

Esercitazione 05

2025/10/10 – Array + Stringhe

L. Alessandrini

Credits: A. Montenegro

Esercizi parte 1 di 2

Array

Esercizio 01

Conversione in Binario

Testo

Scrivere un programma che permetta all'utente di inserire un numero intero positivo. Scrivere la codifica binaria del numero in un array di lunghezza massima 10. Stampare il numero in binario.

Esempio

10 -> 0000001010

Esercizio 02

Coppie di Numeri di cui Uno è il Doppio Dell'altro

Testo

Scrivere un programma che permetta all'utente di inserire una sequenza di numeri. Il programma cerca successivamente le coppie di numeri per cui il primo è il doppio del secondo e le stampa.

Esempio

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10] -> 2 – 1, 4 – 2, 6 – 3, 8 – 4, 10 – 5

Esercizio 03

Set di un Array

Testo

Scrivere un programma che permetta all'utente di inserire una sequenza di numeri. Il programma salva e stampa il set dei numeri inseriti (i.e., stampa la sequenza senza ripetizioni).

Esempio

`[1, 2, 3, 2, 1, 3, 4] -> {1, 2, 3, 4}`

Esercizio 04

Unione di Set

Testo

Scrivere un programma che permetta all'utente di inserire due sequenze di numeri. Il programma salva e stampa il set unione dei set relativi alle due sequenze di numeri inseriti.

Esempio

$[1, 2, 3, 2]$ e $[1, 3, 4] \rightarrow \{1, 2, 3, 4\}$

Esercizio 05

Intersezione di Set

Testo

Scrivere un programma che permetta all'utente di inserire due sequenze di numeri. Il programma salva e stampa il set intersezione dei set relativi alle due sequenze di numeri inseriti.

Esempio

$[1, 2, 3, 2]$ e $[1, 3, 4] \rightarrow \{1, 3\}$

Recap

Stringhe

Cosa sono le Stringhe

Introduzione

Le **stringhe** sono **particolari array** di caratteri che hanno **come ultimo elemento** il carattere **'\0'**.

{'c', 'i', 'a', 'o', '\0'} -> stringa (lunghezza = 5)

vs

{'c', 'i', 'a', 'o'} -> array di caratteri (lunghezza = 4)

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100
char stringa[MAX_LEN];
// [...]
printf("%s", stringa);
```

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100
char stringa[MAX_LEN];
// [...]
printf("%s", stringa);
```

Attenzione! In questi 100 caratteri, deve rientrare anche il terminatore

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100
char stringa[MAX_LEN];
// [...]
printf("%s", stringa);
```



```
int i = 0;
while(stringa[i] != '\0'){
    printf("%c", stringa[i]);
    i++;
}
```

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100
char stringa[MAX_LEN];
scanf("%s", stringa);
// oppure
gets(stringa);
```

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100
char stringa[MAX_LEN];
scanf("%s", stringa);
// oppure
gets(stringa);
```

- Consuma caratteri da stdin fino a quando non trova uno **spazio** bianco oppure uno **'\n'**

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100  
char stringa[MAX_LEN];  
scanf("%s", stringa);  
// oppure  
gets(stringa);
```

- Consuma caratteri da stdin fino a quando non trova uno **spazio** bianco oppure uno **'\n'**
- Ciò che **non consuma** rimane nello stdin

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100
char stringa[MAX_LEN];
scanf("%s", stringa);
// oppure
gets(stringa);
```

- Consuma caratteri da stdin fino a quando non trova uno **spazio** bianco oppure uno **'\n'**
- Ciò che **non consuma** rimane nello stdin
- Può causare **buffer overflow**

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100  
  
char stringa[MAX_LEN];  
scanf("%s", stringa);  
// oppure  
gets(stringa);
```

- **Consuma** caratteri da stdin fino a quando non trova uno **spazio** bianco oppure uno **'\n'**
- Ciò che **non consuma** rimane nello stdin
- Può causare **buffer overflow**
- Per il momento definire MAX_LEN **abbastanza grande**

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

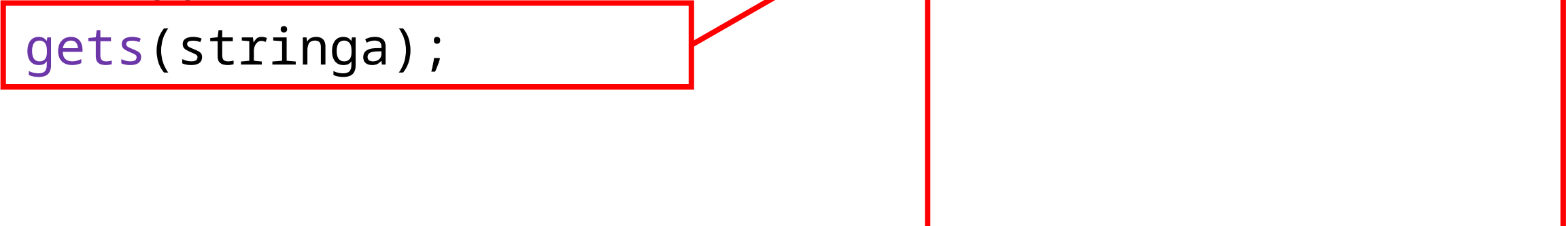
```
#define MAX_LEN 100
char stringa[MAX_LEN];
scanf("%s", stringa);
// oppure
gets(stringa);
```

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100  
  
char stringa[MAX_LEN];  
    scanf("%s", stringa);  
    // oppure  
    gets(stringa);
```



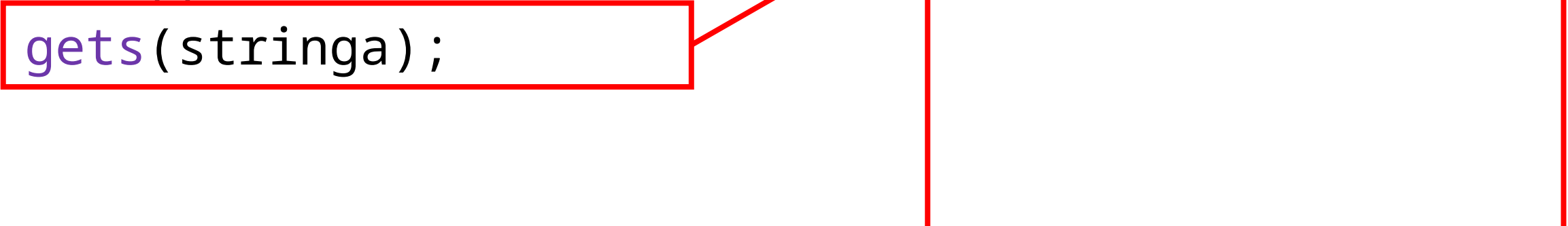
- Consuma caratteri da stdin fino a quando non trova uno '\n' (che viene consumato ma non inserito nella stringa)

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100  
  
char stringa[MAX_LEN];  
    scanf("%s", stringa);  
    // oppure  
    gets(stringa);
```



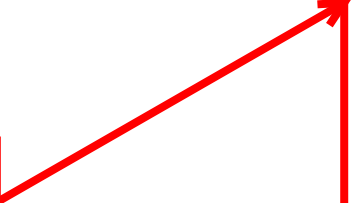
- Consuma caratteri da stdin fino a quando non trova uno '\n' (che viene consumato ma non inserito nella stringa)
- Acquisisce spazi bianchi

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100  
  
char stringa[MAX_LEN];  
    scanf("%s", stringa);  
    // oppure  
    gets(stringa);
```



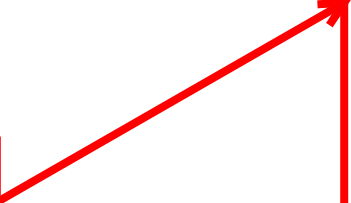
- **Consuma** caratteri da stdin fino a quando non trova uno '\n' (che viene consumato ma non inserito nella stringa)
- Acquisisce spazi bianchi
- Può causare **buffer overflow**

Cosa sono le Stringhe

Introduzione

A differenza dei normali array, le stringhe possono essere **acquisite** e **stampate** anche senza cicli

```
#define MAX_LEN 100  
  
char stringa[MAX_LEN];  
scanf("%s", stringa);  
// oppure  
gets(stringa);
```



- **Consuma** caratteri da stdin fino a quando non trova uno '\n' (che viene consumato ma non inserito nella stringa)
- Acquisisce spazi bianchi
- Può causare **buffer overflow**
- Per il momento definire MAX_LEN **abbastanza grande**

Cosa sono le Stringhe

Introduzione

gets non è più riconosciuta nelle versioni recenti di C, per via della vulnerabilità che introduce (non abbiamo controllo su quanti caratteri possiamo inserire) si può usare fgets!

Cosa sono le Stringhe

Introduzione

gets non è più riconosciuta nelle versioni recenti di C, per via della vulnerabilità che introduce (non abbiamo controllo su quanti caratteri possiamo inserire) si può usare fgets!

Aspect	gets()	fgets()
Buffer Size Control	No size control; prone to buffer overflow.	Allows specifying the maximum size; safe.
Newline Handling	Discards the newline character.	Retains the newline character.
Error Handling	Limited error handling capabilities.	Returns NULL on error or EOF.
Status	Deprecated in C11 and later.	Recommended for modern use.

Fonte 1: <https://www.geeksforgeeks.org/c/gets-in-c/>

Fonte 2: <https://www.geeksforgeeks.org/c/fgets-function-in-c/>

Cosa sono le Stringhe

Introduzione

gets non è più riconosciuta nelle versioni recenti di C, per via della vulnerabilità che introduce (non abbiamo controllo su quanti caratteri possiamo inserire) si può usare fgets!

Aspect	gets()	fgets()
Buffer Size Control	No size control; prone to buffer overflow.	Allows specifying the maximum size; safe.
Newline Handling	Discards the newline character.	Retains the newline character.
Error Handling	Limited error handling capabilities.	Returns NULL on error or EOF.
Status	Deprecated in C11 and later.	Recommended for modern use.

```
fgets(buff, n, stream);
```

- **buff**: stringa in cui vogliamo salvare
- **n**: Massimo numero di caratteri da leggere, includendo il terminatore
- **stream**: Da dove leggere i caratteri, per ora stdin

Fonte 1: <https://www.geeksforgeeks.org/c/gets-in-c/>

Fonte 2: <https://www.geeksforgeeks.org/c/fgets-function-in-c/>

Cosa sono le St

Introduzione

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

// MACRO
#define N 10

int main(int argc, char* argv[]) {
    char str[N], str2[N];

    printf("Inserisci una stringa con spazi: ");
    // leggiamo da input una stringa con spazi
    fgets(str, N, stdin);

    // verifichiamo se, dopo la lettura,
    // rimane qualcosa nel buffer se superiamo
    // i 9 caratteri in input
    fgets(str2, N, stdin);

    printf("Stringa inserita: %s\n", str);
    printf("Rimanente nel buffer: %s\n", str2);
    return 0;
}
```

Co

Intr

Inserisci una stringa con spazi: prova inserimento
Stringa inserita: prova ins
Rimanente nel buffer: erimento

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

```
#include <string.h>
```

```
int main(int argc, char* argv[]) {  
    char str[N], str2[N];  
  
    printf("Inserisci una stringa con spazi: ");  
    // leggiamo da input una stringa con spazi  
    fgets(str, N, stdin);  
  
    // verifichiamo se, dopo la lettura,  
    // rimane qualcosa nel buffer se superiamo  
    // i 9 caratteri in input  
    fgets(str2, N, stdin);  
  
    printf("Stringa inserita: %s\n", str);  
    printf("Rimanente nel buffer: %s\n", str2);  
    return 0;  
}
```

Cosa sono le Stringhe

Introduzione

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

// MACRO
#define N 10
```

A differenza della gets, consuma l'invio e lo inserisce nella stringa:

Inserisci una stringa con spazi: prova
inserimento

Stringa inserita: prova

Rimanente nel buffer: inserimen

```
fgets(str2, N, stdin);

printf("Stringa inserita: %s\n", str);
printf("Rimanente nel buffer: %s\n", str2);
return 0;
}
```

Cosa sono le Stringhe

Introduzione

A differenza della gets, consuma una stringa:

Inserisci una stringa con spazi: prova
inserimento

Stringa inserita: prova

Rimanente nel buffer: inserimen

Digitando
"prova<invio>inserimento",
ha salvato prova<invio>

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

// MACRO
#define N 100
```

```
fgets(str2, N, stdin);
```

```
printf("Stringa inserita: %s\n", str);
printf("Rimanente nel buffer: %s\n", str2);
return 0;
}
```

Cosa sono le Stringhe

Introduzione

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

// MACRO
#define N 10
```

A differenza della gets, consuma l'invio e lo inserisce nella stringa:

Inserisci una stringa con spazi: prova
inserimento
Stringa inserita: prova

Rimanente nel buffer: inserimen

Dove è finito "to"?

```
fgets(str2, N, stdin);

printf("Stringa inserita: %s\n", str);
printf("Rimanente nel buffer: %s\n", str2);
return 0;
```

Cosa sono le Stringhe

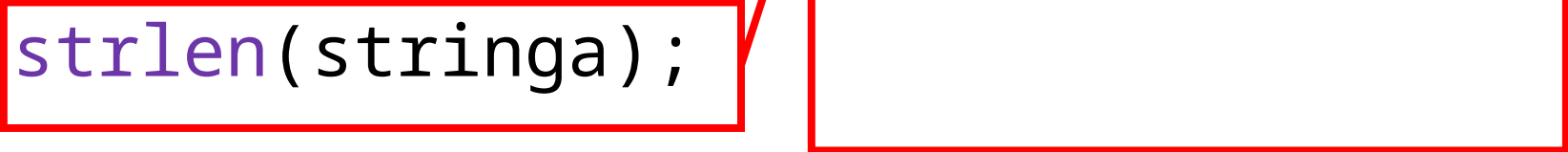
Lunghezza di una stringa

```
#define MAX_LEN 100  
char stringa[MAX_LEN];  
int lunghezza = strlen(stringa);
```

Cosa sono le Stringhe

Lunghezza di una stringa

```
#define MAX_LEN 100  
char stringa[MAX_LEN];  
int lunghezza = strlen(stringa);
```

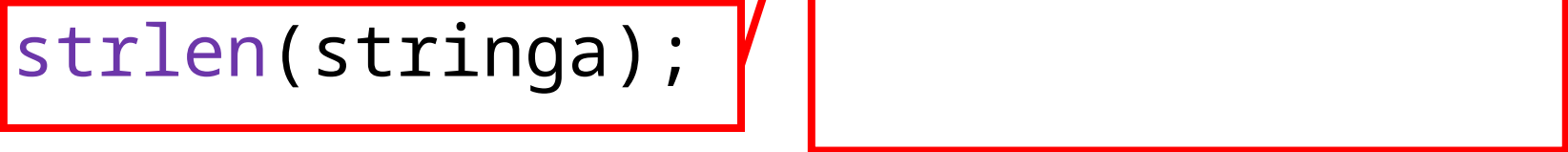


- Restituisce il **numero di caratteri** in stringa

Cosa sono le Stringhe

Lunghezza di una stringa

```
#define MAX_LEN 100  
char stringa[MAX_LEN];  
int lunghezza = strlen(stringa);
```



- Restituisce il **numero di caratteri** in stringa
- **Non** si tiene conto del carattere '\0'

Cosa sono le Stringhe

Lunghezza di una stringa

```
#define MAX_LEN 100  
char stringa[MAX_LEN];  
int lunghezza = strlen(stringa);
```

- Restituisce il **numero di caratteri** in stringa
- **Non** si tiene conto del carattere '\0'
- Ciò che restituisce è l'indice in stringa del carattere di terminazione

Cosa sono le Stringhe

Lunghezza di una stringa

```
#define MAX_LEN 100
char stringa[MAX_LEN];
int lunghezza = strlen(stringa);
```

- Restituisce il **numero di caratteri** in stringa
- **Non** si tiene conto del carattere '\0'
- Ciò che restituisce è l'indice in stringa del carattere di terminazione

Supponiamo la stringa:

'c'	'i'	'a'	'o'	'\0'
-----	-----	-----	-----	------

Cosa sono le Stringhe

Lunghezza di una stringa

```
#define MAX_LEN 100
char stringa[MAX_LEN];
int lunghezza = strlen(stringa);
```

- Restituisce il **numero di caratteri** in stringa
- **Non** si tiene conto del carattere '\0'
- Ciò che restituisce è l'indice in stringa del carattere di terminazione

Supponiamo la stringa:

'c'	'i'	'a'	'o'	'\0'
0	1	2	3	4

- `strlen(stringa)` restituisce 4
- `stringa[4]` corrisponde a '\0'

Cosa sono le Stringhe

Lunghezza di una stringa

```
#define MAX_LEN 100
```

```
char stringa[MAX_LEN];
```

```
int lunghezza = strlen(stringa);
```

```
int lunghezza = 0;  
while(stringa[lunghezza] != '\0'){  
    lunghezza++;  
}  
return lunghezza;
```

Corrisponde a ...

Supponiamo la stringa:

'c'	'i'	'a'	'o'	'\0'
0	1	2	3	4

- `strlen(stringa)` restituisce 4
- `stringa[4]` corrisponde a `'\0'`

Cosa sono le Stringhe

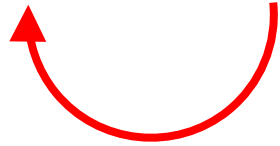
Copia di stringhe

```
strcpy(stringa1, stringa2);
```

Cosa sono le Stringhe

Copia di stringhe

```
strcpy(stringa1, stringa2);
```

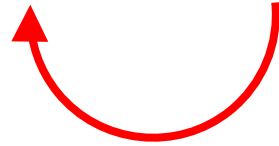


`stringa2` copiata
dentro `stringa1`

Cosa sono le Stringhe

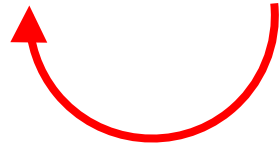
Copia di stringhe

```
strcpy(stringa1, stringa2);
```



`stringa2` copiata
dentro `stringa1`

```
strcpy(stringa1, "Ciao");
```

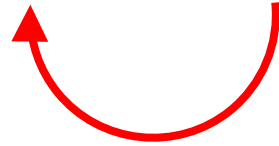


`"Ciao"` copiato
dentro `stringa1`

Cosa sono le Stringhe

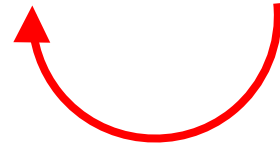
Copia di stringhe

```
strcpy(stringa1, stringa2);
```



`stringa2` copiata
dentro `stringa1`

```
strcpy(stringa1, "Ciao");
```



`"Ciao"` copiato
dentro `stringa1`

In alternativa si può usare un
ciclo for e manipolare ogni
elemento della stringa

Cosa sono le Stringhe

Copia di stringhe

`stringa2` copiata
dentro `stringa1`

```
strcpy(stringa1, stringa2);
```



```
int i = 0;  
while (stringa2[i] != '\0') {  
    stringa1[i] = stringa2[i];  
    i++;  
}  
stringa1[i] = '\0';
```

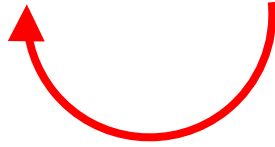
Cosa sono le Stringhe

Concatenazione di stringhe

Cosa sono le Stringhe

Concatenazione di stringhe

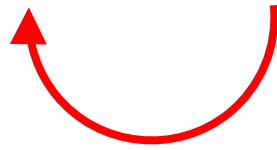
```
strcat(stringa1, stringa2);
```



Cosa sono le Stringhe

Concatenazione di stringhe

```
strcat(stringa1, stringa2);
```

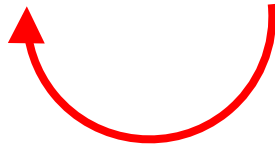


stringa2
concatenata a
stringa1,
nella stessa
stringa1

Cosa sono le Stringhe

Concatenazione di stringhe

```
strcat(stringa1, stringa2);
```



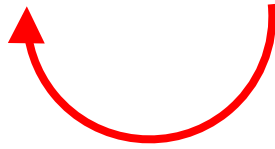
stringa2
concatenata a
stringa1,
nella stessa
stringa1

```
stringa1 ← abc  
stringa2 ← def  
Dopo la strcat  
stringa1 ← abcdef
```

Cosa sono le Stringhe

Concatenazione di stringhe

```
strcat(stringa1, stringa2);
```



stringa2
concatenata a
stringa1,
nella stessa
stringa1

```
strcat(stringa1, "Ciao");
```

"Ciao"
concatenata a
stringa1,

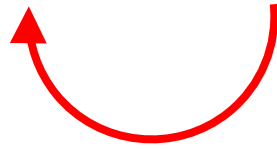
```
stringa1 ← abc  
stringa2 ← def  
Dopo la strcat  
stringa1 ← abcdef
```

```
stringa1 ← abc  
Dopo la strcat  
stringa1 ← abcciao
```

Cosa sono le Stringhe

Concatenazione di stringhe

```
strcat(stringa1, stringa2);
```



stringa2
concatenata a
stringa1,
nella stessa
stringa1

```
strcat(stringa1, "Ciao");
```

"Ciao"
concatenata a
stringa1,

In alternativa si può usare un ciclo for e manipolare ogni elemento della stringa

```
stringa1 ← abc  
stringa2 ← def  
Dopo la strcat  
stringa1 ← abcdef
```

```
stringa1 ← abc  
Dopo la strcat  
stringa1 ← abcciao
```


Cosa sono le Stringhe

Concatenazione di stringhe

```
strcat(stringa1, stringa2);
```

stringa2
concatenata a
stringa1,
nella stessa
stringa1

```
stringa1 ← abc  
stringa2 ← def  
Dopo la strcat  
stringa1 ← abcdef
```

```
int i = 0, lunghezza1 = strlen(stringa1);  
while (stringa2[i] != '\0') {  
    stringa1[lunghezza1 + i] = stringa2[i];  
    i++;  
}  
stringa1[lunghezza1 + i] = '\0';
```

Cosa sono le Stringhe

Confronto di stringhe

Cosa sono le Stringhe

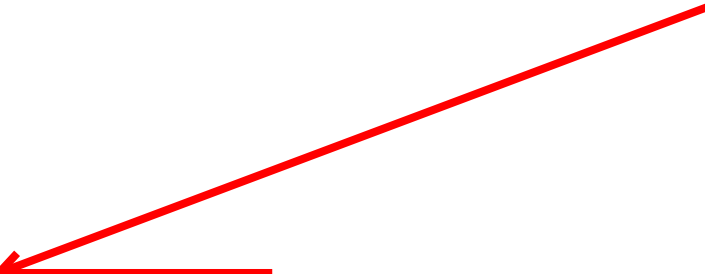
Confronto di stringhe

```
strcmp(stringa1, stringa2);
```

Cosa sono le Stringhe

Confronto di stringhe

```
strcmp(stringa1, stringa2);
```



=0

Se le stringhe
sono uguali

Cosa sono le Stringhe

Confronto di stringhe

```
strcmp(stringa1, stringa2);
```

=0

Se le stringhe
sono uguali

<0

Se la prima stringa
è minore della
seconda (in ASCII)

Cosa sono le Stringhe

Confronto di stringhe

```
strcmp(stringa1, stringa2);
```

=0

Se le stringhe
sono uguali

<0

Se la prima stringa
è minore della
seconda (in ASCII)

>0

Se la prima stringa è
maggiore della
seconda (in ASCII)

Cosa sono le Stringhe

Confronto di stringhe

```
strcmp(stringa1, stringa2);
```

In alternativa si può usare
un ciclo for e manipolare
ogni elemento della stringa

=0

Se le stringhe
sono uguali

<0

Se la prima stringa
è minore della
seconda (in ASCII)

>0

Se la prima stringa è
maggiore della
seconda (in ASCII)

Cosa sono le Stringhe

Confronto di stringhe

```
strcmp(stringa1, stringa2);
```

=0

Se le stringhe
sono uguali

<0

Se la prima stringa
è minore della
seconda (in ASCII)

>0

Se la prima stringa è
maggiore della
seconda (in ASCII)

```
int i = 0;
while (stringa1[i] != '\0' && stringa1[i] == stringa2[i])
{
    i++;
}
return stringa1[i] - stringa2[i];
```


Cosa sono le Stringhe

Esempio Programma

```
// Librerie
#include <stdio.h>
#include <string.h>

// MACRO
#define N 10

int main(int argc, char* argv[]) {
    char str[N]="aaa";
    char str2[N]="ccccc";

    printf("differenza tra str e str2: %d\n", strcmp(str, str2));
    printf("str lengths: str: %d, str2: %d\n", strlen(str),
strlen(str2));

    printf("terminator character for str: %d\n",
str[strlen(str)]);
}
```

```
// Librerie
#include <stdio.h>
#include <string.h>
```

```
// MACRO
#define N 10
```

```
int main(int argc, char* argv[]) {
    char str[N]="aaa";
    char str2[N]="ccccc";

    printf("differenza tra str e str2: %d\n", strcmp(str, str2));
    printf("str lengths: str: %d, str2: %d\n", strlen(str),
strlen(str2));

    printf("terminator character for str: %d\n",
str[strlen(str)]);
}
```

differenza tra str e str2: -2
str lengths: str: 3, str2: 5
terminator character for str: 0

Cosa sono le Stringhe

Esempio Programma

Disclaimer: ciò che seguirà, fino agli esercizi, è una semplificazione di ciò che accade realmente. Capiremo meglio quando vedremo lo stack!

Cosa sono le Str

Cosa vi aspettate come output?

(abbiamo violato il constraint della lunghezza N, per str servirebbero 11 caratteri)

```
// Librerie
#include <stdio.h>
#include <string.h>

// MACRO
#define N 10

int main(int argc, char* argv[]) {
    char str[N]="abcdefghij";
    char str2[N]="ccccc";
    char c;

    printf("differenza tra str e str2: %d\n",
    strcmp(str, str2));
    printf("str lengths: str: %d,
    str2: %d\n", strlen(str), strlen(str2));

    printf("str1: %s\n", str);

    printf("str2: %s\n", str2);
}
```

Cosa sono le Str

```
// Li differenza tra str e str2: -2
#include <string.h>
#include <stdio.h>

// Main
#define N 10

int main(int argc, char* argv[]) {
    char str[N]="abcdefghij";
    char str2[N]="ccccc";
    char c;

    printf("differenza tra str e str2: %d\n",
strcmp(str, str2));
    printf("str lengths: str: %d,
str2: %d\n", strlen(str), strlen(str2));

    printf("str1: %s\n", str);

    printf("str2: %s\n", str2);
}
```

Cosa sono le Stringhe

In memoria ho
allocato 10 cellette, \0
è stato inserito
all'undicesima, ed è
stato sovrascritto dalla
prima "c" della
seconda stringa

```
// Li differenza tra str e str2: -2
#include <string.h>
#include <stdio.h>

// Main
#define N 10

int main(int argc, char* argv[]) {
    char str[N]="abcdefghij";
    char str2[N]="ccccc";
    char c;

    printf("differenza tra str e str2: %d\n",
strcmp(str, str2));
    printf("str lengths: str: %d,
str2: %d\n", strlen(str), strlen(str2));

    printf("str1: %s\n", str);

    printf("str2: %s\n", str2);
}
```

Cosa

```
#include <stdio.h>
#include <string.h>
#define N 10

int main(int argc, char* argv[]) {
    char str[N];
    char str2[N];
    printf("inserisci str: ");
    scanf("%s", str);
    getchar(); // consumare \n
    printf("inserisci str2: ");
    scanf("%s", str2);

    printf("differenza tra str e str2: %d\n", strcmp(str, str2));
    printf("str lengths: str: %d, str2: %d\n", strlen(str), strlen(str2));

    printf("str1: %s\n", str);

    printf("str2: %s\n", str2);

    int contatore = 0;
    while (str[contatore] != '\0') {
        contatore++;
    }
    printf("str length calculated manually: %d\n", contatore);
}
return 0;
```

Cosa vi aspettate come output?

- Input per str1: abcdefghijklm
- Input per str2: vvvvvvv

(abbiamo violato il constraint della lunghezza N)

Cosa

```
#include <stdio.h>
#include <string.h>
#define N 10

int main(int argc, char* argv[]) {
    char str[N];
    char str2[N];
    printf("inserisci str: ");
    scanf("%s", str);
    getchar(); // consumare \n
    printf("inserisci str2: ");
    scanf("%s", str2);

    printf("differenza tra str e str2: %d\n", strcmp(str, str2));
    printf("str lengths: str: %d, str2: %d\n", strlen(str), strlen(str2));

    printf("str differenza tra str e str2: -21\n");
    printf("str str lengths: str: 16, str2: 6\n");
    printf("str str1: abcdefghijvvvvvv\n");
    printf("str str2: vvvvvv\n");
    printf("str str length calculated manually: 16\n");

    int contatore = 0;
    while (str[contatore] != '\0') {
        contatore++;
    }
    printf("str length calculated manually: %d\n", contatore);
}
return 0;
```

Cosa vi aspettate come output?

- Input per str1: abcdefghijklm
- Input per str2: vvvvvvv

(abbiamo violato il constraint della lunghezza N)

Cosa

```
#include <stdio.h>
#include <string.h>
#define N 10
```

Al solito, allochiamo PRIMA in memoria str1, poi str2. Inserisco prima str1, poi str2, e str2 sovrascrive la parte di str1 che ha sforato

Cosa vi aspettate come output?

- Input per str1: abcdefghijklm
- Input per str2: vvvvvvvv
(abbiamo violato il constraint della lunghezza N)

```
printf("str lengths: str: %d, str2: %d\n", strlen(str), strlen(str2));

printf("str differenza tra str e str2: -21");
printf("str lengths: str: 16, str2: 6");
printf("str1: abcdefghijvvvvvv");
printf("str2: vvvvvvv");
printf("str length calculated manually: 16");

int contatore;
while (str[contatore] != '\0')
    contatore++;
printf("str length calculated manually: %d\n", contatore);
}
return 0;
```

Esercizi parte 2 di 2

Stringhe

Esercizio 06

Concatenazione di Stringhe

Testo

Scrivere un programma che permetta all'utente di prendere in input due stringhe. Il programma concatena le due stringhe e stampa il risultato a schermo.

Esempio

Input1: Luca

Input 2: Magri

Output: LucaMagri

Esercizio 07

Stringhe Palindrome

Testo

Scrivere un programma che permetta all'utente di inserire una stringa (assumere venga inserita senza spazi). Il programma verifica se la stringa è palindroma oppure no.

Esempio

`itopinonavevanonipoti -> OK`

`ciao -> NO`

Esercizio 08

Split di Stringhe

Testo

Scrivere un programma che consenta di inserire una serie di caratteri (senza spazio). Il programma deve stampare 3 sequenze:

1. cifre pari inserite dall'utente (nell'ordine in cui sono state inserite)
2. cifre dispari inserite dall'utente (nell'ordine in cui sono state inserite)
3. tutti gli altri caratteri meno che cifre pari e dispari (nell'ordine in cui sono stati inseriti)

Esempio

```

                                6
abc1h3j567k1  -> 1357
                                abchjkl
```