



POLITECNICO
MILANO 1863

Esercitazione 4

Array e Stringhe

14.10.2024 - Alessandro Montenegro

Array

Esercizi

Inversione di un Array

Testo

Scrivere un programma che permetta all'utente di inserire un array di numeri **v1**. Il programma copia l'array **v1** al contrario nell'array **v2**. Il programma stampa **v2**.

Esempio

[1, 2, 3, 4, 5] -> [5, 4, 3, 2, 1]

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

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

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

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

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

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

Stringhe

Funzioni Utili

Stringhe

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)

Stringhe


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);
```

Stringhe

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++;
}
```

Stringhe

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**

Stringhe

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);
```

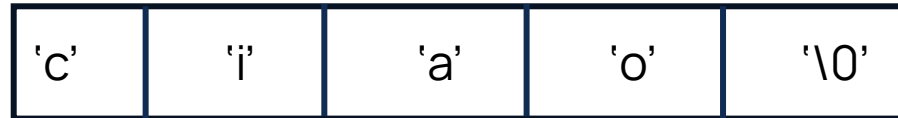
- **Consuma** caratteri da stdin fino a quando non trova uno '\n' (che viene consumato ma non inserito nella stringa)
- Acquisisce spazi bianchi
- Può causare **buffer overflow**
- Per il momento definire MAX_LEN **abbastanza grande**

Stringhe: Lunghezza

```
#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

stringa



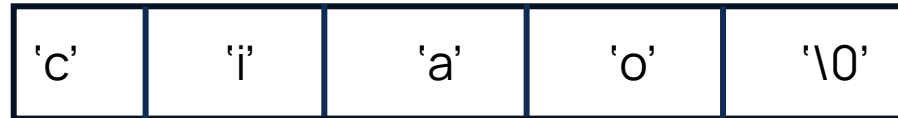
- `strlen(stringa)` restituisce 4
- `stringa[4]` corrisponde a `'\0'`

Stringhe: Lunghezza

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

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

stringa



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

Stringhe: Copia

```
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

Stringhe: Copia

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

stringa2 copiata dentro
stringa1

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

Stringhe: Concatenazione

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



`stringa2` concatenata a
`stringa1`, nella stessa
`stringa1`

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



`"Ciao"` concatenato 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
```

Stringhe: Concatenazione

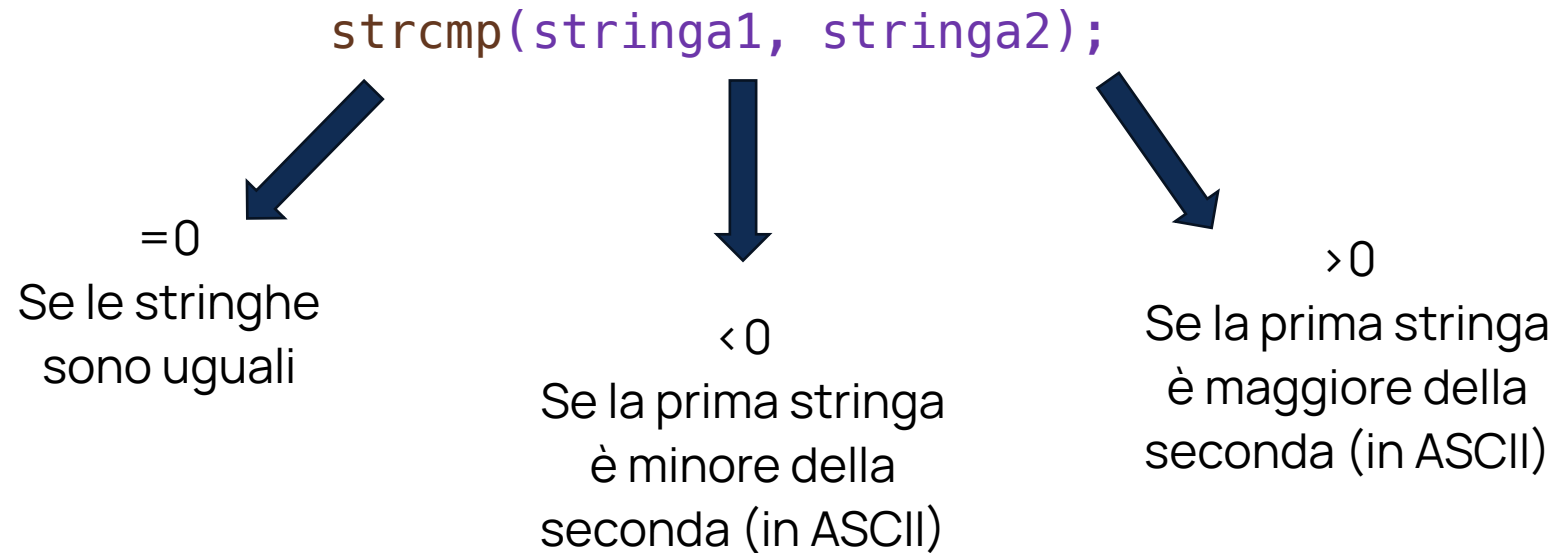
```
strcat(stringa1, stringa2);
```

`stringa2` concatenata a `stringa1`, nella stessa `stringa1`



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

Stringhe: Confronto



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

Stringhe: Confronto

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

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

Stringhe

Esercizi

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: Giacomo

Input 2: Boracchi

Output: GiacomoBoracchi

Stringhe Palindrome

Testo

Scrivere un programma che permetta all'utente di inserire una stringa. Il programma verifica se la stringa è palindroma oppure no.

Esempio

itopinonavevanonipoti - > OK

ciao - > NO

Split di Stringa

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

```
abc1h3j567kl -> 6  
                  1357  
                  abchjkl
```

Anagrammi

Testo

Si scriva un programma che permetta all'utente di inserire due stringhe. Il programma verifica se le stringhe sono anagrammi.

Esempio

«listen» e «silent» -> OK

«hello» e «world» -> NO



POLITECNICO
MILANO 1863

Contatti

Alessandro Montenegro

Email: alessandro.montenegro@polimi.it

Sito: <https://montenegroalessandro.github.io/InfoA2425/index.html>