APPELLO SCRITTO DI PROGRAMMAZIONE 1
CORSO DI LAUREA IN MATEMATICA
UNIVERSITÀ DEGLI STUDI DI MILANO

2013–2014

20.VI.2014

VINCENZO MARRA

INDICE

Esercizio 1	1
Leggere stringhe e caratteri	1
Esercizio 2	2
Convertire una stringa in maiuscola o minuscola.	2
Esercizio 3	2
Calcolare la frequenza delle lettere in una stringa.	2
Esercizio 4	4
Traslare le lettere di una stringa.	4
Esercizio 5	4
Procedura `main`: Un semplice programma di elaborazione delle stringhe.	4
Istruzioni per la consegna	6

ESERCIZIO 1

Leggere stringhe e caratteri.

Si scriva una funzione

```
int leggistr(char *)
```

che legga una stringa inserita dall'utente, fino al carattere `'\n'` compreso. Il parametro in ingresso è un puntatore alla zona di memoria in cui sarà memorizzata la stringa letta. La funzione restituisce -1 se si è verificato un errore in lettura, o se si è raggiunta la fine dello stream, e 0 se la lettura è andata a buon fine. Usate la funzione della libreria standard

```
char* fgets(char *letta, int lung, FILE* stream)
```

applicata allo stream `stdin`. Si ricorda che `letta` è il puntatore all'array di caratteri dove sarà memorizzata la stringa letta, e che `lung` è il massimo numero di caratteri da leggere: ponete `lung` pari a 256. La funzione `fgets` restituisce un puntatore `NULL` se si raggiunge la fine dello stream o si riscontra un errore in lettura.

Ultima revisione: 27 giugno 2014.

Si scriva poi una seconda funzione

```
int leggicar(void)
```

che legga un carattere inserito dall'utente. A differenza della funzione `getchar` della libreria standard, la funzione `int leggicar()` legge l'intera riga digitata dall'utente, fino al carattere `'\n'` compreso, usando la funzione `fgets` con parametro `lung` pari a 256; pone il risultato in un array di caratteri temporaneo, costituito da 256 elementi; restituisce il primo carattere dell'array al chiamante, ignorando gli eventuali caratteri rimanenti. Nel caso in cui si raggiunga la fine dello stream o si riscontri un errore in lettura, la funzione restituisce `EOF`. (*Nota.* Poiché `EOF` è un valore intero predefinito non ricompreso nella tabella ASCII dei caratteri, il tipo del valore restituito dalla funzione `leggicar()` deve essere `int` e non `char` proprio affinché la funzione possa restituire `EOF` nei casi opportuni.)

Si suggerisce di scrivere una semplice procedura `main` che permetta di testare il buon funzionamento delle due funzioni scritte per risolvere questo esercizio, prima di proseguire con la risoluzione del tema d'esame. Il medesimo suggerimento si applica tacitamente a tutti gli esercizi seguenti.

Esercizio 2

Convertire una stringa in maiuscola o minuscola.

Si scriva una funzione

```
void convma(char *)
```

che accetti in ingresso una stringa, che si assume terminata da `'\n'`, e la converta in lettere maiuscole. Se il puntatore ricevuto in ingresso è `NULL` la funzione restituisce il controllo al chiamante senza eseguire alcuna operazione. Altrimenti la funzione usa la funzione della libreria standard `int toupper(int c)` per convertire ciascun carattere della stringa in maiuscola; essa restituisce il corrispettivo in maiuscola del carattere `c`, o `c` stesso se tale corrispettivo non esiste. Dovrete includere il file di intestazione `ctype.h` della libreria standard, in cui `toupper` è definita.

Si scriva poi un'analoga funzione

```
void convmi(char *letta)
```

che converta la stringa in lettere minuscole. Si usi la funzione `int tolower(int c)` per la conversione del carattere `c` in minuscola.

Esercizio 3

Calcolare la frequenza delle lettere in una stringa.

Si scriva una funzione

```
void freq(char *s)
```

che accetti in ingresso una stringa `s`, che si assume terminata da `'\n'`, e calcoli, per ciascuna lettera dell'alfabeto inglese, la frequenza della lettera all'interno della stringa.

Più in dettaglio, la funzione calcola la frequenza sia delle lettere in minuscola che delle lettere in maiuscola. L'alfabeto inglese in minuscola corrisponde ai caratteri

Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	*[NULL]*
1	1	1	1	*[START OF HEADING]*
2	2	10	2	*[START OF TEXT]*
3	3	11	3	*[END OF TEXT]*
4	4	100	4	*[END OF TRANSMISSION]*
5	5	101	5	*[ENQUIRY]*
6	6	110	6	*[ACKNOWLEDGE]*
7	7	111	7	*[BELL]*
8	8	1000	10	*[BACKSPACE]*
9	9	1001	11	*[HORIZONTAL TAB]*
10	A	1010	12	*[LINE FEED]*
11	B	1011	13	*[VERTICAL TAB]*
12	C	1100	14	*[FORM FEED]*
13	D	1101	15	*[CARRIAGE RETURN]*
14	E	1110	16	*[SHIFT OUT]*
15	F	1111	17	*[SHIFT IN]*
16	10	10000	20	*[DATA LINK ESCAPE]*
17	11	10001	21	*[DEVICE CONTROL 1]*
18	12	10010	22	*[DEVICE CONTROL 2]*
19	13	10011	23	*[DEVICE CONTROL 3]*
20	14	10100	24	*[DEVICE CONTROL 4]*
21	15	10101	25	*[NEGATIVE ACKNOWLEDGE]*
22	16	10110	26	*[SYNCHRONOUS IDLE]*
23	17	10111	27	*[ENG OF TRANS. BLOCK]*
24	18	11000	30	*[CANCEL]*
25	19	11001	31	*[END OF MEDIUM]*
26	1A	11010	32	*[SUBSTITUTE]*
27	1B	11011	33	*[ESCAPE]*
28	1C	11100	34	*[FILE SEPARATOR]*
29	1D	11101	35	*[GROUP SEPARATOR]*
30	1E	11110	36	*[RECORD SEPARATOR]*
31	1F	11111	37	*[UNIT SEPARATOR]*
32	20	100000	40	*[SPACE]*
33	21	100001	41	!
34	22	100010	42	"
35	23	100011	43	#
36	24	100100	44	$
37	25	100101	45	%
38	26	100110	46	&
39	27	100111	47	'
40	28	101000	50	(
41	29	101001	51	)
42	2A	101010	52	*
43	2B	101011	53	+
44	2C	101100	54	,
45	2D	101101	55	-
46	2E	101110	56	.
47	2F	101111	57	/
48	30	110000	60	**0**
49	31	110001	61	**1**
50	32	110010	62	**2**
51	33	110011	63	**3**
52	34	110100	64	**4**
53	35	110101	65	**5**
54	36	110110	66	**6**
55	37	110111	67	**7**
56	38	111000	70	**8**
57	39	111001	71	**9**
58	3A	111010	72	:
59	3B	111011	73	;
60	3C	111100	74	<
61	3D	111101	75	=
62	3E	111110	76	>
63	3F	111111	77	?
64	40	1000000	100	@
65	41	1000001	101	**A**
66	42	1000010	102	**B**
67	43	1000011	103	**C**
68	44	1000100	104	**D**
69	45	1000101	105	**E**
70	46	1000110	106	**F**
71	47	1000111	107	**G**
72	48	1001000	110	**H**
73	49	1001001	111	**I**
74	4A	1001010	112	**J**
75	4B	1001011	113	**K**
76	4C	1001100	114	**L**
77	4D	1001101	115	**M**
78	4E	1001110	116	**N**
79	4F	1001111	117	**O**
80	50	1010000	120	**P**
81	51	1010001	121	**Q**
82	52	1010010	122	**R**
83	53	1010011	123	**S**
84	54	1010100	124	**T**
85	55	1010101	125	**U**
86	56	1010110	126	**V**
87	57	1010111	127	**W**
88	58	1011000	130	**X**
89	59	1011001	131	**Y**
90	5A	1011010	132	**Z**
91	5B	1011011	133	[
92	5C	1011100	134	\
93	5D	1011101	135	]
94	5E	1011110	136	^
95	5F	1011111	137	_
96	60	1100000	140	`
97	61	1100001	141	**a**
98	62	1100010	142	**b**
99	63	1100011	143	**c**
100	64	1100100	144	**d**
101	65	1100101	145	**e**
102	66	1100110	146	**f**
103	67	1100111	147	**g**
104	68	1101000	150	**h**
105	69	1101001	151	**i**
106	6A	1101010	152	**j**
107	6B	1101011	153	**k**
108	6C	1101100	154	**l**
109	6D	1101101	155	**m**
110	6E	1101110	156	**n**
111	6F	1101111	157	**o**
112	70	1110000	160	**p**
113	71	1110001	161	**q**
114	72	1110010	162	**r**
115	73	1110011	163	**s**
116	74	1110100	164	**t**
117	75	1110101	165	**u**
118	76	1110110	166	**v**
119	77	1110111	167	**w**
120	78	1111000	170	**x**
121	79	1111001	171	**y**
122	7A	1111010	172	**z**
123	7B	1111011	173	{
124	7C	1111100	174	|
125	7D	1111101	175	}
126	7E	1111110	176	~
127	7F	1111111	177	*[DEL]*

FIGURA 1. Il codice ASCII.

il cui codice ASCII è compreso nell'intervallo 97–122, estremi compresi. L'alfabeto inglese in maiuscola corrisponde ai caratteri il cui codice ASCII è compreso nell'intervallo 65–90, estremi compresi. Si veda la Figura 1.

La funzione memorizza le frequenze delle lettere minuscole e delle lettere maiuscole dell'alfabeto inglese in due array di `double`, ciascuno contenente 26 elementi corrispondenti alle lettere, di nome:

```
frqmi
frqma
```

rispettivamente. La funzione non sovrascrive gli elementi di questi array con le nuove frequenze, ma *aggiunge* ciascuna frequenza al valore precedentemente contenuto nell'array relativo. Per esempio, se l'elemento `freqmi[0]` vale 0.5 e la stringa `s` è costituita dal solo carattere `'a'` seguito da `'\n'`, la funzione aggiunge a 0.5 il valore 1, con risultato finale 1.5. I due array di frequenze devono essere dichiarati come variabili globali, e dunque accessibili a tutte le funzioni del file sorgente.

Per implementare la funzione `void freq(char *s)` si scriva una funzione ausiliaria

```
double calcf(char *s, char c)
```

che restituisce la frequenza del carattere `c` all'interno della stringa `s`, come valore di tipo `double`.

Esercizio 4

Traslare le lettere di una stringa.

Si scriva una funzione che permetta di traslare in avanti di un numero specificato di posizioni le lettere della stringa ricevuta in ingresso. Si usi l'alfabeto inglese. Il prototipo della funzione è

```
void trasla(char *s, int t, int minma)
```

La funzione accetta in ingresso la stringa (puntata da) `s` da traslare, che si assume terminata da `'\n'`, e due interi `t` e `minma`, di cui `t` è assunto non negativo. Se `minma` è positivo, la funzione assume che la stringa `s` sia costituita da lettere in maiuscola (assieme ad eventuali altri caratteri non alfabetici, per esempio gli spazi). Altrimenti la funzione assume che la stringa sia costituita da lettere in minuscola. Se `s` è `NULL` la funzione restituisce immediatamente il controllo al chiamante. Altrimenti la funzione sostituisce ciascuna lettera della stringa `s` con la lettera che si trova `t` posizioni più avanti nell'alfabeto inglese in minuscola o maiuscola, a seconda del caso. Qualunque valore non negativo di `t` è ammesso: la traslazione del carattere `c` di `t` posizioni in avanti è definita modulo il numero di lettere dell'alfabeto, qui pari a 26: quindi, per esempio, la traslazione di `"A"` con `t` uguale a 26 è di nuovo `"A"`. La funzione lascia invariati i caratteri non alfabetici.

Supponiamo che la stringa `s` sia `"ciao, come stai"`. Allora l'invocazione di `trasla` con parametri `s`, `3` e `-1` modificherà la stringa `s` come segue:

```
fldr, frph vwdl
```

Come detto, i caratteri non alfabetici sono ignorati. Per stabilire se un carattere è alfabetico si usi la funzione `int isalpha(int c)` di `ctype`, che restituisce un valore non nullo se `c` è un carattere alfabetico, e zero altrimenti.

Se invece la stringa `s` fosse `"ZZAA"`, l'invocazione di `trasla` con parametri `s`, `2` e `1` modificherebbe la stringa `s` come segue:

```
BBCC
```

Esercizio 5

Procedura `main`: Un semplice programma di elaborazione delle stringhe.

Si scriva una funzione

```
int main(void)
```

permetta di eseguire alcune semplici operazioni sulle stringhe inserite dall'utente, sfruttando il codice scritto per la soluzione degli esercizi precedenti. Per memorizzare la stringa corrente inserita dall'utente usate un array di 256 caratteri.

```
a/A     0.133333        0.504762
b/B     0.000000        0.047619
c/C     0.000000        0.000000
d/D     0.052632        0.000000
e/E     0.319298        0.000000
f/F     0.210526        0.142857
g/G     0.157895        0.000000
h/H     0.052632        0.142857
i/I     0.066667        0.095238
j/J     0.066667        0.180952
k/K     0.000000        0.000000
l/L     0.000000        0.000000
m/M     0.000000        0.000000
n/N     0.000000        0.000000
o/O     0.000000        0.047619
p/P     0.000000        0.047619
q/Q     0.000000        0.000000
r/R     0.119298        0.000000
s/S     0.157895        0.190476
t/T     0.000000        0.000000
u/U     0.000000        0.000000
v/V     0.000000        0.000000
w/W     0.210526        0.000000
x/X     0.000000        0.000000
y/Y     0.000000        0.000000
```

FIGURA 2. Frequenze delle lettere minuscole (seconda colonna) e maiuscole (terza colonna).

All'avvio, il programma visualizza il menu seguente:

1. `Inserisci stringa.`
2. `Visualizza stringa corrente.`
3. `Converti in maiuscola.`
4. `Converti in minuscola.`
5. `Trasla.`
6. `Visualizza frequenze.`
7. `Esci.`

Se l'utente seleziona `7` il programma termina. Altrimenti, dopo l'esecuzione di una qualunque opzione, il programma visualizza nuovamente il menu.

Se l'utente seleziona `2` è visualizzata la stringa corrente, o un messaggio appropriato se non è stata ancora inserita alcuna stringa.

Se l'utente seleziona `3` la stringa corrente è convertita in maiuscola; è visualizzato un un messaggio appropriato se non è stata ancora inserita alcuna stringa. Analogamente per `4`.

Per poter traslare la stringa corrente è necessario che essa sia già stata convertita in maiuscola o in minuscola tramite le opzioni `3` o `4`. Si intende nel seguito che la traslazione di una stringa in maiuscola produce una stringa in maiuscola, e quella di una in minuscola produce una stringa in minuscola. Se l'utente sceglie `5`, ma non è stata ancora inserita alcuna stringa, è visualizzato un messaggio appropriato.

Se esiste invece una stringa corrente, ma essa non è stata ancora convertita in maiuscola oppure in minuscola, il programma informa l'utente della necessità di convertirla prima di poter eseguire la traslazione, e torna al menu. Se esiste una stringa corrente ed essa è già stata precedentemente convertita (in maiuscola o in minuscola), il programma chiede all'utente di quante lettere voglia traslare la stringa. Se l'intero letto è negativo il programma informa l'utente di non poter procedere, e torna al menu. Altrimenti il programma sostituisce la stringa corrente con la sua traslazione in avanti del numero di lettere indicato dall'utente.

Se l'utente seleziona `6` il programma visualizza l'elenco delle frequenze di ciascuna lettera minuscola e maiuscola dell'alfabeto inglese nell'insieme di tutte le stringhe inserite fino a quel momento dall'utente. All'avvio del programma, e fino a quando l'utente non abbia inserito almeno una stringa contenente almeno un carattere alfabetico, tali frequenze sono tutte nulle. La Figura 2 riporta un esempio di possibile formattazione delle frequenze in colonne, cui è consigliato di attenersi.

Istruzioni per la consegna

- Funzioni da consegnare.

```
int leggistr(char *)
int leggicar(void)
void convmi(char *)
void convma(char *)
void freq(char *)
double calcf(char *, char c)
void trasla(char *, int, int)
int main(void)
```

  Tutte le eventuali altre funzioni ausiliarie che abbiate scritto.

- Sito per la consegna:

  `https://upload.mat.unimi.it`

- Autenticarsi con le proprie credenziali di posta elettronica d'ateneo.
- Consegnare solo il codice sorgente (file `*.c`), non l'eseguibile.

(V. Marra) Dipartimento di Matematica *Federigo Enriques*, Università degli Studi di Milano, via Cesare Saldini, 50, I-20133 Milano
E-mail address: vincenzo.marra@unimi.it