

Esercitazione 02

2025/09/28 – Strutture di Controllo 1 (IF, ELSE)

L. Alessandrini

Credits per esercizi: A. Montenegro

Esercizio 01

Conversione da Fahrenheit a Celsius

Testo

Si scriva un programma che prenda in input un numero che rappresenti una temperatura in gradi Fahrenheit e che stampi a schermo la corrispondente temperatura in gradi Celsius.

Si usi la seguente formula di conversione:

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

Esempio

$$137^{\circ}\text{F} = 58.33^{\circ}\text{C}$$

Come Funziona il buffer?

Input da parte dell'utente

Dobbiamo immaginare il buffer di input come una “sequenza di caselle” all'interno del quale io vado ad inserire i vari caratteri.

Cosa potrebbe succedere con più scansioni successive mediante `scanf`?

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Inserisco un carattere, lo salvo nella variabile `primo_carattere`, e catturo il ritorno della `scanf` in `ritorno_scanf_primo_carattere`

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16
17    RETURN VALUE
18    On success, these functions return the number of input items successfully matched and assigned; this can be fewer than
19    provided for, or even zero, in the event of an early matching failure.
20
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Inserisco un carattere, lo salvo nella variabile `primo_carattere`, e catturo il ritorno della `scanf` in

Dal manuale della `scanf`

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Acquisisco un intero, e catturo l'output della `scanf`

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Acquisisco un altro carattere, e catturo l'output della scanf

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Catturo un eventuale invio rimasto nel buffer con una `scanf`

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Nota! Se non ci sono caratteri rimanenti nel buffer, questa linea si aspetterà un carattere in input aggiuntivo

Catturo un eventuale invio rimasto nel buffer con una `scanf`

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer



Terminale:

Inserisci un carattere:

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer

t	\n						
---	----	--	--	--	--	--	--

Terminale:

Inserisci un carattere: t

Inserisco t e premo invio

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer



Terminale:

```
Inserisci un carattere: t
```

La scanf consuma il primo carattere (t) e lo salva in primo_carattere
Avendo letto un carattere, ritorno_scanf_primo_carattere vale 1

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12    // acquisizione di un numero intero
13    printf("Inserisci un numero intero: ");
14    ritorno_scanf_intero = scanf("%d", &numero_intero);
15
16    // acquisizione del secondo carattere
17    printf("Inserisci un altro carattere: ");
18    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
19
20    printf("Ora consumo eventuali caratteri rimanenti");
21    // scanf dell'invio restato nel buffer
22    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
23
24    printf("\n\nprimo carattere: %c\n", primo_carattere);
25    printf("numero intero: %d\n", numero_intero);
26    printf("secondo carattere: %c", secondo_carattere);
27    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
28    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
29    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
30    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
31    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
32    printf("Fine del programma\n");
33
34    return 0;
35 }
36
37
```

Stato buffer

\n							
----	--	--	--	--	--	--	--

Terminale:

```
Inserisci un carattere: t
```

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer

\n	5	\n					
----	---	----	--	--	--	--	--

Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
```

Alla richiesta del numero inseriamo 5 e premiamo invio

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer



Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
```

La scanf dell'intero ignora il primo \n e consuma l'intero

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer

\n							
----	--	--	--	--	--	--	--

Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
```

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer

\n							
----	--	--	--	--	--	--	--

Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere:
```

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer

\n							
----	--	--	--	--	--	--	--

Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere:
```

A questo punto il programma **NON** attenderà l'input dell'utente ...

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer



Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimanenti
```

A questo punto il programma **NON** attenderà l'input dell'utente ...

Perché nel buffer c'è un carattere, che verrà consumato dalla scanf a linea 20!

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer



Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimanenti
```

Siccome non ci sono caratteri rimanenti, l'ultima scanf inserita attende un input da parte dell'utente

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer

r	\n						
---	----	--	--	--	--	--	--

Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimanenti
```

Inseriamo r e
premiamo invio

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer



Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimanenti
```

Che verrà consumato dalla scanf

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer

\n							
----	--	--	--	--	--	--	--

Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimanenti
```

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Stato buffer

\n							
----	--	--	--	--	--	--	--

Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimanenti

primo carattere: t
numero intero: 5
secondo carattere:
ultimo carattere (invio): r
Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma
```

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main() {
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimasti nel buffer");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Nota come l'output vada a capo nonostante non ci sia un invio nella printf!

Stato buffer

\n							
----	--	--	--	--	--	--	--

Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimasti nel buffer

primo carattere: t
numero intero: 5
secondo carattere:
ultimo carattere (invio): r
Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma
```

Come Funziona il buffer?

Cosa ci aspettiamo in output dato l'input?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimasti nel buffer");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
```

Nota come l'output vada a capo nonostante non ci sia un invio nella printf!

Quella scanf ha consumato l'invio dopo l'inserimento del 5

Stato buffer

\n							
----	--	--	--	--	--	--	--

Terminale:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimasti nel buffer

primo carattere: t
numero intero: 5
secondo carattere:
ultimo carattere (invio): r
Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma
```

Come Funziona il buffer?

Abbiamo capito che è importante pulire il buffer

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11    fflush(stdin); // pulizia del buffer di input
12
13
14
15    // acquisizione di un numero intero
16    printf("Inserisci un numero intero: ");
17    ritorno_scanf_intero = scanf("%d", &numero_intero);
18    fflush(stdin); // pulizia del buffer di input
19
20    // acquisizione del secondo carattere
21    printf("Inserisci un altro carattere: ");
22    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
23    fflush(stdin); // pulizia del buffer di input
24
25    printf("Ora consumo eventuali caratteri rimanenti");
26    // scanf dell'invio restato nel buffer
27    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
28
29    printf("\n\nprimo carattere: %c\n", primo_carattere);
30    printf("numero intero: %d\n", numero_intero);
31    printf("secondo carattere: %c", secondo_carattere);
32    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
33    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
34    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
35    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
36    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
37    printf("Fine del programma\n");
38
39    return 0;
40 }
41
```

Come Funziona il buffer?

Abbiamo capito che è importante pulire il buffer

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11    fflush(stdin); // pulizia del buffer di input
12
13
14
15    // acquisizione di un numero intero
16    printf("Inserisci un numero intero: ");
17    ritorno_scanf_intero = scanf("%d", &numero_intero);
18    fflush(stdin); // pulizia del buffer di input
19
20    // acquisizione del secondo carattere
21    printf("Inserisci un altro carattere: ");
22    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
23    fflush(stdin); // pulizia del buffer di input
24
25    printf("Ora consumo eventuali caratteri rimanenti");
26    // scanf dell'invio restato nel buffer
27    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
28
29    printf("\n\nprimo carattere: %c\n", primo_carattere);
30    printf("numero intero: %d\n", numero_intero);
31    printf("secondo carattere: %c", secondo_carattere);
32    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
33    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
34    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
35    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
36    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
37    printf("Fine del programma\n");
38
39    return 0;
40 }
41
```

Abbiamo aggiunto gli fflush dopo le scanf intermedie ...
Vediamo se cambia qualcosa:

Come Funziona il buffer?

Abbiamo capito che è importante pulire il buffer

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11    fflush(stdin); // pulizia del buffer di input
12
13
14
15    // acquisizione di un numero intero
16    printf("Inserisci un numero intero: ");
17    ritorno_scanf_intero = scanf("%d", &numero_intero);
18    fflush(stdin); // pulizia del buffer di input
19
20    // acquisizione del secondo carattere
21    printf("Inserisci un altro carattere: ");
22    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
23    fflush(stdin); // pulizia del buffer di input
24
25    printf("Ora consumo eventuali caratteri rimanenti");
26    // scanf dell'invio restato nel buffer
27    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
28
29    printf("\n\nprimo carattere: %c\n", primo_carattere);
30    printf("numero intero: %d\n", numero_intero);
31    printf("secondo carattere: %c", secondo_carattere);
32    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
33    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
34    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
35    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
36    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
37    printf("Fine del programma\n");
38
39    return 0;
40 }
41
```

Abbiamo aggiunto gli fflush dopo le scanf intermedie ...
Vediamo se cambia qualcosa:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimanenti

primo carattere: t
numero intero: 5
secondo carattere:
ultimo carattere (invio): r
Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma
```

senza fflush(stdin)
(precedente output)

Come Funziona il buffer?

Abbiamo capito che è importante pulire il buffer

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11    fflush(stdin); // pulizia del buffer di input
12
13
14
15    // acquisizione di un numero intero
16    printf("Inserisci un numero intero: ");
17    ritorno_scanf_intero = scanf("%d", &numero_intero);
18    fflush(stdin); // pulizia del buffer di input
19    printf("Inserisci un altro carattere: Ora consumo eventuali caratteri rimanenti");
20
21    // acquisizione del secondo carattere
22    printf("Inserisci un altro carattere: ");
23    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
24    fflush(stdin); // pulizia del buffer di input
25    printf("Inserisci un altro numero intero: ");
26    ritorno_scanf_intero = scanf("%d", &numero_intero);
27    fflush(stdin); // pulizia del buffer di input
28    printf("Inserisci un altro carattere: ");
29    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d", ritorno_scanf_primo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma");
35
36    return 0;
37 }
```

Con fflush(stdin)
(nuovo programma)

Abbiamo aggiunto gli fflush
dopo le scanf intermedie ...
Vediamo se cambia qualcosa:

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimanenti

primo carattere: t
numero intero: 5
secondo carattere:
ultimo carattere (invio): r
Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma
```

senza fflush(stdin)
(precedente output)

Come Funziona il buffer?

Abbiamo capito che è importante pulire il buffer

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11    fflush(stdin); // pulizia del buffer di input
12
13
14
15    // acquisizione di un numero intero
16    printf("Inserisci un numero intero: ");
17    ritorno_scanf_intero = scanf("%d", &numero_intero);
18    fflush(stdin); // pulizia del buffer di input
19
20    // acquisizione di un altro carattere
21    printf("Inserisci un altro carattere: ");
22    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
23    fflush(stdin); // pulizia del buffer di input
24
25    // acquisizione di un secondo carattere
26    printf("Inserisci un secondo carattere: ");
27    ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
28    fflush(stdin); // pulizia del buffer di input
29
30    // acquisizione di un ultimo carattere (invio)
31    printf("Inserisci un ultimo carattere (invio): ");
32    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
33    fflush(stdin); // pulizia del buffer di input
34
35    // stampa dei risultati
36    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
37    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
38    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
39    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
40    printf("Fine del programma\n");
41    return 0;
42 }
```

Abbiamo aggiunto gli fflush
dopo le scanf intermedie ...
Vediamo se cambia qualcosa:

NON HA FUNZIONATO

**Con fflush(stdin)
(nuovo programma)**

**senza fflush(stdin)
(precedente output)**

fflush

Cosa succede?

Leggiamo il manuale di `fflush` da Linux.

Ad esempio, dopo aver installato il seguente pacchetto su una qualsiasi distro Arch-based: [man-pages](#)^{1,2}, digitando da terminale:

```
username@hostname$ man fflush
```

Possiamo leggere il manuale di `fflush`

1. [Arch Wiki](#)

2. [kernel.org](#)

fflush

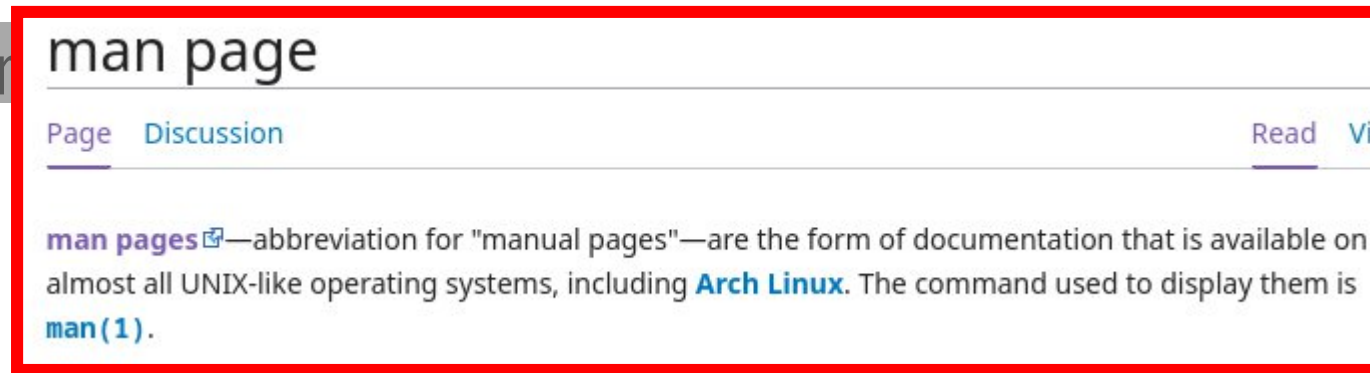
Cosa succede?

Leggiamo il manuale di `fflush` da Linux.

Ad esempio, dopo aver installato il seguente pacchetto su una qualsiasi distro Arch-based: [man-pages](#)^{1,2}, digitando da terminale:

username@hostname

Possiamo leggere il



1. [Arch Wiki](#)

2. [kernel.org](#)

fflush

Cosa succede?

Leggiamo il manuale di fflush da Linux.

Ad qu us

Linux *man-pages*: [home](#) || [git](#) | [readme](#) | [contributing](#) | [conventions](#) | [mailing list](#) | [archives](#) | [releases](#) | [book](#) | [online pages](#) [Search online pages](#)

The Linux *man-pages* project

The Linux *man-pages* project documents the [Linux kernel](#) and C library interfaces that are employed by user-space programs. With respect to the C library, the primary focus is the [GNU C library \(glibc\)](#), although, where known, documentation of variations in other C libraries available for Linux is also included.

[Project maintenance](#) | [Missing pages](#)

Page	Discussion	Read	Write
man pages  —abbreviation for "manual pages"—are the form of documentation that is available on almost all UNIX-like operating systems, including Arch Linux . The command used to display them is man(1) .			

Possiamo leggere il

1. [Arch Wiki](#)
2. [kernel.org](#)

fflush

Cosa succede?

[35.3Kb/s]/[163.9Kb/s] 100% 18.67 68 47°C 100% Tue 23 Sep 16:12

fflush(3)

Library Functions Manual

fflush(3)

NAME

fflush - flush a stream

LIBRARY

Standard C library ([libc](#), [-lc](#))

SYNOPSIS

#include <stdio.h>

int fflush(FILE *_Nullable [stream](#));

DESCRIPTION

For output streams, [fflush\(\)](#) forces a write of all user-space buffered data for the given output or update [stream](#) via the stream's underlying write function.

For input streams associated with seekable files (e.g., disk files, but not pipes or terminals), [fflush\(\)](#) discards any buffered data that has been fetched from the underlying file, but has not been consumed by the application.

The open status of the stream is unaffected.

If the [stream](#) argument is NULL, [fflush\(\)](#) flushes [all](#) open output streams.

For a nonlocking counterpart, see [unlocked_stdio\(3\)](#).

RETURN VALUE

Upon successful completion 0 is returned. Otherwise, EOF is returned and [errno](#) is set to indicate the error.

ERRORS

EBADF [stream](#) is not an open stream, or is not open for writing.

The function [fflush\(\)](#) may also fail and set [errno](#) for any of the errors specified for [write\(2\)](#).

ATTRIBUTES

For an explanation of the terms used in this section, see [attributes\(7\)](#).

Interface	Attribute	Value
fflush()	Thread safety	MT-Safe

STANDARDS

C11, POSIX.1-2008.

HISTORY

C89, POSIX.1-2001, POSIX.1-2008.

POSIX.1-2001 did not specify the behavior for flushing of input streams, but the behavior is specified in POSIX.1-2008.

NOTES

Note that [fflush\(\)](#) flushes only the user-space buffers provided by the C library. To ensure that the data is physically stored on disk the kernel buffers must be flushed too, for example, with [sync\(2\)](#) or [fsync\(2\)](#).

SEE ALSO

[fsync\(2\)](#), [sync\(2\)](#), [write\(2\)](#), [fclose\(3\)](#), [fileno\(3\)](#), [fopen\(3\)](#), [fpurge\(3\)](#), [setbuf\(3\)](#), [unlocked_stdio\(3\)](#)

Linux man-pages 6.15

2025-05-17

fflush(3)

Manual page fflush(3) line 1/55 (END) (press h for help or q to quit)

fflush

Cosa succede?

[35.3kb/s]/[163.9kb/s] 100%

fflush(3)

NAME
fflush - flush a stream

LIBRARY
Standard C library

SYNOPSIS
#include <stdio.h>

int fflush(FILE *stream);

DESCRIPTION
For output streams
For input streams
The open status of the stream
If the stream argument is NULL
For a nonlocking counterpart, see unlocked_stdio(3).

RETURN VALUE
Upon successful completion, 0 is returned.
Upon error, -1 is returned, and errno is set to indicate the error.

ERRORS
EBADF stream is not open for writing.

ATTRIBUTES
For an explanation of the attribute values, see Attributes(7).

Interface
fflush()

STANDARDS
C11, POSIX.1-2008.

HISTORY
C89, POSIX.1-2001, POSIX.1-2008.
POSIX.1-2001 did not specify the behavior for flushing of input streams, but the behavior is specified in POSIX.1-2008.

NOTES
Note that fflush() flushes only the user-space buffers provided by the C library. To ensure that the data is physically stored on disk the kernel buffers must be flushed too, for example, with sync(2) or fsync(2).

SEE ALSO
fsync(2), sync(2), write(2), fclose(3), fileno(3), fopen(3), fpurge(3), setbuf(3), unlocked_stdio(3)

Linux man-pages 6.15
Manual page fflush(3) line 1/55 (END) (press h for help or q to quit)

2025-05-17
fflush(3)

DESCRIPTION

For output streams, fflush() forces a write of all user-space buffered data for the given output or update stream via the stream's underlying write function.

For input streams associated with seekable files (e.g., disk files, but not pipes or terminals), fflush() discards any buffered data that has been fetched from the underlying file, but has not been consumed by the application.

The open status of the stream is unaffected.

If the stream argument is NULL, fflush() flushes all open output streams.

For a nonlocking counterpart, see unlocked_stdio(3).

fflush

Cosa succede?

Pertanto, in alcuni casi `fflush(stdin)` potrebbe non fare ciò che ci aspettiamo. (ad alcuni di voi potrebbe andare, a seconda di alcune impostazioni / versioni del compilatore)

Caratteri in più nel Buffer ...

Quindi come si fa?

Caratteri in più nel Buffer ...

Quindi come si fa?

```
ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
```

Com'era prima ...

Caratteri in più nel Buffer ...

Quindi come si fa?

```
ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
```

Com'era prima ...



```
ritorno_scanf_secondo_carattere = scanf(" %c", &secondo_carattere);
```

Nuova Versione!

Caratteri in più nel Buffer ...

Quindi come si fa?

```
ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
```

Com'era prima ...



```
ritorno_scanf_secondo_carattere = scanf(" %c", &secondo_carattere);
```

Nuova Versione!

Caratteri in più nel Buffer ...

Quindi come si fa?

```
ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
```

Com'era prima ...

Ci sono altri modi, verremo
più avanti



```
ritorno_scanf_secondo_carattere = scanf(" %c", &secondo_carattere);
```

Nuova Versione!

Caratteri in più nel Buffer ...

Quindi come si fa?

```
1  #include <stdio.h>
2  int main(){
3      // Variabili
4      int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5      int ritorno_scanf_ultimo_carattere;
6      char primo_carattere, secondo_carattere, carattere_rimanente;
7
8      // acquisizione del primo carattere
9      printf("Inserisci un carattere: ");
10     ritorno_scanf_primo_carattere = scanf(" %c", &primo_carattere);
11
12
13
14     // acquisizione di un numero intero
15     printf("Inserisci un numero intero: ");
16     ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18     // acquisizione del secondo carattere
19     printf("Inserisci un altro carattere: ");
20     ritorno_scanf_secondo_carattere = scanf(" %c", &secondo_carattere);
21
22     printf("Ora consumo eventuali caratteri rimanenti");
23     // scanf dell'invio restato nel buffer
24     ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26     printf("\n\nprimo carattere: %c\n", primo_carattere);
27     printf("numero intero: %d\n", numero_intero);
28     printf("secondo carattere: %c", secondo_carattere);
29     printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30     printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31     printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32     printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33     printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34     printf("Fine del programma\n");
35
36     return 0;
37 }
38
```

Caratteri in più nel Buffer ...

Quindi come si fa?

```
1  #include <stdio.h>
2  int main(){
3      // Variabili
4      int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5      int ritorno_scanf_ultimo_carattere;
6      char primo_carattere, secondo_carattere, carattere_rimanente;
7
8      // acquisizione del primo carattere
9      printf("Inserisci un carattere: ");
10     ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14     // acquisizione di un numero intero
15     printf("Inserisci un numero intero: ");
16     ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18     // acquisizione del secondo carattere
19     printf("Inserisci un altro carattere: ");
20     ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22     printf("Ora consumo eventuali caratteri rimanenti");
23     // scanf dell'invio restato nel buffer
24     ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26     printf("\n\nprimo carattere: %c\n", primo_carattere);
27     printf("numero intero: %d\n", numero_intero);
28     printf("secondo carattere: %c", secondo_carattere);
29     printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30     printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31     printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32     printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33     printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34     printf("Fine del programma\n");
35
36     return 0;
37 }
38
```

Caratteri in più nel Buffer ...

Quindi come si fa?

```
1  #include <stdio.h>
2  int main(){
3      // Variabili
4      int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5      int ritorno_scanf_ultimo_carattere;
6      char primo_carattere, secondo_carattere, carattere_rimanente;
7
8      // acquisizione del primo carattere
9      printf("Inserisci un carattere: ");
10     ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14     // acquisizione di un numero intero
15     printf("Inserisci un numero intero: ");
16     ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18     // acquisizione del secondo carattere
19     printf("Inserisci un altro carattere: ");
20     ritorno_scanf_secondo_carattere = scanf("%c", &secondo_carattere);
21
22     printf("Ora consumo eventuali caratteri rimanenti");
23     // scanf dell'invio restato nel buffer
24     ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26     printf("\n\nprimo carattere: %c\n", primo_carattere);
27     printf("numero intero: %d\n", numero_intero);
28     printf("secondo carattere: %c", secondo_carattere);
29     printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30     printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31     printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32     printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33     printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34     printf("Fine del programma\n");
35
36     return 0;
37 }
38
```

In questo caso vogliamo esplicitamente catturare eventuali \n rimasti

Caratteri in più nel Buffer ...

Quindi come si fa?

```
1  #include <stdio.h>
2  int main(){
3      // Variabili
4      int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5      int ritorno_scanf_ultimo_carattere;
6      char primo_carattere, secondo_carattere, carattere_rimanente;
7
8      // acquisizione del primo carattere
9      printf("Inserisci un carattere: ");
10     ritorno_scanf_primo_carattere = scanf(" %c", &primo_carattere);
11
12
13
14     // acquisizione di un numero intero
15     printf("Inserisci un numero intero: ");
16     ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18     // acquisizione del secondo carattere
19     printf("Inserisci un altro carattere: ");
20     ritorno_scanf_secondo_carattere = scanf(" %c", &secondo_carattere);
21
22     printf("Ora consumo eventuali caratteri rimanenti");
23     // scanf dell'invio restato nel buffer
24     ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26     printf("\n\nprimo carattere: %c\n", primo_carattere);
27     printf("numero intero: %d\n", numero_intero);
28     printf("secondo carattere: %c", secondo_carattere);
29     printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30     printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31     printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32     printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33     printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34     printf("Fine del programma\n");
35
36     return 0;
37 }
38
```

Caratteri in più nel Buffer ...

Quindi come si fa?

```
1  #include <stdio.h>
2  int main(){
3      // Variabili
4      int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5      int ritorno_scanf_ultimo_carattere;
6      char primo_carattere, secondo_carattere, carattere_rimanente;
7
8      // acquisizione del primo carattere
9      printf("Inserisci un carattere: ");
10     ritorno_scanf_primo_carattere = scanf(" %c", &primo_carattere);
11
12
13
14     // acquisizione di un numero intero
15     printf("Inserisci un numero intero: ");
16     ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18     // acquisizione del secondo carattere
19     printf("Inserisci un altro carattere: ");
20     ritorno_scanf_secondo_carattere = scanf(" %c", &secondo_carattere);
21
22     printf("Ora consumo eventuali caratteri rimanenti");
23     // scanf dell'invio restato nel buffer
24     ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26     printf("\n\nprimo carattere: %c\n", primo_carattere);
27     printf("numero intero: %d\n", numero_intero);
28     printf("secondo carattere: %c", secondo_carattere);
29     printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30     printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31     printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32     printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33     printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34     printf("Fine del programma\n");
35
36     return 0;
37 }
38
```

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimanentir

primo carattere: t
numero intero: 5
secondo carattere:
ultimo carattere (invio): r
Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma
```

scanf("%c" , ...)
(precedente output)

Caratteri in più nel Buffer ...

Quindi come si fa?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf(" %c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione di un altro carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_primo_carattere = scanf(" %c", &primo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti\n");
23    // scanf dell'ultimo carattere (invio):
24    ritorno_scanf_ultimo_carattere = scanf(" %c", &carattere_rimanente);
25
26    printf("\n\n");
27    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
28    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
29    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_primo_carattere);
30    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
31    printf("Fine del programma\n");
32
33    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
34    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
35    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_primo_carattere);
36    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
37
38    return 0;
}
```

scanf(" %c", ...)
(nuovo output)

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimanenti

primo carattere: t
numero intero: 5
secondo carattere:
ultimo carattere (invio): r
Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma
```

scanf("%c", ...)
(precedente output)

Caratteri in più nel Buffer ...

Quindi come si fa?

Ora il secondo carattere cattura la "r" anziché l'invio! Infatti non va nemmeno a capo prima di stampare "ultimo carattere")

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf("%c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_numero_intero = scanf("%d", &numero_intero);
17
18    // acquisizione di un altro carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_ultimo_carattere = scanf("%c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti\n");
23    // scanf dell'ultimo carattere (invio):
24    ritorno_scanf_ultimo_carattere = scanf("%c", &secondo_carattere);
25
26    printf("\n\n");
27    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
28    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_numero_intero);
29    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_ultimo_carattere);
30    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
31    printf("Fine del programma\n");
32
33    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
34    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_numero_intero);
35    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_ultimo_carattere);
36    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
37
38    return 0;
39 }
```

`scanf(" %c", ...)`
(nuovo output)

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: Ora consumo eventuali caratteri rimanenti

primo carattere: t
numero intero: 5
secondo carattere:
ultimo carattere (invio): r
Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma
```

`scanf("%c", ...)`
(precedente output)

Caratteri in più nel Buffer ...

Quindi come si fa?

```
1  #include <stdio.h>
2  int main(){
3      // Variabili
4      int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5      int ritorno_scanf_ultimo_carattere;
6      char primo_carattere, secondo_carattere, carattere_rimanente;
7
8      // acquisizione del primo carattere
9      printf("Inserisci un carattere: ");
10     ritorno_scanf_primo_carattere = scanf(" %c", &primo_carattere);
11
12
13
14     // acquisizione di un numero intero
15     printf("Inserisci un numero intero: ");
16     ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18     // acquisizione di un altro carattere
19     printf("Inserisci un altro carattere: ");
20     ritorno_scanf_secondo_carattere = scanf(" %c", &secondo_carattere);
21
22     printf("Ora consumo eventuali caratteri rimanenti\n");
23     // scanf dell'ultimo carattere (invio):
24     ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26     printf("\n\nRitorno della scanf per il primo carattere: %d", ritorno_scanf_primo_carattere);
27     printf("Ritorno della scanf per l'intero: %d", ritorno_scanf_intero);
28     printf("Ritorno della scanf per il secondo carattere: %d", ritorno_scanf_secondo_carattere);
29     printf("Ritorno della scanf per l'ultimo carattere (invio): %d", ritorno_scanf_ultimo_carattere);
30     printf("Fine del programma\n");
31
32     return 0;
33 }
```

Come correttamente ci
aspettavamo, l'ultima scanf
senza lo spazio prima del %c
cattura il rimanente invio rimasto
nel buffer

`scanf(" %c", ...)`
(nuovo output)

`scanf("%c", ...)`
(precedente output)

```
carattere: t
ro intero: 5
o carattere: Ora consumo eventuali caratteri rimanenti
t
secondo carattere:
ultimo carattere (invio): r
Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma
```

Caratteri in più nel Buffer ...

Quindi come si fa?

```
1  #include <stdio.h>
2  int main(){
3      // Variabili
4      int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5      int ritorno_scanf_ultimo_carattere;
6      char primo_carattere, secondo_carattere, carattere_rimanente;
7
8      // acquisizione del primo carattere
9      printf("Inserisci un carattere: ");
10     ritorno_scanf_primo_carattere = scanf(" %c", &primo_carattere);
11
12
13
14     // acquisizione di un numero intero
15     printf("Inserisci un numero intero: ");
16     ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18     // acquisizione del secondo carattere
19     printf("Inserisci un altro carattere: ");
20     ritorno_scanf_secondo_carattere = scanf(" %c", &secondo_carattere);
21
22     printf("Ora consumo eventuali caratteri rimanenti");
23     // scanf dell'invio restato nel buffer
24     ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26     printf("\n\nprimo carattere: %c\n", primo_carattere);
27     printf("numero intero: %d\n", numero_intero);
28     printf("secondo carattere: %c", secondo_carattere);
29     printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30     printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31     printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32     printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33     printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34     printf("Fine del programma\n");
35
36     return 0;
37 }
38
```

```
Inserisci un carattere:      t
Inserisci un numero intero:      5
Inserisci un altro carattere:    r
Ora consumo eventuali caratteri rimanenti

primo carattere: t
numero intero: 5
secondo carattere: rultimo carattere (invio):

Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma
```

scanf(" %c", ...)
(nuovo output)

Caratteri in più nel Buffer ...

Quindi come si fa?

```

1  #include <stdio.h>
2  int main(){
3      // Variabili
4      int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5      int ritorno_scanf_ultimo_carattere;
6      char primo_carattere, secondo_carattere, carattere_rimanente;
7
8      // acquisizione del primo carattere
9      printf("Inserisci un carattere: ");
10     ritorno_scanf_primo_carattere = scanf(" %c", &primo_carattere);
11
12
13
14     // acquisizione di un numero intero
15     printf("Inserisci un numero intero: ");
16     ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18     // acquisizione del secondo carattere
19     printf("Inserisci un altro carattere: ");
20     ritorno_scanf_secondo_carattere = scanf(" %c", &secondo_carattere);
21
22     printf("Ora consumo eventuali caratteri rimanenti");
23     // scanf dell'invio restato nel buffer
24     ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26     printf("\n\nprimo carattere: %c\n", primo_carattere);
27     printf("numero intero: %d\n", numero_intero);
28     printf("secondo carattere: %c", secondo_carattere);
29     printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30     printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31     printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32     printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33     printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34     printf("Fine del programma\n");
35
36     return 0;
37 }
38

```

```

Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: r
Ora consumo eventuali caratteri rimanenti

primo carattere: t
numero intero: 5
secondo carattere: rultimo carattere (invio):

Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma

```

scanf(" %c", ...)
(nuovo output)

Caratteri in più nel Buffer ...

Quindi come si fa?

```
1 #include <stdio.h>
2 int main(){
3     // Variabili
4     int numero_intero, ritorno_scanf_intero, ritorno_scanf_primo_carattere, ritorno_scanf_secondo_carattere;
5     int ritorno_scanf_ultimo_carattere;
6     char primo_carattere, secondo_carattere, carattere_rimanente;
7
8     // acquisizione del primo carattere
9     printf("Inserisci un carattere: ");
10    ritorno_scanf_primo_carattere = scanf(" %c", &primo_carattere);
11
12
13
14    // acquisizione di un numero intero
15    printf("Inserisci un numero intero: ");
16    ritorno_scanf_intero = scanf("%d", &numero_intero);
17
18    // acquisizione del secondo carattere
19    printf("Inserisci un altro carattere: ");
20    ritorno_scanf_secondo_carattere = scanf(" %c", &secondo_carattere);
21
22    printf("Ora consumo eventuali caratteri rimanenti");
23    // scanf dell'invio restato nel buffer
24    ritorno_scanf_ultimo_carattere = scanf("%c", &carattere_rimanente);
25
26    printf("\n\nprimo carattere: %c\n", primo_carattere);
27    printf("numero intero: %d\n", numero_intero);
28    printf("secondo carattere: %c", secondo_carattere);
29    printf("ultimo carattere (invio): %c\n", carattere_rimanente);
30    printf("Ritorno della scanf per il primo carattere: %d\n", ritorno_scanf_primo_carattere);
31    printf("Ritorno della scanf per l'intero: %d\n", ritorno_scanf_intero);
32    printf("Ritorno della scanf per il secondo carattere: %d\n", ritorno_scanf_secondo_carattere);
33    printf("Ritorno della scanf per l'ultimo carattere (invio): %d\n", ritorno_scanf_ultimo_carattere);
34    printf("Fine del programma\n");
35
36    return 0;
37 }
38
```

Il programma funziona come ci aspettiamo anche se inseriamo degli spazi nell'input, che vengono consumati e non catturati dalle scanf in questo modo

```
Inserisci un carattere: t
Inserisci un numero intero: 5
Inserisci un altro carattere: r
Ora consumo eventuali caratteri rimanenti
```

```
primo carattere: t
numero intero: 5
secondo carattere: r
ultimo carattere (invio):
Ritorno della scanf per il primo carattere: 1
Ritorno della scanf per l'intero: 1
Ritorno della scanf per il secondo carattere: 1
Ritorno della scanf per l'ultimo carattere (invio): 1
Fine del programma
```

`scanf(" %c", ...)`
(nuovo output)

Esercizio 02

Calcolo del Modulo

Testo

Si scriva un programma che prenda in input un numero e se ne stampi il modulo.

Esempi

$$2 \rightarrow 2 \quad -2 \rightarrow 2$$

Esercizio 03

Equazione di Secondo Grado

Testo

Si scriva un programma che prenda tre numeri in input che rappresentano i coefficienti dell'equazione di secondo grado. Si stampino a schermo le radici dell'equazione in questione (se esistono).

Esempio

$$-x^2 - 2x + 1 \rightarrow x_1 = 1, x_2 = 2$$

Bonus

Si stampi a schermo l'equazione

Esercizio 04 – Già visto ...

Massimo tra Tre Numeri

Testo

Si scriva un programma che prenda in input tre numeri e che stampi a schermo il massimo dei tre.

Esempio

$$a = 3 \quad b = 7 \quad c = 5 \quad \rightarrow \quad 7$$

Bonus

Stampare il minimo dei tre

Esercizio 05

Ordinamento di Tre Numeri

Testo

Si scriva un programma che prenda in input tre numeri. Il programma deve stampare i tre numeri acquisiti a schermo in ordine ascendente.

Esempio

$$a = 3 \quad b = 7 \quad c = 5 \quad \rightarrow \quad (3,5,7)$$

Bonus

Stampare i numeri in ordine discendente

Esercizio 06

Inversione di un Numero a Due Cifre

Testo

Si scriva un programma che prenda in input un numero intero positivo e che sia composto esattamente da due cifre. Si stampi a schermo il numero con le cifre invertite.

Esempi

$31 \rightarrow 13$ $137 \rightarrow \text{NO}$ $-31 \rightarrow \text{NO}$

Esercizio 07

Lettera Successiva

Testo

Si scriva un programma che prenda in input una lettera (minuscola o maiuscola) e si stampi (se possibile) la lettera successiva in ordine alfabetico.

Esempi

$$A \rightarrow B \quad c \rightarrow d \quad z \rightarrow z$$

Esercizio 08

Maiuscola o Minuscola Corrispondente

Testo

Si scriva un programma che prenda in input una lettera minuscola/maiuscola e si stampi la lettera corrispondente maiuscola/minuscola.

Esempi

$$A \rightarrow a \quad c \rightarrow C \quad z \rightarrow Z$$

Es. 09 Leggi di De Morgan

Formulazione

$$\neg(a \wedge b) = \neg a \vee \neg b$$
$$\neg(a \vee b) = \neg a \wedge \neg b$$

Verifica Sperimentale

Implementare un programma che verifichi che due caratteri inseriti dall'utente non siano entrambe lettere maiuscole (prima formula). Provare a verificare il corretto funzionamento con entrambe le formulazioni

A,a = ok, a,A = ok, AA = ko, aa=ok

Esercizio per la prossima volta

Sasso carta forbice e ... do while!

Testo

Implementare una partita di sasso carta e forbice per due giocatori:

- Non giocatore vs computer
- Le giocate sono sequenziali, immaginate il programma come un “verificatore” per capire, date le giocate di una partita fisica, chi abbia vinto

Esercizio per la prossima volta

Sasso carta forbice

Testo

Implementare una par

– Non giocatore vs com

– Le giocate sono sequ

“verificatore” per capi

ocatori:

ne un

chi abbia vinto

```
#include <stdio.h>
int main() {
    int p1, p2;
    int vittoria=0;

    do {
        printf("\n0 per sasso, 1 per carta, 2 per forbice\n");

        printf("player 1: ");
        scanf("%d", &p1);
        printf("player 2: ");
        scanf("%d", &p2);

        if ( ... ) { // check mosse non valide
            printf("Mossa non valida!\n");
        } else if (...) { // check pareggio
            printf("Pareggio!\n");
        } else if (...) { // check vittorie p1
            printf("Vince player 1!\n");
            vittoria = 1;
        } else {
            printf("Vince player 2!\n");
            vittoria = 1;
        }

    } while (!vittoria);

    return 0;
}
```

Thank You!

Mail: luca1.alessandrini@polimi.it

Website: <https://alessandriniluca.github.io/>