



Corso di Laurea in Informatica (*Track B*) - A.A. 2018/2019

Laboratorio di Informatica

Linguaggio C

(Stringhe e Caratteri)

docente: Veronica Rossano

veronica.rossano@uniba.it

Stringhe

20/03/19 Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

2

Stringhe

- «Stringa» è il nome che identifica un insieme di caratteri chiusi tra virgolette
 - `char string1[] = "first";`

Stringhe

- «Stringa» è il nome che identifica un insieme di caratteri chiusi tra virgolette
 - `char string1[] = "first";`
 - Equivalente a un array di caratteri
 - `char string1[] = {'f','i','r','s','t','\0'};`
 - Ha sei elementi (Stringhe di N caratteri hanno N+1 elementi), memorizzati in locazioni contigue
 - Il carattere '\0' termina le stringhe ed è detto terminatore.
 - Nell'inizializzazione delle stringhe è inserito in automatico, altrimenti deve essere inserito esplicitamente

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

3

20/03/19 Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

4

Stringhe

- Trattandosi di array, è possibile accedere ai caratteri individuali

`string1[3]` è il carattere 's'

- Nel caso degli array di stringhe, & non è richiesto nella scanf

- `scanf("%s", string2);`
- Perchè?

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

5

Stringhe

- Trattandosi di array, è possibile accedere ai caratteri individuali

`string1[3]` è il carattere 's'

- Nel caso degli array di stringhe, & non è richiesto nella scanf

- `scanf("%s", string2);`
- Perchè?
- Il nome dell'array è un puntatore all'indirizzo del primo elemento

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

6

Stringhe - Esempio

```

1 #include "stdio.h"
2 #define LENGTH 10
3 int main(void) {
4     char string[LENGTH];
5     printf("Inserisci Stringa: ");
6     scanf("%s", string);
7     printf("Stringa letta: ");
8     puts(string);
9 }
```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

7

Stringhe - Esempio

```

1 #include "stdio.h"
2 #define LENGTH 10
3 int main(void) {
4     char string[LENGTH];
5     printf("Inserisci Stringa: ");
6     scanf("%s", string);
7     printf("Stringa letta: ");
8     puts(string);
9 }
```

Dichiaro una stringa di
dimensione 10

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

8

Stringhe - Esempio

```

1 #include <stdio.h>
2 #define LENGTH 10
3 int main(void) {
4     char string[LENGTH];
5     printf("Inserisci Stringa: ");
6     scanf("%s", string);
7     printf("Stringa letta: ");
8     puts(string);
9 }
```

**Leggo l'input
Importante: %s non %c**

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

9

Stringhe - Esempio

```

1 #include <stdio.h>
2 #define LENGTH 10
3 int main(void) {
4     char string[LENGTH];
5     printf("Inserisci Stringa: ");
6     scanf("%s", string);
7     printf("Stringa letta: ");
8     puts(string);
9 }
```

**Domanda
Cosa stampa se inserisco una stringa la cui dimensione è maggiore a 10?**

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

10

Stringhe - Esempio

```

1 #include <stdio.h>
2 #define LENGTH 10
3 int main(void) {
4     char string[LENGTH];
5     printf("Inserisci Stringa: ");
6     scanf("%s", string);
7     printf("Stringa letta: ");
8     puts(string);
9 }
```

**Domanda
Cosa stampa se inserisco una stringa la cui dimensione è maggiore a 10?**

```

gcc version 4.6.3
:|
Inserisci Stringa: stringamoltolunga
Stringa letta: stringamoltolunga
:|
```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

11

Stringhe - Esempio

- Il Linguaggio C non effettua nessun controllo sulla lunghezza delle stringhe
 - Il realtà non effettua nessun controllo sugli array!
- Stringhe anche più lunghe di LENGTH saranno correttamente memorizzate
- Non abbiamo però la garanzia che quelle locazioni di memoria non vengano occupate in futuro
 - Possibili comportamenti inattesi!
- **Accorgimento:** limitare il numero di caratteri letti in input, sempre!

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

12

Stringhe

```

1 #include "stdio.h"
2 #define LENGTH 10
3 int main(void) {
4     char string[LENGTH];
5     printf("Inserisci Stringa: ");
6     scanf("%9s", string);
7     printf("Stringa letta: ");
8     puts(string);
9 }
```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

13

Stringhe

```

1 #include "stdio.h"
2 #define LENGTH 10
3 int main(void) {
4     char string[LENGTH];
5     printf("Inserisci Stringa: ");
6     → scanf("%9s", string);
7     printf("Stringa letta: ");
8     puts(string);
9 }
```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

14

Limite la lettura ai primi N-1 caratteri (l'ultimo è il terminatore)

Stringhe

```

1 #include "stdio.h"
2 #define LENGTH 10
3 int main(void) {
4     char string[LENGTH];
5     printf("Inserisci Stringa: ");
6     scanf("%9s", string);
7     printf("Inserisci Stringa: ");
8     puts(string);
9 }
```

Limite la lettura ai primi N-1 caratteri (l'ultimo è il terminatore)

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

15

Stringhe

```

1 #include "stdio.h"
2 #define LENGTH 10
3 int main(void) {
4     char string[LENGTH];
5     printf("Inserisci Stringa: ");
6     → scanf("%9s", string);
7     printf("Stringa letta: ");
8     puts(string);
9 }
```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

16

Importante: la scanf() si ferma se trova un terminatore o uno spazio

Stringhe

```

1 #include "stdio.h"
2 #define LENGTH 10
3 int main(void) {
4     char string[LENGTH];
5     printf("Inserisci Stringa: ");
6     scanf("%9s", string);
7     printf("Stringa: %s", string);
8     puts(string);
9 }
```

Importante: la scanf() si ferma se trova un terminatore o uno spazio

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

17

Stringhe – Funzioni di Elaborazione

- C include numerose funzioni per l'elaborazione di stringhe e caratteri

- Disponibili nelle librerie `<ctype.h>` e `<string.h>`
- Funzioni per l'elaborazione dei caratteri
 - Controllo che un dato carattere sia un numero o una lettera
 - Controllo che un dato carattere sia maiuscolo o minuscolo
 - Controllo che un dato carattere sia un simbolo di punteggiatura, uno spazio, etc.

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

18

Stringhe – Funzioni di Elaborazione

- C include numerose funzioni per l'elaborazione di stringhe e caratteri

- Disponibili nella libreria `<ctype.h>` e `<string.h>`
- Funzioni per l'elaborazione dei caratteri
 - Controllo che un dato carattere sia un numero o una lettera
 - Controllo che un dato carattere sia maiuscolo o minuscolo
 - Controllo che un dato carattere sia un simbolo di punteggiatura, uno spazio, etc.

- Funzioni per l'elaborazione delle stringhe

- Confronto tra due stringhe (verifica se sono uguali o meno)
- Conversione stringhe → valori numerici (e viceversa)
- Copia (parziale o totale) di una stringa in un'altra stringa o concatenazione tra stringhe
- Suddivisione di una frase in singoli termini (*tokenizzazione*)

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

19

Funzioni per l'elaborazione dei caratteri

Prototipo	Descrizione
<code>int isdigit(int c);</code>	Returns true if c is a digit and false otherwise. { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 }.
<code>int isalpha(int c);</code>	Returns true if c is a letter and false otherwise. { a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z }
<code>int isalnum(int c);</code>	Returns true if c is a digit or a letter and false otherwise. { a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z , 1, 2, 3, 4, 5, 6, 7, 8, 9 }
<code>int isxdigit(int c);</code>	Returns true if c is a hexadecimal digit character and false otherwise. { 0 1 2 3 4 5 6 7 8 9 A B C D E F a b c d e f }

In C i valori booleani sono codificati come interi:
True = 1 False = 0

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

20

Funzioni per l'elaborazione dei caratteri

Prototipo	Descrizione
<code>int islower(int c);</code>	Returns true if <code>c</code> is a lowercase letter and false otherwise.
<code>int isupper(int c);</code>	Returns true if <code>c</code> is an uppercase letter; false otherwise.
<code>int tolower(int c);</code>	If <code>c</code> is an uppercase letter, <code>tolower</code> returns <code>c</code> as a lowercase letter. Otherwise, <code>tolower</code> returns the argument unchanged.
<code>int toupper(int c);</code>	If <code>c</code> is a lowercase letter, <code>toupper</code> returns <code>c</code> as an uppercase letter. Otherwise, <code>toupper</code> returns the argument unchanged.
<code>int isspace(int c);</code>	Returns true if <code>c</code> is a white-space character—newline ('\n'), space (' '), form feed ('\f'), carriage return ('\r'), horizontal tab ('\t'), or vertical tab ('\v')—and false otherwise.

20/03/19

In C i valori booleani sono codificati come interi:
True = 1 False = 0

21

Funzioni per l'elaborazione dei caratteri

Prototipo	Descrizione
<code>int iscntrl(int c);</code>	Returns true if <code>c</code> is a control character and false otherwise.
<code>int ispunct(int c);</code>	Returns true if <code>c</code> is a punctuation character. A punctuation character is any graphic character (as in <code>isgraph</code>) that is not alphanumeric (as in <code>isalnum</code>). False otherwise.
<code>int isprint(int c);</code>	Returns true value if <code>c</code> is a printing character including space (' ') and false otherwise.
<code>int isgraph(int c);</code>	Returns true if <code>c</code> is a printing character other than space (' ') and false otherwise.

20/03/19

In C i valori booleani sono codificati come interi:
True = 1 False = 0

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

22

Funzioni per l'elaborazione dei caratteri

```

1 #include <stdio.h>
2 #include <ctype.h>           isdigit(char) restituisce 1 se il carattere è un
3                                         numero, altrimenti sarà uguale a 0
4
5 int main() {
6     char a = 'a'; // IMPORTANTE
7     char n = '9'; // 'virgolette singole' per i char
8
9     // Funzione isdigit()
10    if(isdigit(a)) printf("\n%c is a digit", a);
11    else printf("\n%c is not a digit", a);
12
13    if(isdigit(n)) printf("\n%c is a digit", n);
14    else printf("\n%c is not a digit", n);
}

```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

23

Funzioni per l'elaborazione dei caratteri

```

1 #include <stdio.h>
2 #include <ctype.h>           isdigit(char) restituisce a 1 se il carattere è un
3                                         numero, altrimenti sarà uguale a 0
4
5 int main() {
6     char a = 'a'; // IMPORTANTE
7     char n = '9'; // 'virgolette singole' per i
8
9     // Funzione isdigit()
10    if(isdigit(a)) printf("\n%c is a digit", a);
11    else printf("\n%c is not a digit", a);
12
13    if(isdigit(n)) printf("\n%c is a digit", n);
14    else printf("\n%c is not a digit", n);
}

```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

24

gcc version 4.6.3
 ➤
 a is not a digit
 9 is a digit

Funzioni per l'elaborazione dei caratteri

```

1 // LABORATORIO DI INFORMATICA
2 // a.a. 2017/2018
3 // Corso di Laurea in informatica I.P.S.
4 // docente: Cataldo Musto
5
6 #include <stdio.h>
7 #include <cctype.h>
8
9+ int main() {
10    char a = 'a'; // IMPORTANTE
11    char up = 'A'; // virgolette 'singole'
12    char n = '1'; // per le variabili
13    char s = "#"; // char
14
15    // funzione 'isalpha()'
16    if(isalpha(a)) printf("\n%c is a digit", a);
17    else printf("\n%c is not a digit", a);
18
19    if(isalpha(n)) printf("\n%c is a digit", n);
20    else printf("\n%c is not a digit", n);
21 }

```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

25

isalpha(char) restituisce a 1 se il carattere è alfabetico, altrimenti sarà uguale a 0

Funzioni per l'elaborazione dei caratteri

```

1 // LABORATORIO DI INFORMATICA
2 // a.a. 2017/2018
3 // Corso di Laurea in informatica I.P.S.
4 // docente: Cataldo Musto
5
6 #include <stdio.h>
7 #include <cctype.h>
8
9+ int main() {
10    char a = 'a'; // IMPORTANTE
11    char up = 'A'; // virgolette 'singole'
12    char n = '1'; // per le variabili
13    char s = "#"; // char
14
15    // funzione 'isalpha()'
16    if(isalpha(a)) printf("\n%c is alphabetic", a);
17    else printf("\n%c is not alphabetic", a);
18
19    if(isalpha(n)) printf("\n%c is alphabetic", n);
20    else printf("\n%c is not alphabetic", n);
21 }

```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

26

isalpha(char) restituisce a 1 se il carattere è alfabetico, altrimenti sarà uguale a 0

gcc version 4.6.3
 ►
 a is alphabetic
 1 is not alphabetic ►

Funzioni per l'elaborazione dei caratteri

```

1 // LABORATORIO DI INFORMATICA
2 // a.a. 2017/2018
3 // Corso di Laurea in informatica I.P.S.
4 // docente: Cataldo Musto
5
6 #include <stdio.h>
7 #include <cctype.h>
8
9+ int main() {
10    char a = 'a'; // IMPORTANTE
11    char up = 'A'; // virgolette 'singole'
12    char n = '1'; // per le variabili
13    char s = "#"; // char
14
15    // funzione 'isalnum()'
16    if(isalnum(a)) printf("\n%c is alpha-numeric", a);
17    else printf("\n%c is not alpha-numeric", a);
18
19    if(isalnum(n)) printf("\n%c is alpha-numeric", n);
20    else printf("\n%c is not alpha-numeric", n);
21
22    if(isalnum(s)) printf("\n%c is alpha-numeric", s);
23    else printf("\n%c is not alpha-numeric", s);
24 }

```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

27

isalnum(char) restituisce a 1 se il carattere è alfanumerico, altrimenti sarà uguale a 0

Funzioni per l'elaborazione dei caratteri

```

1 // LABORATORIO DI INFORMATICA
2 // a.a. 2017/2018
3 // Corso di Laurea in Informatica T.P.S.
4 // docente: Cataldo Musto
5
6 #include <stdio.h>
7 #include <cctype.h>
8
9+ int main() {
10    char a = 'a'; // IMPORTANTE
11    char up = 'A'; // virgolette 'singole'
12    char n = '1'; // per le variabili
13    char s = "#"; // char
14
15    // funzione 'isalnum()'
16    if(isalnum(a)) printf("\n%c is alpha-numeric", a);
17    else printf("\n%c is not alpha-numeric", a);
18
19    if(isalnum(n)) printf("\n%c is alpha-numeric", n);
20    else printf("\n%c is not alpha-numeric", n);
21
22    if(isalnum(s)) printf("\n%c is alpha-numeric", s);
23    else printf("\n%c is not alpha-numeric", s);
24 }

```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

28

isalnum(char) restituisce a 1 se il carattere è alfanumerico, altrimenti sarà uguale a 0.

gcc version 4.6.3
 ►
 a is alpha-numeric
 1 is alpha-numeric
 # is not alpha-numeric ►

Funzioni per l'elaborazione dei caratteri

```

1 // LABORATORIO DI INFORMATICA
2 // a.a. 2017/2018
3 // Corso di Laurea in Informatica T.P.S. isupper(char) restituisce a 1 se il carattere è
4 // docente: Cataldo Musto maiuscolo, altrimenti sarà uguale a 0.

5
6 #include <stdio.h>
7 #include <ctype.h>

8
9 int main() {
10     char a = 'a'; // IMPORTANTE
11     char up = 'A'; // virgolette 'singole'
12     char n = '1'; // per le variabili
13     char s = '#'; // char
14
15
16     if(islower(a)) printf("\n%c is lower-case", a);
17     else printf("\n%c is upper-case", a);
18
19     if(islower(up)) printf("\n%c is lower-case", up);
20     else printf("\n%c is upper-case", up);
21 }
```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

29

Funzioni per l'elaborazione dei caratteri

```

1 // LABORATORIO DI INFORMATICA
2 // a.a. 2017/2018
3 // Corso di Laurea in Informatica T.P.S.
4 // docente: Cataldo Musto
5
6 #include <stdio.h>
7 #include <ctype.h>
8
9 int main() {
10     char a = 'a'; // IMPORTANTE
11     char up = 'A'; // virgolette 'singole'
12     char n = '1'; // per le variabili
13     char s = '#'; // char
14
15     // funzione 'toupper()'
16     printf("\n%c to upper case: %c", a, toupper(a));
17 }
```

toupper(char) converte un carattere in maiuscolo.

gcc version 4.6.3
 >
 a to upper case: A

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

31

Funzioni per l'elaborazione dei caratteri

```

1 // LABORATORIO DI INFORMATICA
2 // a.a. 2017/2018
3 // Corso di Laurea in Informatica T.P.S. isupper(char) restituisce a 1 se il carattere è
4 // docente: Cataldo Musto maiuscolo, altrimenti sarà uguale a 0.

5
6 #include <stdio.h>
7 #include <ctype.h>
8
9 int main() {
10     char a = 'a'; // IMPORTANTE
11     char up = 'A'; // virgolette 'singole'
12     char n = '1'; // per le variabili
13     char s = '#'; // char
14
15
16     if(islower(a)) printf("\n%c is lower-case", a);
17     else printf("\n%c is upper-case", a);
18
19     if(islower(up)) printf("\n%c is lower-case", up);
20     else printf("\n%c is upper-case", up);
21 }
```

gcc version 4.6.3
 >
 a is lower-case
 A is not lower-case

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

30

Recap

Prototipo	Descrizione
<code>int isdigit(int c);</code>	Returns true if c is a digit and false otherwise.
<code>int isalpha(int c);</code>	Returns true if c is a letter and false otherwise.
<code>int isalnum(int c);</code>	Returns true if c is a digit or a letter and false otherwise.
<code>int isxdigit(int c);</code>	Returns true if c is a hexadecimal digit character and false otherwise.
<code>int islower(int c);</code>	Returns true if c is a lowercase letter and false otherwise.
<code>int isupper(int c);</code>	Returns true if c is an uppercase letter; false otherwise.
<code>int tolower(int c);</code>	If c is an uppercase letter, tolower returns c as a lowercase letter. Otherwise, tolower returns the argument unchanged.

In C i valori booleani sono codificati come interi:
 True = 1 False = 0

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

32

Recap (cont.)

Prototipo	Descrizione
<code>int toupper(int c);</code>	If <code>c</code> is a lowercase letter, <code>toupper</code> returns <code>c</code> as an uppercase letter. Otherwise, <code>toupper</code> returns the argument unchanged.
<code>int isspace(int c);</code>	Returns true if <code>c</code> is a white-space character—newline ('\n'), space (' '), form feed ('\f'), carriage return ('\r'), horizontal tab ('\t'), or vertical tab ('\v')—and false otherwise.
<code>int iscntrl(int c);</code>	Returns true if <code>c</code> is a control character and false otherwise.
<code>int ispunct(int c);</code>	Returns true if <code>c</code> is a printing character other than a space, a digit, or a letter and false otherwise.
<code>int isprint(int c);</code>	Returns true value if <code>c</code> is a printing character including space (' ') and false otherwise.
<code>int isgraph(int c);</code>	Returns true if <code>c</code> is a printing character other than space (' ') and false otherwise.

In C i valori booleani sono codificati come interi:

True = 1 False = 0

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

33

Problema 3.1

Scrivere un programma che acquisisca in input la password inserita da un utente. Verificare che la password contenga almeno una lettera maiuscola ed almeno un numero. Stampare in input un messaggio di conferma se la password è corretta. In alternativa, stampare un messaggio di errore.

Input?

Output?

Quale tipologia di istruzioni ci serve?

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

34

Problema 3.1

Scrivere un programma che acquisisca in input la password inserita da un utente. Verificare che la password contenga almeno una lettera maiuscola ed almeno un numero. Stampare in input un messaggio di conferma se la password è corretta. In alternativa, stampare un messaggio di errore.

Input?

Output?

Quale tipologia di istruzioni ci serve?

Esempio

Input: cat4ldo
Output: «Password non corretta, inserire almeno una lettera maiuscola»

Input: Password
Output: «Password non corretta, inserire almeno un numero»

Input: Pa55word
Output: «Password impostata correttamente.»

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

35

Problema 3.1

Scrivere un programma che acquisisca in input la password inserita da un utente. Verificare che la password contenga almeno una lettera maiuscola ed almeno un numero. Stampare in input un messaggio di conferma se la password è corretta. In alternativa, stampare un messaggio di errore.

Input?

Stringa, inserita dall'utente

Output?

Messaggio di conferma o messaggio di errore

Quale tipologia di istruzioni ci serve?

- Istruzioni per la manipolazione dei caratteri
- (che altro?)

Esempio

Input: cat4ldo
Output: «Password non corretta, inserire almeno una lettera maiuscola»

Input: Password
Output: «Password non corretta, inserire almeno un numero»

Input: Pa55word
Output: «Password impostata correttamente.»

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

36

Problema 3.1

Scrivere un programma che acquisisca in input la password inserita da un utente. Verificare che la password contenga almeno una lettera maiuscola ed almeno un numero.
Stampare in input un messaggio di conferma se la password è corretta. In alternativa, stampare un messaggio di errore.

Input?

Stringa, inserita dall'utente.

Output?

Messaggio di conferma o messaggio di errore.

Quale tipologia di istruzioni ci serve?

- Istruzioni per la manipolazione dei caratteri
- Istruzioni di iterazione
 - Suggerimento: una stringa è un array di caratteri

Esempio

Input: cat4ldo
Output: «Password non corretta, inserire almeno una lettera maiuscola»

Input: Password

Output: «Password non corretta, inserire almeno un numero»

Input: Pa55word

Output: «Password impostata correttamente.»

Codificare la soluzione su Repl.it

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

37

Soluzione 3.1 (parte 1)

```

1 #include <stdio.h>
2 #include <ctype.h> // includo la libreria per gestire le
3 // funzioni sui caratteri
4 #define PASSWORD_LENGTH 11 // lunghezza massima della password
5
6 int main() {
7     int correct = 0; /* variabile di controllo, diventa = 1 quando
8     è stata inserita una password corretta*/
9     char password[PASSWORD_LENGTH]; // vettore contenente la password
10
11    int upper = 0; // variabili di controllo del numero di maiuscole
12    int digit = 0; // e del numero di cifre numeriche
13
14    while ( correct == 0 ) {
15        //INSERIMENTO INPUT
16        printf("Inserisci la tua password (max. 10 caratteri): ");
17        scanf("%10s", password); // leggo esattamente nove caratteri
18
19        upper = 0; // NOTA: perchè inizializzo queste variabili a zero?
20        digit = 0;

```

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

38

Soluzione 3.1 (parte 1)

```

1 #include <stdio.h>
2 #include <ctype.h> // includo la libreria per gestire le
3 // funzioni sui caratteri
4 #define PASSWORD_LENGTH 11 // lunghezza massima della password
5
6 int main() {
7     int correct = 0; // variabile di controllo, diventa = 1 quando
8     è stata inserita una password corretta
9     char password[PASSWORD_LENGTH]; // vettore contenente la password
10
11    int upper = 0; // variabili di controllo del numero di maiuscole
12    int digit = 0; // e del numero di cifre numeriche
13
14    while ( correct == 0 ) {
15        //INSERIMENTO INPUT
16        printf("Inserisci la tua password (max. 10 caratteri): ");
17        scanf("%10s", password); // leggo esattamente nove caratteri
18
19        upper = 0; // NOTA: perchè inizializzo queste variabili a zero?
20        digit = 0;

```

Rigo 7 – dichiaro una variabile per uscire dal ciclo

39

Soluzione 3.1 (parte 1)

```

1 #include <stdio.h>
2 #include <ctype.h> // includo la libreria per gestire le
3 // funzioni sui caratteri
4 #define PASSWORD_LENGTH 11 // lunghezza massima della password
5
6 int main() {
7     int correct = 0; // variabile di controllo, diventa = 1 quando
8     è stata inserita una password corretta
9     char password[PASSWORD_LENGTH]; // vettore contenente la password
10
11    int upper = 0; // variabili di controllo del numero di maiuscole
12    int digit = 0; // e del numero di cifre numeriche
13
14    while ( correct == 0 ) {
15        //INSERIMENTO INPUT
16        printf("Inserisci la tua password (max. 10 caratteri): ");
17        scanf("%10s", password); // leggo esattamente nove caratteri
18
19        upper = 0; // NOTA: perchè inizializzo queste variabili a zero?
20        digit = 0;

```

Rigo 7 – dichiaro una variabile per uscire dal ciclo

Rigo 14 – il programma viene eseguito finché la variabile **sentinella** è **uguale a zero**

40

Soluzione 3.1 (parte 1)

```

1 #include <stdio.h>
2 #include <ctype.h> // includo la libreria per gestire le
3             // funzioni sui caratteri
4 #define PASSWORD_LENGTH 11 // lunghezza massima della password
5
6 int main() {
7     int correct = 0; // variabile di controllo, diventa = 1 quando
8             // è stata inserita una password corretta
9     char password[PASSWORD_LENGTH]; // vettore contenente la password
10
11    int upper = 0; // variabili di controllo del numero di maiuscole
12    int digit = 0; // e del numero di cifre numeriche
13
14    while ( correct == 0 ) {
15        //INSEGNAMENTO INPUT
16        printf("Inserisci la tua password (max. 10 caratteri): ");
17        scanf("%10s", password); // leggo esattamente nove caratteri
18
19        upper = 0; // NOTA: perchè inizializzo queste variabili a zero?
20        digit = 0;

```

Rigo 7 – dichiaro una variabile per uscire dal ciclo

Rigo 14 – il programma viene eseguito finché la variabile **sentinella** è uguale a zero

Rigo 19-20 – perchè inizializzo nuovamente queste variabili?

20/03/19 Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

41

Soluzione 3.1 (parte 2)

```

21 // ELABORAZIONE INPUT
22 for ( unsigned i=0; i<PASSWORD_LENGTH ; i++ ) {
23     digit += isdigit(password[i]);
24     upper += isupper(password[i]);
25
26     if ((digit > 0) && (upper > 0))
27         i = PASSWORD_LENGTH; // a che serve?
28 }
29
30 // VISUALIZZAZIONE OUTPUT
31 if ( (digit>0) && (upper>0) ) {
32     puts("Password impostata correttamente");
33     correct++;
34 }
35 else if ( digit == 0 )
36     puts("Inserire almeno un carattere numerico");
37     if (upper == 0)
38         puts("Inserire almeno un carattere maiuscolo");
39 }

```

20/03/19 Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

42

Soluzione 3.1 (parte 2)

```

21 // ELABORAZIONE INPUT
22 for ( unsigned i=0; i<PASSWORD_LENGTH ; i++ ) {
23     digit += isdigit(password[i]);
24     upper += isupper(password[i]);
25
26     if ((digit > 0) && (upper > 0))
27         i = PASSWORD_LENGTH; // a che serve?
28 }
29
30 // VISUALIZZAZIONE OUTPUT
31 if ( (digit>0) && (upper>0) ) {
32     puts("Password impostata correttamente");
33     correct++;
34 }
35 else if ( digit == 0 )
36     puts("Inserire almeno un carattere numerico");
37     if (upper == 0)
38         puts("Inserire almeno un carattere maiuscolo");
39 }

```

Riga 22-28
scorro tutti i caratteri della password, conteggiando il numero di caratteri maiuscoli e di cifre numeriche

20/03/19 Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

43

Soluzione 3.1 (parte 2)

```

21 // ELABORAZIONE INPUT
22 for ( unsigned i=0; i<PASSWORD_LENGTH ; i++ ) {
23     digit += isdigit(password[i]);
24     upper += isupper(password[i]);
25
26     if ((digit > 0) && (upper > 0))
27         i = PASSWORD_LENGTH; // a che serve?
28 }
29
30 // VISUALIZZAZIONE OUTPUT
31 if ( (digit>0) && (upper>0) ) {
32     puts("Password impostata correttamente");
33     correct++;
34 }
35 else if ( digit == 0 )
36     puts("Inserire almeno un carattere numerico");
37     if (upper == 0)
38         puts("Inserire almeno un carattere maiuscolo");
39 }

```

Riga 22-28
scorro tutti i caratteri della password, conteggiando il numero di caratteri maiuscoli e di cifre numeriche

Riga 26-27
Non ho bisogno di eseguire tutti i cicli! E' sufficiente farlo finché si trova un carattere e una maiuscola. In questo modo si riducono gli accessi in memoria e il programma è più efficiente.

20/03/19 Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

44

Soluzione 3.1 (parte 2)

```

21 // ELABORAZIONE INPUT
22 for ( unsigned i=0; i<PASSWORD_LENGTH ; i++) {
23     digit += isdigit(password[i]);
24     upper += isupper(password[i]);
25
26     if ((digit > 0) && (upper > 0))
27         i = PASSWORD_LENGTH; // a che serve?
28 }
29
30 // VISUALIZZAZIONE OUTPUT
31 if ( (digit>0) && (upper>0) ) {
32     puts("Password impostata correttamente");
33     correct++;
34 }
35 else if ( digit == 0)
36     puts("Inserire almeno un carattere numerico");
37     if (upper == 0)
38         puts("Inserire almeno un carattere maiuscolo");
39 }

```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

45

Riga 30-38

Stampa dell'output.
Stampa un messaggio di avvenuta impostazione se l'utente ha inserito almeno un numero e almeno una maiuscola, altrimenti stampa un messaggio di errore.

Soluzione 3.1 (parte 2)

```

21 // ELABORAZIONE INPUT
22 for ( unsigned i=0; i<PASSWORD_LENGTH ; i++) {
23     digit += isdigit(password[i]);
24     upper += isupper(password[i]);
25
26     if ((digit > 0) && (upper > 0))
27         i = PASSWORD_LENGTH; // a che serve?
28 }
29
30 // VISUALIZZAZIONE OUTPUT
31 if ( (digit>0) && (upper>0) ) {
32     puts("Password impostata correttamente");
33     correct++;
34 }
35 else if ( digit == 0)
36     puts("Inserire almeno un carattere numerico");
37     if (upper == 0)
38         puts("Inserire almeno un carattere maiuscolo");
39 }

```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

46

Riga 30-38

Stampa dell'output.
Stampa un messaggio di avvenuta impostazione se l'utente ha inserito almeno un numero e almeno una maiuscola, altrimenti stampa un messaggio di errore.

Torniamo alla domanda precedente.

Soluzione 3.1 (parte 1)

```

1 #include <stdio.h>
2 #include <ctype.h> // includo la libreria per gestire le
3         funzioni sui caratteri
4 #define PASSWORD_LENGTH 10 // lunghezza massima della password
5
6 int main() {
7     int correct = 0; // variabile di controllo, diventa = 1 quando
8         è stata inserita una password corretta
9     char password[PASSWORD_LENGTH]; // vettore contenente la password
10
11    int upper = 0; // variabili di controllo del numero di maiuscole
12    int digit = 0; // e del numero di cifre numeriche
13
14    while ( correct == 0 ) {
15        //INSERIMENTO INPUT
16        printf("Inserisci la tua password (max. 10 caratteri): ");
17        scanf("%9s", &password); // leggo esattamente nove caratteri
18
19        upper = 0; // NOTA: perché inizializzo queste variabili a zero?
20        digit = 0;

```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

47

Rigo 7 – dichiaro una variabile per uscire dal ciclo

Rigo 14 – il programma viene eseguito finché la variabile **sentinella** è uguale a zero

Rigo 19-20 – perché inizializzo nuovamente queste variabili?

Soluzione 3.1 (parte 1)

```

1 #include <stdio.h>
2 #include <ctype.h> // includo la libreria per gestire le
3         funzioni sui caratteri
4 #define PASSWORD_LENGTH 10 // lunghezza massima della password
5
6 int main() {
7     int correct = 0; // variabile di controllo, diventa = 1 quando
8         è stata inserita una password corretta
9     char password[PASSWORD_LENGTH]; // vettore contenente la password
10
11    int upper = 0; // variabili di controllo del numero di maiuscole
12    int digit = 0; // e del numero di cifre numeriche
13
14    while ( correct == 0 ) {
15        //INSERIMENTO INPUT
16        printf("Inserisci la tua password (max. 10 caratteri): ");
17        scanf("%9s", &password); // leggo esattamente nove caratteri
18
19        upper = 0; // NOTA: perché inizializzo queste variabili a zero?
20        digit = 0;

```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

48

Rigo 19-20 – perché inizializzo nuovamente queste variabili?

Commentare le righe 19 e 20 ed eseguire il programma inserendo queste password:

- password
- passwOrd
- Password

La terza password viene accettata, nonostante non sia corretta

Soluzione 3.1 (parte 1)

```

1 #include <stdio.h>
2 #include <ctype.h> // includo la libreria per gestire le
3             // funzioni sui caratteri
4 #define PASSWORD_LENGTH 10 // lunghezza massima della password
5
6 int main() {
7     int correct = 0; // variabile di controllo, diventa = 1 quando
8             // è stata inserita una password corretta
9     char password[PASSWORD_LENGTH]; // vettore contenente la password
10
11    int upper = 0; // variabili di controllo del numero di maiuscole
12    int digit = 0; // e del numero di cifre numeriche
13
14    while ( correct == 0 ) {
15        // INSERIMENTO INPUT
16        printf("Inserisci la tua password (max. 10 caratteri): ");
17        scanf("%9s", &password); // leggo esattamente nove caratteri
18
19        upper = 0; // NOTA: perché inizializzo queste variabili a zero
20        digit = 0;

```

Se non inizializziamo le variabili ad ogni ciclo,
rimarrebbero memorizzati i valori dei cicli precedenti,
e password non corrette potrebbero essere accettate

**Rigo 19-20 – perché
inizializzo nuovamente
queste variabili?**

Commentare le righe 19
e 20 ed eseguire il
programma inserendo
queste password:

- password
- passwOrd
- Password

La terza password viene
accettata, nonostante
non sia corretta

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

49

Stringhe – Funzioni di Elaborazione

• Confrontare le stringhe

- Il computer confronta i codici numerici ASCII dei caratteri delle stringhe


```
int strcmp( const char *s1, const char *s2 );
```
- Confronta la stringa s1 con s2
 - Restituisce zero se sono uguali, un numero negativo se s1 < s2, o un numero positivo se s1 > s2 (es. Roma > Bari, Albero < Bari)

20/03/19 Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

50

Stringhe – Funzioni di Elaborazione

• Confrontare le stringhe

- Il computer confronta i codici numerici ASCII dei caratteri delle stringhe


```
int strcmp( const char *s1, const char *s2 );
```
- Confronta la stringa s1 con s2
 - Restituisce zero se sono uguali, un numero negativo se s1 < s2, o un numero positivo se s1 > s2 (es. Roma > Bari, Albero < Bari)
- Confronto N caratteri della stringa s1 con s2
 - Restituisce gli stessi valori come sopra
 - Restituisce zero se i primi N caratteri sono uguali.

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

51

Stringhe – Confronto tra Stringhe

```

1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     char string1[] = "test-Ok"; // dichiaro
6     char string2[] = "test-Ok"; // tre variabili
7     char string3[] = "test-Non-Ok"; // di tipo stringa
8
9     if(strcmp(string1,string2)) // string1 == string2
10     printf("Stringhe %s e %s uguali\n", string1, string2);
11     else printf("Stringhe %s e %s non uguali\n", string1, string2);
12
13     if(strcmp(string1,string3)) // string1 != string3
14     printf("Stringhe %s e %s uguali\n", string1, string3);
15     else printf("Stringhe %s e %s non uguali\n", string1, string3);
16
17     int n = 5; // caratteri da confrontare
18     if(!strcmp(string1,string3,n)) // string1 == string3
19     printf("Stringhe %s e %s uguali nei primi %d caratteri\n", string1, string3, n);
20     else printf("Stringhe %s e %s NON uguali nei primi %d caratteri\n", string1, string3, n);
21 }

```

20/03/19 Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

52

Stringhe – Confronto tra Stringhe

```

1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     char string1[] = "test-Ok";      // dichiaro
6     char string2[] = "test-Ok";      // tre variabili
7     char string3[] = "test-Non-Ok"; // di tipo stringa
8
9     if(strcmp(string1,string2) == 0) // string1 == string2
10    printf("Stringhe %s e %s uguali\n", string1, string2);
11    else printf("Stringhe %s e %s non uguali\n", string1, string2);
12
13    if(strcmp(string1,string3) != 0) // string1 != string3
14    printf("Stringhe %s e %s uguali\n", string1, string3);
15    else printf("Stringhe %s e %s non uguali\n", string1, string3);
16
17    int n = 5; // caratteri da confrontare
18    if(lstrcmp(string1,string3,n)) // string1 == string3
19    printf("Stringhe %s e %s uguali nei primi %d caratteri\n", string1, string3, n);
20    else printf("Stringhe %s e %s NON uguali nei primi %d caratteri\n", string1, string3, n);
21 }

```

Perché si usa il «!=» (not) ?

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

53

Stringhe – Confronto tra Stringhe

```

1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     char string1[] = "test-Ok";      // dichiaro
6     char string2[] = "test-Ok";      // tre variabili
7     char string3[] = "test-Non-Ok"; // di tipo stringa
8
9     if(strcmp(string1,string2) == 0) // string1 == string2
10    printf("Stringhe %s e %s uguali\n", string1, string2);
11    else printf("Stringhe %s e %s non uguali\n", string1, string2);
12
13    if(strcmp(string1,string3) != 0) // string1 != string3
14    printf("Stringhe %s e %s uguali\n", string1, string3);
15    else printf("Stringhe %s e %s non uguali\n", string1, string3);
16
17    int n = 5; // caratteri da confrontare
18    if(lstrcmp(string1,string3,n)) // string1 == string3
19    printf("Stringhe %s e %s uguali nei primi %d caratteri\n", string1, string3, n);
20    else printf("Stringhe %s e %s NON uguali nei primi %d caratteri\n", string1, string3, n);
21 }

```

Perché si usa il «!=» (not) ?

Perché `strcmp` restituisce 0 se le due stringhe sono uguali e lo 0 viene identificato in C come «negazione», quindi bisogna utilizzare l'operatore di negazione «!=»

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

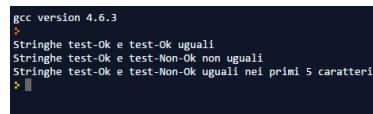
54

Stringhe – Confronto tra Stringhe

```

1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     char string1[] = "test-Ok";      // dichiaro
6     char string2[] = "test-Ok";      // tre variabili
7     char string3[] = "test-Non-Ok"; // di tipo stringa
8
9     if(strcmp(string1,string2) == 0) // string1 == string2
10    printf("Stringhe %s e %s uguali\n", string1, string2);
11    else printf("Stringhe %s e %s non uguali\n", string1, string2);
12
13    if(strcmp(string1,string3) != 0) // string1 != string3
14    printf("Stringhe %s e %s uguali\n", string1, string3);
15    else printf("Stringhe %s e %s non uguali\n", string1, string3);
16
17    int n = 5; // caratteri da confrontare
18    if(lstrcmp(string1,string3,n)) // string1 == string3
19    printf("Stringhe %s e %s uguali nei primi %d caratteri\n", string1, string3, n);
20    else printf("Stringhe %s e %s NON uguali nei primi %d caratteri\n", string1, string3, n);
21 }

```



20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

55

Funzioni per la ricerca nelle stringhe

Prototipo	Descrizione
<code>char *strchr(const char *s, int c);</code>	Locates the first occurrence of character <code>c</code> in string <code>s</code> . If <code>c</code> is found, a pointer to <code>c</code> in <code>s</code> is returned. Otherwise, a <code>NULL</code> pointer is returned.
<code>size_t strcspn(const char *s1, const char *s2);</code>	Determines and returns the length of the initial segment of string <code>s1</code> consisting of characters not contained in string <code>s2</code> .
<code>size_t strspn(const char *s1, const char *s2);</code>	Determines and returns the length of the initial segment of string <code>s1</code> consisting only of characters contained in string <code>s2</code> .
<code>char *strpbrk(const char *s1, const char *s2);</code>	Locates the first occurrence in string <code>s1</code> of any character in string <code>s2</code> . If a character from string <code>s2</code> is found, a pointer to the character in string <code>s1</code> is returned. Otherwise, a <code>NULL</code> pointer is returned.

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

56

Funzioni per la ricerca nelle stringhe

Prototipo	Descrizione
<code>char *strchr(const char *s, int c);</code>	Locates the last occurrence of <code>c</code> in string <code>s</code> . If <code>c</code> is found, a pointer to <code>c</code> in string <code>s</code> is returned. Otherwise, a <code>NULL</code> pointer is returned.
<code>char *strstr(const char *s1, const char *s2);</code>	Locates the first occurrence in string <code>s1</code> of string <code>s2</code> . If the string is found, a pointer to the string in <code>s1</code> is returned. Otherwise, a <code>NULL</code> pointer is returned.
<code>char *strtok(char *s1, const char *s2);</code>	A sequence of calls to <code>strtok</code> break string <code>s1</code> into "tokens"—logical pieces such as words in a line of text—separated by characters contained in string <code>s2</code> . The first call contains <code>s1</code> as the first argument, and subsequent calls to continue tokenizing the same string contain <code>NULL</code> as the first argument. A pointer to the current token is returned by each call. If there are no more tokens when the function is called, <code>NULL</code> is returned.
<code>int strlen(char *s1);</code>	Returns the length of the string <code>s1</code> .

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

57

Funzioni per la ricerca nelle stringhe - Esempi

```
1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main()
5 {
6     char string[] = "prova_stringa";
7
8     // Trova la prima occorrenza della stringa, e restituisce la parte rimanente
9     printf("Parte rimanente dopo la s: %s\n", strchr(string, 's')); // stampa 'stringa'
10    // Restituisce il numero di caratteri prima della prima occorrenza di quel carattere
11    printf("Lunghezza Stringa prima della S: %d\n", strcspn(string, "s")); // stampa '6' - nb: parametro è una stringa non un char
12
13    int length = strlen(string); // lunghezza stringa
14    printf("%d", length); // stampa la lunghezza della stringa = 13
15
16 }
```

gcc-version 4.6.3
 Parte rimanente dopo la s: stringa
 Lunghezza Stringa prima della S: 6
 13:

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

58

Funzioni per la conversione delle stringhe

Prototype	Description
<code>double atof(const char *nPtr);</code>	Converts the string <code>nPtr</code> to double.
<code>int atoi(const char *nPtr);</code>	Converts the string <code>nPtr</code> to int.
<code>long atol(const char *nPtr);</code>	Converts the string <code>nPtr</code> to long int.
<code>double strtod(const char *nPtr, char **endPtr);</code>	Converts the string <code>nPtr</code> to double.
<code>long strtol(const char *nPtr, char **endPtr, int base);</code>	Converts the string <code>nPtr</code> to long.
<code>unsigned long strtoul(const char *nPtr, char **endPtr, int base);</code>	Converts the string <code>nPtr</code> to unsigned long.

Prendono in input una stringa e restituiscono in output la stringa convertita in un numero.

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

59

Funzioni per la conversione delle stringhe - Esempi

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     double d; /* variable to hold converted string */
7
8     d = atof("99.0");
9
10    printf( "%s%.3f\n%s%.3f\n",
11            "The string \"99.0\" converted to double is ", d,
12            "The converted value divided by 2 is ",
13            d / 2.0 );
14
15    return 0;
16 }
```

The string "99.0" converted to double is 99.000
 The converted value divided by 2 is 49.500

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

60

Funzioni per la conversione delle stringhe - Esempi

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     double d; /* variable to hold converted string */
7
8     d = atof("99.0");
9
10    printf("%%.3f\n%.3f\n",
11           "The string \"99.0\" converted to double is ", d,
12           "The converted value divided by 2 is ",
13           d / 2.0);
14
15    return 0;
16
17 }
```

The string "99.0" converted to double is 99.000
The converted value divided by 2 is 49.500

La funzione prende in input un valore in formato stringa e lo converte in un float (o in un altro formato, a seconda della funzione)

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

61

Funzioni per la conversione delle stringhe - Esempi

```

3 #include <stdio.h>
4 #include <stdlib.h>
5
6 int main()
7 {
8     /* initialize string pointer */
9     const char *string = "51.2% are admitted";
10
11    double d; /* variable to hold converted sequence */
12    char *stringPtr; /* create char pointer */
13
14    d = strtod(string, &stringPtr);
15
16    printf("The string \"%s\" is converted to the\n", string);
17    printf("double value %.2f and the string \"%s\"\n", d, stringPtr);
18
19    return 0;
20
21 } /* end main */
```

The string "51.2% are admitted" is converted to the double value 51.20 and the string "% are admitted"

La funzione prende in input un valore in formato stringa e lo converte in un float (o in un altro formato, a seconda della funzione). La parte rimanente viene memorizzata in una ulteriore stringa.

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

62

Funzioni per la manipolazione delle stringhe

Prototype	Description
char *strcpy(char *s1, const char *s2)	Copies string s2 into array s1. The value of s1 is returned.
char *strncpy(char *s1, const char *s2, size_t n)	Copies at most n characters of string s2 into array s1. The value of s1 is returned.
char *strcat(char *s1, const char *s2)	Appends string s2 to array s1. The first character of s2 overwrites the terminating null character of s1. The value of s1 is returned.
char *strncat(char *s1, const char *s2, size_t n)	Appends at most n characters of string s2 to array s1. The first character of s2 overwrites the terminating null character of s1. The value of s1 is returned.

Permettono di copiare una parte dei caratteri di una stringa in un'altra stringa oppure di concatenare due stringhe in una più grande

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

63

Funzioni per la manipolazione delle stringhe

```

4 #include <stdio.h>
5 #include <string.h>
6
7 int main()
8 {
9     char stringa_a[30] = "Provo ";
10    char stringa_b[30] = "Concatenazione";
11    char stringa_c[30] = "";
12    char stringa_d[30] = "Copia Stringa";
13    char stringa_e[30] = "";
14
15    // Concateno la stringa B alla stringa A
16    printf("%s", strcat(stringa_a, stringa_b));
17    // Concateno i primi 15 caratteri della stringa A a C
18    printf("\n%s", strncat(stringa_c, stringa_a, 15));
19
20    // Copio una stringa in un'altra
21    printf("\n%s", stringa_d);
22    strcpy(stringa_e, stringa_d);
23    printf("\n%s", stringa_e);
24
25    // Lunghezza della stringa
26    printf("\nLunghezza Stringa A: %d", strlen(stringa_a));
27 }
```

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

64

Funzioni per la manipolazione delle stringhe

```

4  #include <stdio.h>
5  #include <string.h>
6
7- int main() {
8
9  char stringa_a[30] = "Provo ";
10 char stringa_b[30] = "Concatenazione";
11 char stringa_c[30] = "";
12 char stringa_d[30] = "Copia Stringa";
13 char stringa_e[30] = "";
14
15 // Concateno la stringa B alla stringa A
16 printf("%s", strcat(stringa_a, stringa_b));
17 // Concateno i primi 15 caratteri della stringa A a C
18 printf("\n%s", strncat(stringa_c, stringa_a, 15));
19
20 // Copio una stringa in un'altra
21 printf("\n%s", stringa_d);
22 strcpy(stringa_e, stringa_d);
23 printf("\n%s", stringa_e);
24
25 // Lunghezza della stringa
26 printf("\nLunghezza Stringa A: %d", strlen(stringa_a));
27 }

```

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

65

Elaborazione di Stringhe

- **Approfondimenti:**

- Capitolo 8 del libro Deitel & Deitel

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

66

Domande?

20/03/19

Veronica Rossano - Linguaggio C (parte 3) Laboratorio di Informatica (INF Track B) – Università degli Studi di Bari – A.A. 2018/2019

67