Ambienti di Programmazione per il Software di Base

- Le Stringhe in C
- Input dalla linea di comando in C
- Conversione dei dati in C

Le Stringhe in C : Definizioni

- Stringa:

una sequenza di caratteri memorizzati in locazioni contigue di memoria e terminata dal carattere '\0', detto "terminatore di stringa"

's'	x
't'	x
'r'	x
'i'	x
'n'	x
'g'	x
'a'	x
'\0'	x

Le Stringhe in C : Ingresso/uscita di stringhe

- Creazione:

```
char variabile[10];
char variabile[] = “valore_stringa”;
```

- Acquisizione:

```
scanf(“%s”, address);
```

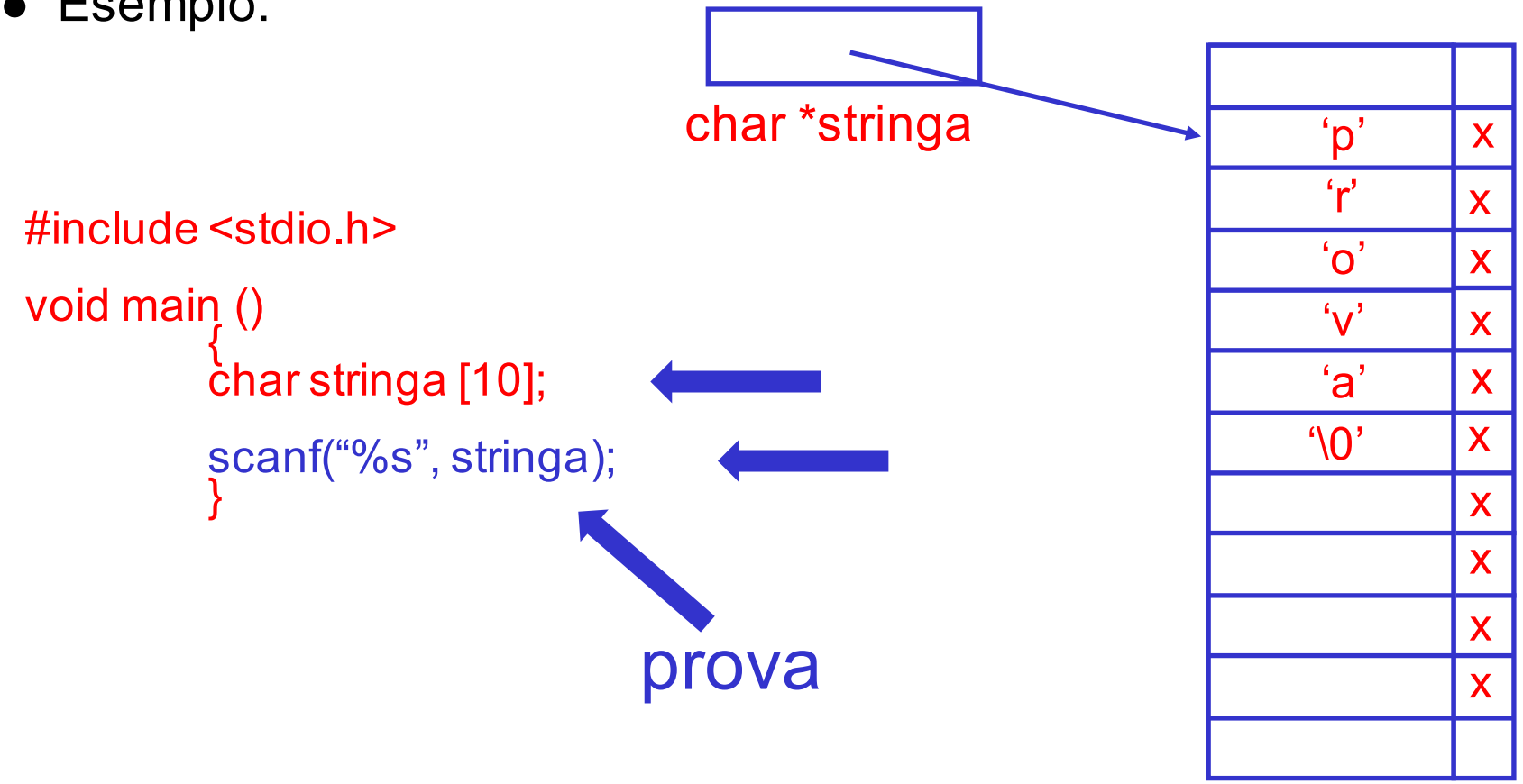
con address espressione di tipo char *

- Modifiche allo stato della memoria:

1. memorizza i caratteri acquisiti da tastiera a partire dalla locazione di indirizzo address, fino al primo di un insieme di caratteri delimitatori (‘\n’, blank, …) escluso
2. aggiunge in coda ai caratteri memorizzati il terminatore di stringa

Le Stringhe in C : Ingresso/uscita di stringhe

- Esempio:



Le Stringhe in C : Ingresso/uscita di stringhe

- Visualizzazione:

  printf("%s", address);

  con address espressione di tipo char *

- Effetto:

  visualizza i caratteri in memoria a partire dalla locazione di indirizzo address, fino al primo terminatore di stringa

- Esempio:

```
#include <stdio.h>

void main ()
        {
        char stringa [10];

        scanf("%s", stringa);
        printf("%s", stringa);
        };
```

poi → scanf("%s", stringa); → poi

Le Stringhe in C : Ingresso/uscita di stringhe

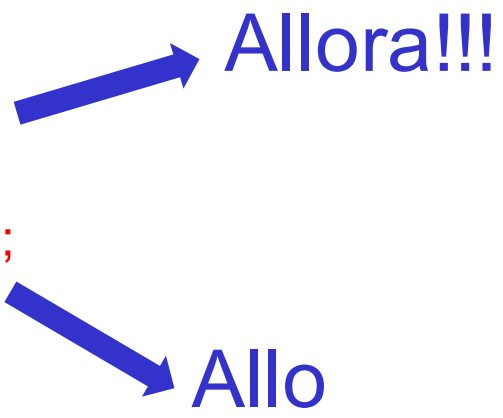
- Costante di tipo stringa:

  sequenza di caratteri racchiusa tra doppi apici

- Esempio:

```
#include <stdio.h>

void main ()
        {
        printf("%s\n", "Allora!!!");
        printf("%s\n", "Allo\0ra!!!");
        }
```



Le Stringhe in C : strlen()

- Attenzione:

  per poter utilizzare le funzioni seguenti è necessario includere la direttiva per il preprocessore #include <string.h>

- La funzione strlen():

  int strlen (char *s)

- Valore restituito:

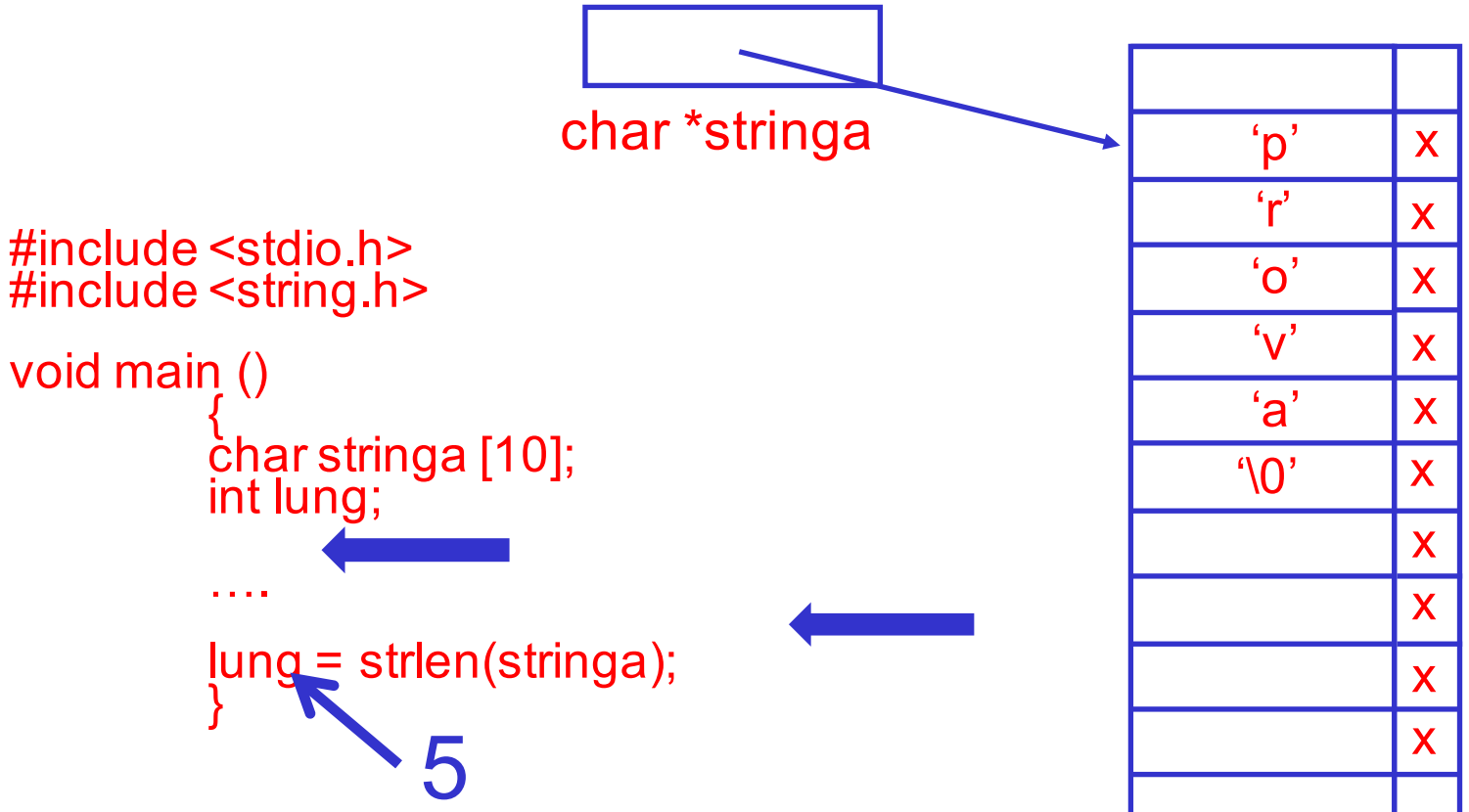
  il numero di caratteri memorizzati a partire da s fino al primo delimitatore di fine stringa escluso

- Molto informalmente:

  restituisce la lunghezza di s

Le Stringhe in C : strlen()

- Esempio:



Le Stringhe in C : strcpy()

- La funzione strcpy():

  char *strcpy (char *s1, char *s2)

- Modifiche allo stato della memoria:

  copia i caratteri memorizzati a partire da s2, fino al primo terminatore di stringa incluso, a partire da s1

- Valore restituito:

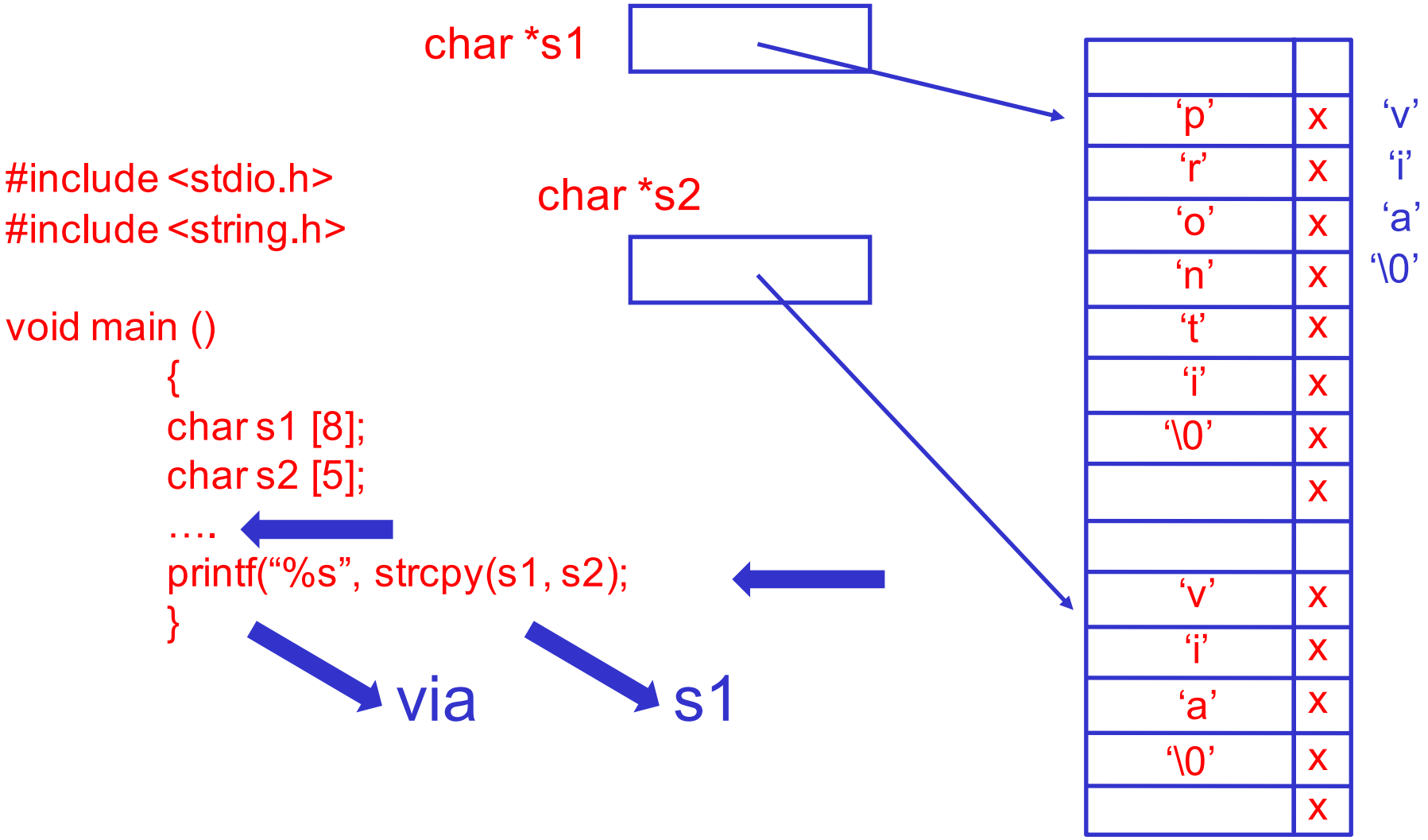
  s1

- Molto informalmente:

  sostituisce la stringa s1 con la stringa s2

Le Stringhe in C : strcpy()

- Esempio:



Le Stringhe in C : strcat()

- La funzione strcat():

  char *strcat(char *s1, char *s2)

- Descrizione:

  copia i caratteri memorizzati a partire da s2, fino al primo carattere di fine stringa incluso, a partire dall'indirizzo del primo carattere di fine stringa successivo a s1

- Valore restituito:

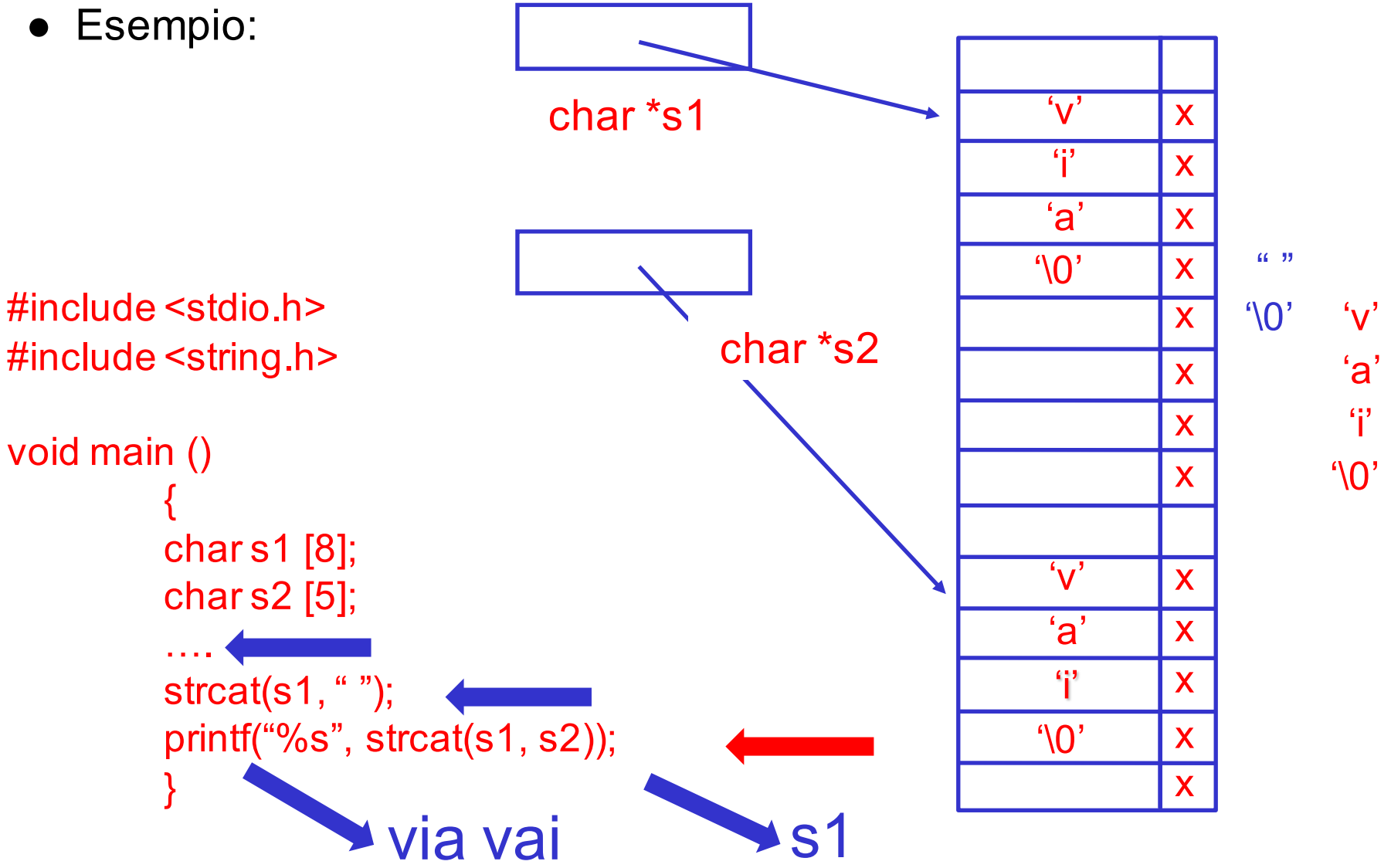
  s1

- Molto informalmente:

  concatena la stringa s2 alla stringa s1

Le Stringhe in C : strcat()

- Esempio:



Le Stringhe in C : strcmp()

- La funzione strcmp():

  int strcmp(char *s1, char *s2);

- Valore restituito:

  confronta i caratteri memorizzati a partire da s1 con quelli memorizzati a partire da s2 fino a incontrare la prima coppia di caratteri differenti o il primo delimitatore di stringa

  Restituisce:

0	se ha raggiunto una coppia di caratteri di fine stringa
un valore positivo	se il carattere raggiunto a partire da s1 precede lessicograficamente quello raggiunto a partire da s2
un valore negativo	altrimenti

Le Stringhe in C : strcmp()

- O più informalmente:

0	se le due stringhe sono uguali
un valore positivo	se la stringa s1 precede lessicograficamente la stringa s2
un valore negativo	se la stringa s2 precede lessicograficamente la stringa s1

Input dalla linea di comando in C

Il C permette di leggere argomenti dalla linea di comando, e questi possono poi essere utilizzati all'interno dei programmi. In fase di lancio del programma, possiamo scrivere gli argomenti dopo il nome del programma da eseguire.
Al fine di essere in grado di utilizzare tali argomenti, e' necessario definirli nella funzione main nel seguente modo:

```
main(int argc, char **argv)
```

- argc e' il numero degli argomenti digitati, incluso il nome del programma
- argv e' un array di stringhe contenente ciascuno un argomento, compreso il nome del programma come primo elemento.

Input dalla linea di comando in C

```
#include <stdio.h>

main(int argc, char **argv)

{ /* programma per stampare gli argomenti dalla linea di comando */

  int i;

  printf("argc=%d\n",argc);

  for(i=0;i < argc;++i)

    printf("argv[%d]:=%s\n",i,argv[i]);

}
```

Input dalla linea di comando in C

Se si e' compilato, chiamandolo args e fatto eseguire scrivendo:

```
args f1 "f2" f3 4 stop!
```

l'output sara':

```
argc=6

argv[0]=args

argv[1]=f1

argv[2]=f2

argv[3]=f3

argv[4]=4

argv[5]=stop!
```

Input dalla linea di comando in C

Note:

- argv[0] e' il nome del programma;
- argc totalizza anche il nome del programma;
- tra gli argomenti, i caratteri "" vengono ignorati (sono considerati solamente delimitatori di argomenti);
- gli spazi bianchi delimitano gli argomenti;
- nel caso in cui sia necessario mantenere spazi bianchi, occorre metterli tra "".

Conversione dei dati in C

Per convertire una stringa in un numero, la via più semplice è usare le funzioni di libreria (del C) atoi e atof :

• int atoi(const char *str); dove l'argomento è una stringa contenente la rappresentazione decimale di un numero intero, esegue la conversione di argomento e restituisce un valore di tipo int .

• double atof(const char *str); dove l'argomento è una stringa contenente la rappresentazione decimale di un numero floating (in notazione normale o esponenziale), esegue la conversione di argomento e restituisce un valore di tipo double

Entrambe le funzioni vanno utilizzate includendo l'*header-file*: <stdlib.h>

Conversione dei dati in C

```
/* Converte una stringa in un intero */

#include <stdio.h>

#include <stdlib.h>

char stringNum[] = "1234";

main()

{

  int intNum;

  intNum = atoi( stringNum );

  printf("intNum = %d\n", intNum );

}
```

Conversione dei dati in C

Per convertire numeri in stringhe, è più conveniente (rispetto ad altre possibilità) usare la funzione di libreria (del C) sprintf. Infatti questa funzione, non solo esegue la conversione, ma permette anche di ottenere una stringa *formattata* nel modo desiderato.
La funzione sprintf é identica alla printf salvo il fatto che scrive in una stringa anziché su stdout.

- Il primo argomento é la variabile stringa (definita come array di tipo char) in cui inserire i dati *formattati.*
- Il secondo argomento é la *control-string* (come il primo della printf).
- Il terzo argomento e i successivi sono i dati da *formattare* (come il secondo e i successivi della printf).

Conversione dei dati in C

```
/* sprintf example */
#include <stdio.h>
int main ()
{
  char buffer [50];
  int n, a=5, b=3;
  n = sprintf (buffer, "%d plus %d is %d", a, b, a+b);
  printf ("[%s] is a %d char long string\n",buffer,n);
  return 0;
}
```

```
Output:
[5 plus 3 is 8] is a 13 char long string
```

Esercizi proposti

1) Scrivere un programma C che prenda come parametri da linea di comando due interi e ne stampi a schermo il prodotto.
2) Scrivere un programma C che richieda di inserire parole da tastiera fino a che non venga digitata la stringa “EOF” e stampi a schermo il numero di volte che compare nel testo il carattere ‘a’
3) Scrivere un programma C che, date due stringhe, verifica se la prima è sub-stringa della seconda.