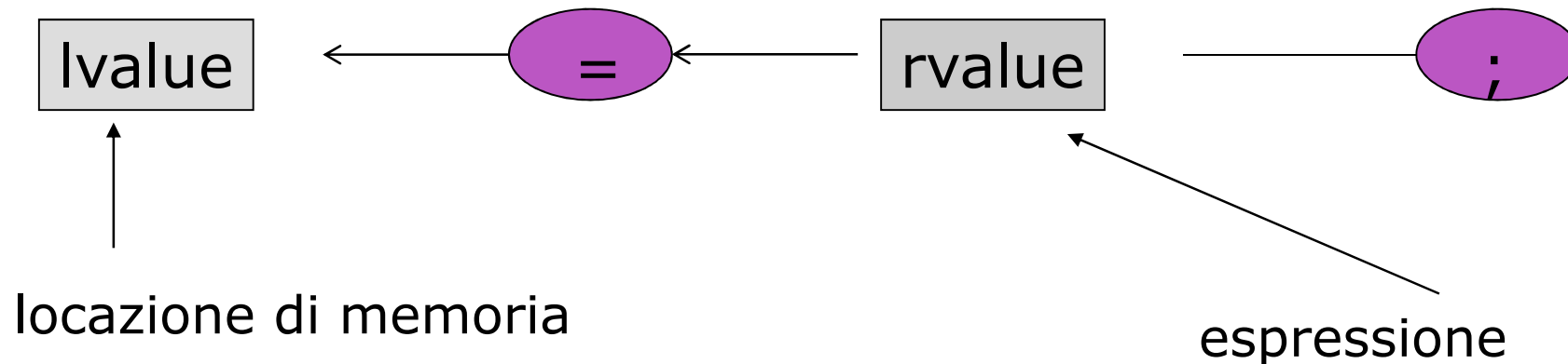
- permette di dichiarare esplicitamente una funzione che non restituisce alcun valore
 - `void funzione (a,b) {...}`
- indica l'assenza di argomenti
 - `int main (void) {... }`



Espressioni

- Un'espressione è una combinazione di **operatori**, **numeri** e **identificatori** che permette di calcolare un valore.
- Elementi fondamentali: variabili, costanti e chiamate di funzioni
 - 5 costante
 - j variabile
 - f(..) chiamata di funzione
 - gli elementi possono essere combinati tramite
 - operatori aritmetici
 - operatori relazionali e di uguaglianza
 - operatori logici

Istruzioni di assegnamento `lvalue=rvalue;`





Operatori aritmetici

addizione	+	a+b
sottrazione	-	a-b
moltiplicazione	*	a*b
divisione	/	a/b parte intera
modulo	%	a%b resto

Operatore	Priorità
()	<ul style="list-style-type: none">- valutate per prime- se nidificate, prima coppia più interna- se allo stesso livello, da sinistra a destra
*, /, %	<ul style="list-style-type: none">- valutate per seconde- se allo stesso livello, da sinistra a destra
-, +	<ul style="list-style-type: none">- valutate per ultime- se allo stesso livello da sinistra a destra

Divisione intera e reale



```
#include <stdio.h>

int main (void)
{
    int primo, secondo, ris, q, r;
    float d, a=33.567, b=2.898;
    primo=34;
    secondo=21;

    q=primo/secondo;
    r=primo%secondo;
    printf("\n\nDivisione: %d\t Resto:%d\n", q, r);

    d= a/b;
    printf("Divisione tra a e b: %f\n", d);

    ris=(primo/secondo)*secondo + primo%secondo;
    printf("Valore di ris: %d\n", ris);

    return 0;
}
```

Divisione: 1 Resto:13
Divisione tra a e b: 11.582816
Valore di ris: 34 (= primo)

Dati due interi positivi, calcola quoziente e resto della div. intera



```
#include <stdio.h>
int main(void)
{
    int a, b, quoziente, resto;

    printf("Inserisci dividendo: ");
    scanf("%d", &a);
    printf("Inserisci divisore: ");
    scanf("%d", &b);

    quoziente = a / b;
    resto = a % b;

    printf("%d : %d = %d con resto di %d\n",
           a, b, quoziente, resto);

    return 0;
}
```

Diversi comportamenti per la divisione con 0



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

```
int main (void)
```

```
{
```

```
    int numcor;
```

```
    double nreale;
```

```
    numcor=2;
```

```
    nreale = numcor / 0.0;
```

```
    printf("\n\nLa divisione: %f\n\n", nreale);
```

```
    return 1;
```

```
}
```

Risposta on TurboC

/* La divisione: +INF */

Risposta in VisualC

/* error C21: divide or mod by zero