[Γα=Ω5]

Strings y vectores de C

Por Ariel Parra

caracteres

Un caracter es una unidad de información que corresponenden a un símbolo,digito,puntuaciones,signo,grafema,grafo,garabato o los más conocidos que son las letras, las cuales forman las palabras que usamos y conocemos

En C los caracteres no son más que un conjunto de numeros para representar un caracter a travez de algun tipo de codificación, por defecto C++ utiliza la decodificación de ASCII (American Standard Code for Information), aunque también soporta UNICODE/UTF-8 (En Windows tienes que declarar el uso explicitamente con la función `SetConsoleOutputCP(CP_UTF8)`).

la sintaxis básica de declaración de un char debe ser con un caracter entre comillas simples `''`, por ejemplo:

```
char c;
char c = 'a';
char c{'a'};
```

ASCII

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

vectores en C

En C, un vector (o arreglo) es una colección de elementos del mismo tipo, almacenados en ubicaciones de memoria contiguas. Se accede a los elementos del vector mediante un índice, utilizando el operador `[]`.

```
int arr[5]; // Declara un vector con capacidad para 5 enteros
arr[0] = 1; // Asigna el valor 1 al primer elemento
```

Recorrer un vector en C:

```
for (int i = 0; i < 5; i++) {
    cout<<("Elemento en arr[" << i << "] = " << arr[i]);
}
```

Strings en C

Al conjunto de caracteres se les llama Strings (también conocidos como cadenas o vectores/arrgelos de char) y se declaran con comillas dobles `""`,en esencia son vectores de caracteres que terminan con un carácter nulo (`\0`).

```
char c[] = "valor";
char c[6] = "valor"; // El compilador añade automáticamente '\0'
char c[] = {'v', 'a', 'l', 'o', 'r', '\0'};
char c[6] = {'v', 'a', 'l', 'o', 'r', '\0'};
```

Matrices

Las matrices en C son arreglos multidimensionales, lo que significa que son arreglos de arreglos. La más común es la matriz bidimensional, que se puede imaginar como una tabla de filas y columnas.

```
int matriz[3][4];
```

La inicialización de una matriz se puede hacer al momento de la declaración, utilizando llaves {} para agrupar los elementos de cada fila:

```
int matriz[3][4] = {
    {1, 2, 3, 4},
    {5, 6, 7, 8},
    {9, 10, 11, 12}
};
```

Recorrer una Matriz

Generalmente, se utilizan bucles anidados para recorrer todos los elementos de una matriz. Aquí hay un ejemplo de cómo imprimir todos los elementos de una matriz bidimensional:

```
for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 4; j++) {
        cout<< matriz[i][j];
    }
    cout<< "\n";//salto de línea
}
```

Aritmética de Caracteres

- Convertir Minúsculas a Mayúsculas

```
char minuscula = 'b';
char mayuscula = minuscula - ('a' - 'A');
```

- Convertir Mayúsculas a Minúsculas

```
char mayuscula = 'B';
char minuscula = mayuscula + ('a' - 'A');
```

- Convertir Caracter a Entero

```
char numC = '1'; int numI = numC - '0'; // numI será 1
```

- Convertir Entero a Caracter

```
int numI = 1; char numC = numI + '0'; // numC será '1'
```

Funciones de Caracteres y Strings en C

<cctype>	<cstdlib>	<cstring>
isalpha(char)⮭	atoi(cstring)⮭	strcpy(string-dest, string-origen)⮭
isdigit(char)⮭	atof(cstring)⮭	strcat(string-dest, string-origen)⮭
isupper(char)⮭	atol(cstring)⮭	strncat(string-dest, string-origen, int)⮭
islower(char)⮭	strtol(cstring, NULL, 0)⮭	strcmp(string, string-comparar)⮭
tolower(char)⮭	~~itoa(int, cstring, 10)⮭~~	strncmp(string, string-comparar, int)⮭
toupper(char)⮭	sprintf(cstring,"%i",int)⮭	strlen(string)⮭

Problemas

Strings:

- **59A** Word
- **281A** Word Capitalization

Vectores:

- **734A** Anton and Danik
- **263A** Beautiful Matrix

Referencias

- Portfolio Courses. (2022). *String In Char Array VS. Pointer To String Literal | C Programming Tutorial* [video]. Recuperado de https://youtu.be/Qp3WatLL_Hc?si=ZClAvUhUhOqwFjNZ
- Microsoft. (2022). *String and character literals (C++)*. Recuperado de https://learn.microsoft.com/en-us/cpp/cpp/string-and-character-literals-cpp?view=msvc-170
- cppreference. (s.f.). *Null-terminated byte strings*. Recuperado de https://en.cppreference.com/w/cpp/string/byte