Esercitazione 1: stringhe

Irene Finocchi

finocchi@di.uniroma1.it

Stringhe: recap

- Stringa = array di caratteri che termina con il carattere '\0'
- 'a' ≠ "a"
- Codice ASCII: associa ad ogni carattere un intero (al carattere di fine stringa '\0' è associato il valore 0)
- Funzioni per manipolare stringhe: string.h

Standard ASCII Set

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Extended ASCII Set

128	Ç	144	É	161	í	177	▒	193	┴	209	╤	225	ß	241	±
129	ü	145	æ	162	ó	178	▓	194	┬	210	╥	226	Γ	242	≥
130	é	146	Æ	163	ú	179	│	195	├	211	╙	227	π	243	≤
131	â	147	ô	164	ñ	180	┤	196	─	212	╘	228	Σ	244	⌠
132	ä	148	ö	165	Ñ	181	╡	197	┼	213	╒	229	σ	245	⌡
133	à	149	ò	166	ª	182	╢	198	╞	214	╓	230	µ	246	÷
134	å	150	û	167	º	183	╖	199	╟	215	╫	231	τ	247	≈
135	ç	151	ù	168	¿	184	╕	200	╚	216	╪	232	Φ	248	°
136	ê	152	ÿ	169	⌐	185	╣	201	╔	217	┘	233	Θ	249	∙
137	ë	153	Ö	170	¬	186	║	202	╩	218	┌	234	Ω	250	·
138	è	154	Ü	171	½	187	╗	203	╦	219	█	235	δ	251	√
139	ï	156	£	172	¼	188	╝	204	╠	220	▄	236	∞	252	ⁿ
140	î	157	¥	173	¡	189	╜	205	═	221	▌	237	φ	253	²
141	ì	158	₧	174	«	190	╛	206	╬	222	▐	238	ε	254	■
142	Ä	159	ƒ	175	»	191	┐	207	╧	223	▀	239	∩	255	
143	Å	160	á	176	░	192	└	208	╨	224	α	240	≡		

Source: www.LookupTables.com

Esercizio 1 — Confrontare due stringhe in ordine lessicografico

```
int my_strcmp (char *s1, char *s2) {
  while ( *s1==*s2 && *s1!='\0' ) {
     s1++;
     s2++;
  }
  return (*s1-*s2);
}
```

Perché manca il test `*s2 != '\0'`?

® Esercizio 1 — Confrontare due stringhe in ordine lessicografico

```
int my_strcmp (char *s1, char *s2) {
  if (*s1 == '\0' || *s1 != *s2)
     return *s1 - *s2;
  else return(my_strcmp (s1+1, s2+1));
}
```

Esercizio 2

Confrontare i primi n caratteri di due stringhe in ordine lessicografico

Find the bug

```
int strncmp(char *s, char *t, int n) {
  int i;
  for(i=0; i<n && s[i]==t[i] &&
                  s[i]!='\0'; i++) ;
  return (*s - *t);
}
```


Esercizio 2

Confrontare i primi n caratteri di due stringhe in ordine lessicografico


```
int strncmp(char *s, char *t, int n) {
  int i;
  for(i=0; i<n && s[i]==t[i] &&
                  s[i]!='\0'; i++) ;
  if (i==n) return 0;
  else return (s[i] - t[i]);
}
```

® Esercizio 2

Confrontare i primi n caratteri di due stringhe in ordine lessicografico

```
int strncmp_r(char *s, char *t, int n){
  if (n==1 || *s=='\0' || *s != *t)
      return *s - *t;
  else return(strncmp_i(s+1, t+1, n-1));
}
```

Esercizio 3

Dati una stringa s e un carattere c, restituire il puntatore alla prima occorrenza di c in s (o NULL)

```
char* my_strchr (char *s, int c) {
  for ( ; *s!='\0' && *s!=c; s++);
  return ( (*s==c) ? s : NULL );
}
```

® Esercizio 3

Dati una stringa s e un carattere c, restituire il puntatore alla prima occorrenza di c in s (o NULL)

```
char* my_strchr (char *s, char c) {
  if (*s == '\0') return NULL;
  else if (*s == c) return s;
  else return my_strchr(s+1,c);
}
```

® Esercizio 3

Dati una stringa s e un carattere c, calcolare in un parametro il puntatore alla prima occorrenza di c in s

```
void my_strchr( char *s, char c,
                char **ptr ) {
  if (*s == '\0') *ptr = NULL;
  else if (*s == c) *ptr = s;
  else my_strchr(s+1,c,ptr);
}
```

Esercizio 4

Dati una stringa s e un carattere c, restituire il puntatore all'ultima occorrenza di c in s (o NULL)

```
char* last_strchr (char *s, int c) {
  int i;
  for (i=strlen(s)-1; i>=0 && s[i]!=c; i--);
  return ( (s[i]==c) ? &s[i]  : NULL );
}
```

® Esercizio 4

Dati una stringa s e un carattere c, restituire il puntatore all'ultima occorrenza di c in s

```
char* last_strchr (char *s, char c) {
  if (*s=='\0') return NULL;
  if (*s!=c) return last_strchr(s+1,c);
  else return ((last_strchr(s+1,c)!=NULL)?
                last_strchr(s+1,c) : s);
}
```

Efficienza? Attenzione alle chiamate ricorsive!

® Esercizio 4

Dati una stringa s e un carattere c, restituire il puntatore all'ultima occorrenza di c in s

```
char* last_strchr (char *s, char c) {
  char *temp;
  if (*s=='\0') return NULL;
  temp = last_strchr(s+1,c);
  if (*s!=c) return temp;
  return ((temp!=NULL) ? temp : s);
}
```

Esercizi

Dare sia un'implementazione iterativa che una ricorsiva

- Data una stringa s che rappresenta un numero intero, restituire il valore corrispondente
- Nell'implementazione ricorsiva, cercare di risolvere il problema nel modo più efficiente possibile: in particolare, cercare di effettuare UNA sola passata sulla stringa.