5 Stringhe

Le stringhe di caratteri sono gestite in C come dei vettori di `char` con alla fine un "tappo" dato dal carattere `'\0'`. E' possibile acquisire un'intera stringa di caratteri in una sola istruzione grazie all'istruzione `scanf('%s',variabile);` oppure `gets(variabile)`. Entrambi devono essere seguiti dall'istruzione `fflush(stdin);` che serve ad evitare errori nella successiva acquisizione di stringhe. Mentre l'istruzione `scanf('%s',variabile);` ferma la sua acquisizione al primo spazio, `gets(variabile)` inserisce nella variable tutto l'input fino al primo invio.

Poichè le stringhe necessitano di un tappo, ogni volta che ne dichiaro una devo aggiungere un elemento per il tappo, ossia se voglio una stringa lunga `N_MAX` dovrò dichiarare un array di `N_MAX+1` elementi.

Esiste una libreria di C che gestisce le stringhe `string.h` essa ci permette di:

- ottenere la lunghezza di una stringa con la funzione `strlen(stringa);`
- confrontare due stringhe `strcmp(stringa1,stringa2)` e restituisce zero se sono uguali, un numero negativo se `stringa1` viene prima in ordine lessicografico di `stringa2` e un numero positivo se `stringa2` viene prima in ordine lessicografico di `stringa1` (se i caratteri sono tutti maiuscoli o tutti minuscoli);
- copiare una stringa `stringa2` in un'altra stringa `stringa1` con la funzione `strcpy(stringa1,stringa2);`
- concatenare due stringhe con la funzione `strcat(stringa1,stringa2);`

5.1 Esercizi

Esercizio 5.1

Implementare la stessa funzionalità della `strcat()` senza ovviamente utilizzare la funzione stessa e senza utilizzare la `strlen()`. Considerare delle stringhe in ingresso di al massimo 20 caratteri.

Esercizio 5.2

1. Scrivere un programma che determini se una frase acquisita da tastiera è palindroma. Controllare se sono palindrome le seguenti frasi (senza spazi e maiuscole):
   - Eri un nano non annuire
   - Ad una vera pia donna dei simili fili misi e annodai: pareva nuda
   - O mordo tua nuora, o ari un autodromo
   - Occorre portar aratro per Rocco
2. (Bonus) Modificare il programma precedente per considerare stringhe in input contenenti anche caratteri speciali e spazi.

Esercizio 5.3

Implementare la seguente variante del cifrario di Cesare[1]. Dopo aver acquisito una messaggio di massimo 160 caratteri (alfabetici minuscoli), il programma dovrà chiedere all'utente una chiave (numero intero K, $1 < K < 25$).

Il programma stamperà il messaggio cifrato, ottenuto traslando ogni lettera di K posizioni in avanti (dove K è la chiave).

Dopo aver stampato il messaggio cifrato, il programma chiede all'utente di inserire un messaggio (che si assume sia stato cifrato con lo stesso algoritmo di cui sopra). Tale messaggio sarà dunque decifrato dal programma, il quale dovrà svolgere l'operazione inversa.

Infine, il messaggio decifrato sarà stampato a video.

Esercizio 5.4

[1] `http://it.wikipedia.org/wiki/Cifrario_di_Cesare`

1. Scrivere un programma che converta una stringa in alfabeto farfallino. Nell'alfabeto farfallino, ogni vocale è raddoppiata, e tra le due vocali è inserita la lettera 'f'. Ad esempio, "ciao" diventa "cifiafaofo".

2. (bonus) Scrivere il medesimo programma utilizzando soltanto una variabile per memorizzare la stringa tradotta (i.e., senza copiare la traduzione in una nuova variabile).

Esercizio 5.5

Scrivere un programma che considera, da una stringa letta da tastiera, solo le lettere maiuscole e minuscole. Il programma dovrà poi calcolare le occorrenze di ogni carattere nella stringa. Le occorrenze saranno poi stampate a video a video come istogramma con gli asterischi. Si assuma che la stringa possa avere una dimensione massima di 256 elementi.

Si veda l'esempio seguente.

```
str = Ciao Come Stai? Non c'e' male GRAZIE! Mi trovo molto bene in questa citta'.

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 |
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 |
```

Esercizio 5.6

Scrivere un programma che determini se due parole acquisite da tastiera sono una l'anagramma dell'altra.

Soluzioni

Soluzione dell'esercizio 5.1

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

#define MAX_LEN 100

void main(){

char str1[MAX_LEN+1],str2[MAX_LEN+1];
char str12[2*MAX_LEN+1];
int lungh1,lungh2;
int i;

printf("Inserire la prima stringa: ");
gets(str1);
fflush(stdin);

printf("Inserire la seconda stringa: ");
gets(str2);
fflush(stdin);

//for (lungh1 = 0; str1[lungh1] == '/0'; lungh1++)
lungh1 = 0;
for (i = 0; str1[i] != '\0'; i++)
    lungh1++;

lungh2 = 0;
for (i = 0; str2[i] != '\0'; i++)
    lungh2++;

//printf("%d   %d",lungh1,lungh2);


for (i = 0; i <= lungh1; i++)
    str12[i] = str1[i];

for (i = 0; i < lungh2; i++)
    str12[i+lungh1+1] = str2[i];

str12[lungh1+lungh2] = '\0';

printf("%s",str12);


}
```

Soluzione dell'esercizio 5.2

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

#define LEN 256

void main(){
```

```
char parola[LEN+1];
int i,j,palindromo;
int n;

printf("Inserire la frase: ");
gets(parola);
fflush(stdin);

n_car = strlen(parola),

palindromo = 1;
j = n_car - 1;
for (i = 0; (i < n_car/2) && (palindromo == 1); i++) {
    if (parola[i] != parola[j])
        palindromo = 0;
        j--;
}

if (palindromo == 1)
    printf("E' palindromo");
else
    printf("Non e' palindromo");

}
```

Soluzione dell'esercizio 5.3

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

#define MAX_LEN 160


void main(){

char messaggio[MAX_LEN+1;
char cifrato[MAX_LEN+1];

int lungh;
int chiave;
int i;

printf("Inserire un messaggio: ");
gets(messaggio);
fflush(stdin);
lungh = strlen(messaggio);

do {
    printf("Inserire una chiave: ");
    scanf("%d",&chiave);
} while ( chiave < 1 || chiave > 25 );
fflush(stdin);

for (i = 0; i <= lungh; i++) {
    if (messaggio[i] > 96 && messaggio[i] < 123)
        cifrato[i] = (messaggio[i] - 97 + chiave) % 26 + 97;
        cifrato[i] = messaggio[i] + chiave;
    else
        cifrato[i] = messaggio[i];
```

```
}

printf("Messaggio cifrato: ");
printf("%s\n",cifrato);

printf("Inserire un messaggio cifrato ");
gets(cifrato);
fflush(stdin);

for (i = 0; i <= lungh; i++) {
    if (cifrato[i] > 96 && cifrato[i] < 123)
        messaggio[i] = (cifrato[i] - 97 + (26 - chiave) ) % 26 + 97;
    else
        messaggio[i] = cifrato[i];

}

printf("Messaggio in chiaro: ");
printf("%s\n",messaggio);

}
```

Soluzione dell'esercizio 5.4

1.

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

#define LEN 256


void main(){

char parola[LEN+1];
char parola_tradotta[LEN+1];
int n,i,j;
int accettabile;

//Acquisizione
printf("Inserire la parola da tradurre");
scanf("%s",parola);

n = strlen(parola);

j = 0;
accettabile = 1;

//Traduzione
for (i = 0; i <= n; i++) {
    // Copio lettera
    if (j < LEN+1) {
        parola_tradotta[j] = parola[i];
        if (parola[i] == 'a' || parola[i] == 'e' || parola[i] == 'i' || parola[i
            ] == 'o' || parola[i] == 'u') {
            parola_tradotta[j+1] = 'f';
            parola_tradotta[j+2] = parola[i];
            j += 3;
        }
        else
            j++;
```

```
    }
    else
        accettabile = 0;

}
if (accettabile == 1)
    printf("%s",parola_tradotta);
else
    printf("Parola troppo lunga da tradurre");

}
```

2.

```
#define LEN 256

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

int main () {

    int i, k, len;
    char str[LEN];

    /* Acquisizione */
    do {
        printf("Inserire una frase da tradurre: ");
        gets(str);
        len = strlen(str);
    } while (len > LEN);

    /* Lettura da sx a dx */
    for (i = 0; i <= len; i++) {
        if (str[i] == 'a'||
            str[i] == 'e'||
            str[i] == 'i'||
            str[i] == 'o'||
            str[i] == 'u') {

            /* Shift a dx di 2 posizioni */
            for(k = len; k > i ; k--)
                str[k + 2] = str[k];

            str[i+1] = 'f';
            str[i+2] = str[i];

            i = i + 2;
            len = len + 2;
        }
    }

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

Soluzione dell'esercizio 5.5

```
#define LEN 256

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

```
/*
  Tabella ascii:

   0 nul    1 soh    2 stx    3 etx    4 eot    5 enq    6 ack    7 bel
   8 bs     9 ht    10 nl    11 vt    12 np    13 cr    14 so    15 si
  16 dle   17 dc1   18 dc2   19 dc3   20 dc4   21 nak   22 syn   23 etb
  24 can   25 em    26 sub   27 esc   28 fs    29 gs    30 rs    31 us
  32 sp    33  !    34  "    35  #    36  $    37  %    38  &    39  '
  40  (    41  )    42  *    43  +    44  ,    45  -    46  .    47  /
  48  0    49  1    50  2    51  3    52  4    53  5    54  6    55  7
  56  8    57  9    58  :    59  ;    60  <    61  =    62  >    63  ?
  64  @    65  A    66  B    67  C    68  D    69  E    70  F    71  G
  72  H    73  I    74  J    75  K    76  L    77  M    78  N    79  O
  80  P    81  Q    82  R    83  S    84  T    85  U    86  V    87  W
  88  X    89  Y    90  Z    91  [    92  \    93  ]    94  ^    95  _
  96  `    97  a    98  b    99  c   100  d   101  e   102  f   103  g
 104  h   105  i   106  j   107  k   108  l   109  m   110  n   111  o
 112  p   113  q   114  r   115  s   116  t   117  u   118  v   119  w
 120  x   121  y   122  z   123  {   124  |   125  }   126  ~   127 del
*/

int main()
{
    int i, j;
    int ast;

    int hist[25]; //z-a -> 90-65 -> 25
    int HIST[25]; //Z-A -> 122-97 -> 25

    char lettera;
    char str[LEN];

    //inizializzazione dell'istogramma
    for (i = 0; i < 25; i++)
        hist[i] = HIST[i] = 0;

    //acquisizione stringa
    printf("str = ");
    gets(str);

    printf("\n");

    /*
     * Esempio:
     * str[3] = 'd'
     *
     * Bisogna incrementare l'istogramma in posizione 4
     * hist['d'-'a'] = hist[100-97] = hist[3]
     */
    for (i = 0; i < strlen(str); i++) {
        if (str[i] > 'a' && str[i] < 'z')
            hist[str[i]-'a']++;

        if (str[i] > 'A' && str[i] < 'Z')
            HIST[str[i]-'A']++;
    }

    //per ogni lettera
    for (i = 0; i < 25*2; i++)
    {
        //stampa il giusto numero di asterischi
        if (i < 25) {
```

```
            ast = hist[i];
            lettera = 'a' + i;
        } else {
            ast = HIST[i-25];
            lettera = 'A' + i - 25;
        }

        printf("%c | ", lettera);

        for (j = 0; j < ast; j++)
            printf("*");

        printf("\n");
    }
}
```

Soluzione dell'esercizio 5.6

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

#define MAX_LEN 100
void main(){

char stringa1[MAX_LEN];
char stringa2[MAX_LEN];

int i,j;
int trovato, uguali;

printf("Inserire la prima stringa: ");
scanf("%s",&stringa1);
fflush(stdin);
printf("Inserire la seconda stringa: ");
scanf("%s",&stringa2);
fflush(stdin);

uguali = 1;
if (strlen(stringa1) != strlen(stringa2))
    uguali = 0;

for (i = 0; i < strlen(stringa1) && uguali; i++) {
    trovato = 0;
    for (j = 0; j < strlen(stringa2) && !trovato; j++)
        if (stringa1[i] == stringa2[j]){
            stringa2[j] = '-';
            trovato = 1;
        }

    if (trovato == 0)
        uguali = 0;
}

if (uguali)
    printf("Le due stringhe sono una l'anagramma dell'altra");
else
    printf("No");
}
```