Estructuras de Datos Estáticas

Rodolfo Christian Catunta Uturunco (Elemental Bolivia)

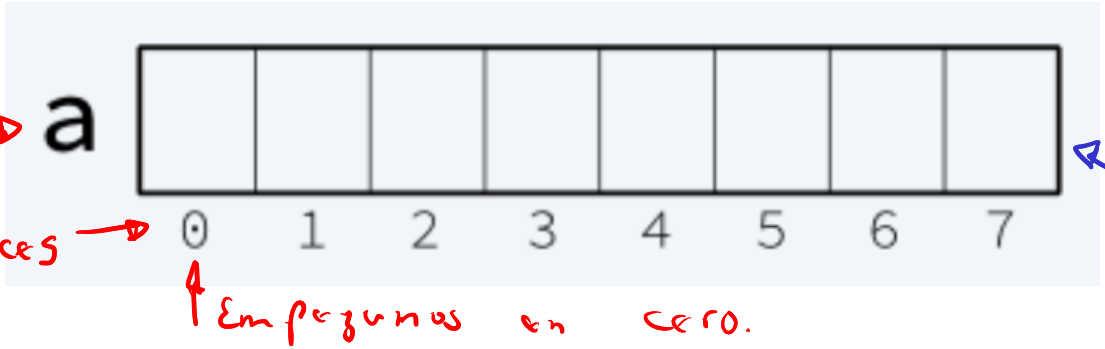
28 de febrero de 2025

Contenido

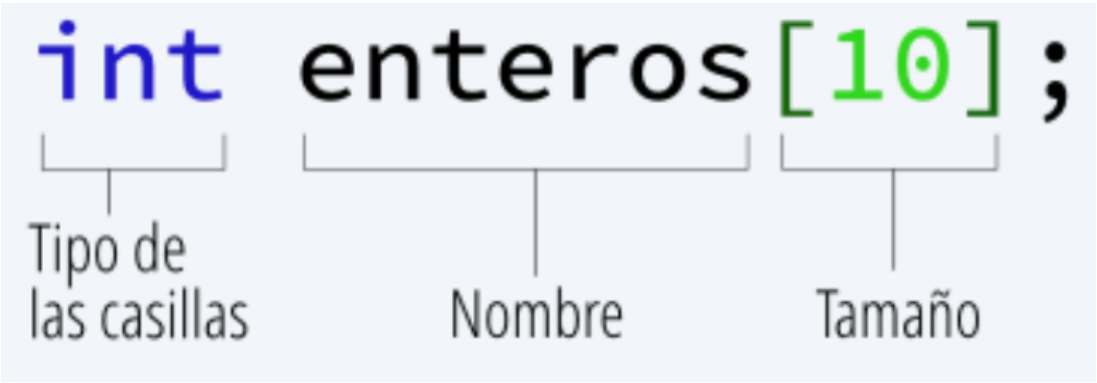
Contenido

Arreglos Unidimensionales

Los arreglos en C++ son una colección de variables del mismo tipo, accesibles a través de un índice. Llamaremos casilla a cualquiera de las variables individuales que componen un arreglo. Un arreglo se puede ver como



Para crearlos se utiliza:



Declaracion

```
1 int main(){
2   int enteros[10]; //Declara un arreglo de tamanio 10 de tipo
      entero
3   float numeros[4]; //Declara un arreglo de tamanio 4 de tipo
      flotante
4   int tam;
5   cin>>tam;
6   int datos[tam]; //Declarara un arreglo de tamanio tam de
      tipo entero
7 }
```

Uso

```
1 int main(){
2   int enteros[10]; //Declara un arreglo de tamanio 10 de tipo
      entero
3   // Al inicio los arreglos comienzan sin valores en sus
      casillas
4   //Para acceder a una casilla hacemos
5   enteros[0] = 8; // Ahora la casilla 0 del arreglo tendra el
      valor de 8
6   enteros[1] = -7; // Ahora la casilla 1 del arreglo tendra
      el valor de -7
7   cout<<enteros[0]<<endl;
8 }
```

Impresión y recorrido

```cpp
int main(){
  int N[7]; // Arreglo de tamanio 7
  //Rellenamos el arreglo, esta es una forma no optima de realizar el rellenado, tambien toma en cuenta que puedes utilizar un for para llenarlo.
  N[0] = 4;
  N[1] = 6;
  N[2] = -1;
  N[3] = 0;
  N[4] = 9;
  N[5] = -14;
  N[6] = 10;
  //Recorremos e imprimimos el arreglo
  for(int i=0;i<7;i++){//El final siempre debe ser el tamanio del arreglo
    cout<<"N["<<i<<"] = "<<N[i]<<endl;
  }
}
```

Ordenamiento

- Burbuja
- Selección
- Mezcla (Merge)
- Quick

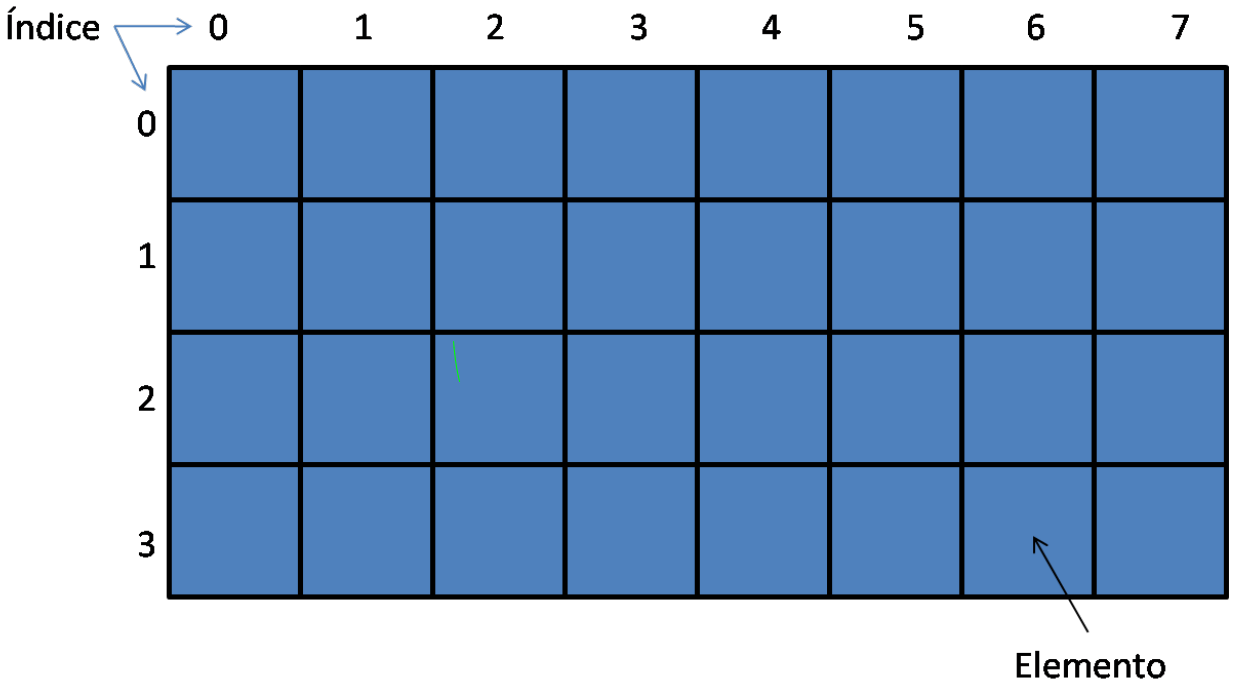
```cpp
#include <iostream>
#include <algorithm> // Libreria necesaria para ordenar

using namespace std;

int main(){
  // Creamos el arreglo
  int arr[10] = {3,5,1,0,8,9,2,6,4,7};
  // Funcion para ordenar el arreglo
  sort(arr, arr+10);

  return 0;
}
```

Son direcciones de memoria.

Contenido

Arreglos Bidimensionales

Una matriz es un arreglo de 2 dimensiones. Lo cual genera una estructura en forma de tabla como en la imagen.



tensor

Lo Arreglo

3D

Una matriz a diferencia de un arreglo tiene 2 índices en lugar de 1, como se muestra en la figura. Cabe también recalcar que cada una de las casillas es un elemento de la matriz.

Arreglos Bidimensionales

Las posiciones de una matriz se acceden primero por la fila y luego por la columna. Las filas son horizontales y las columnas verticales.

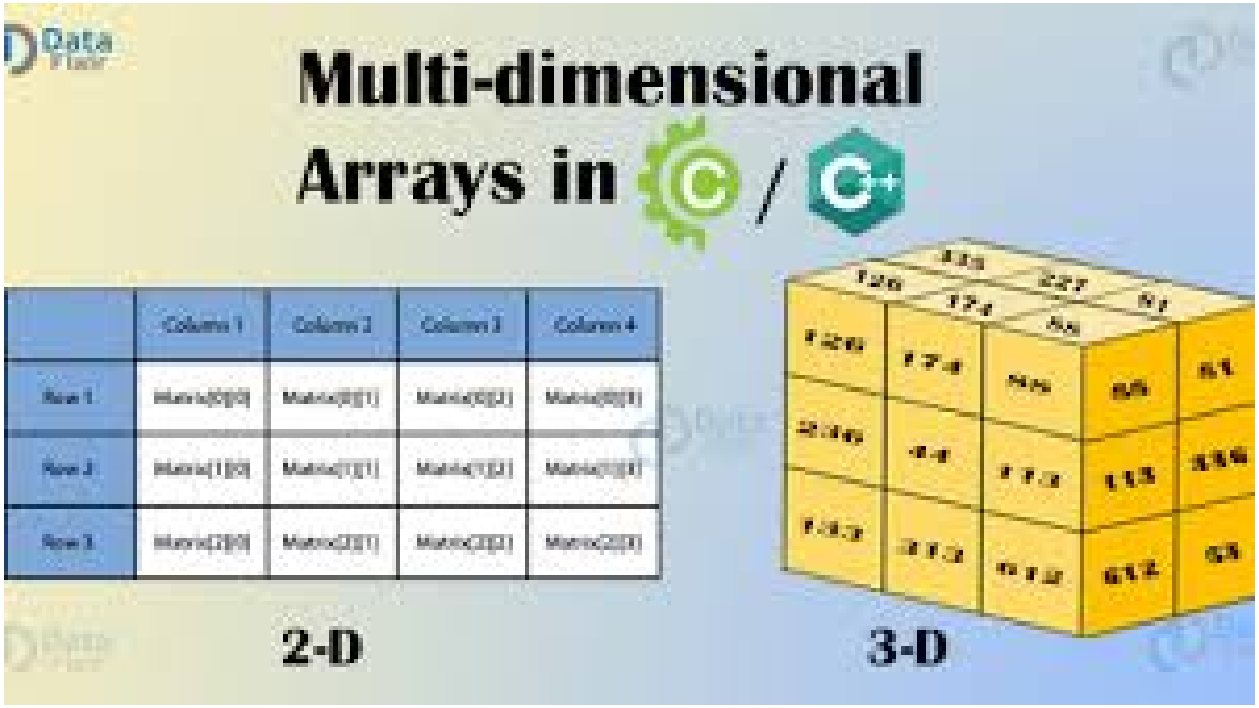
fila

col

	Column 0	Column 1	Column 2	Column 3
Fila Row 0	x[0][0]	x[0][1]	x[0][2]	x[0][3]
Fila Row 1	x[1][0]	x[1][1]	x[1][2]	x[1][3]
Fila Row 2	x[2][0]	x[2][1]	x[2][2]	x[2][3]

Arreglos Multidimensionales

Es posible realizar arreglos de la cantidad de dimensiones que se desee, en la imagen inferior se muestra como se vería una matriz de 3 dimensiones.



Recorrido de una Matriz

```
1  //Llena a la matriz de 0s
2  int main(){
3    int matriz[4][8]; // 4 filas y 8 columnas
4    //Recorremos la matriz
5    for(int i=0;i<4;i++){//Primer ciclo recorre las filas
6      for(int j=0;j<8;j++){//Segundo ciclo recorre las columnas
7        matriz[i][j]=0;
8      }
9    }
10 }
```

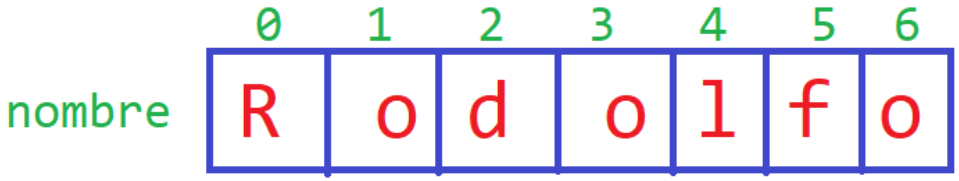
Recorrer una matriz desde el teclado

```
1  int main(){
2    int num_filas, num_columnas;
3    cin>>num_filas>>num_columnas; // Pedimos el numero de filas
       y columnas.
4    int matriz[num_filas][num_columnas]; // Creamos la matriz
       de numeros
5
6    // Peticion de los numeros que tendra la matriz
7    for(int i=0;i<num_filas;i++){// Filas
8      for(int j=0;j<num_columnas;j++){// Columnas
9        cin>>matriz[i][j];
10     }
11   }
12 }
```

Contenido

Cadenas de caracteres

Las cadenas, también conocidas como strings son llamadas así, porque almacenan una cadena de caracteres en su interior.



Un string es similar a un arreglo de caracteres, solo que tiene aún mas características, las cuales serán analizadas en un módulo especializado en Strings.

Recorrido de un string

```
1 int main(){
2   string curso="CreadoresDigitales2021";
3   for(int i=0;i<curso.size();i++){
4     char letra=curso[i];
5     cout<<letra<<" ";
6   }
7 }
```

Código ASCII

En programación resulta de gran utilidad el uso de código ASCII para manipular cadenas. La Tabla ASCII nos muestra todos los caracteres imprimibles que se pueden colocar en un string.

Caracteres ASCII de control

00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc.vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sínc)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

Caracteres ASCII imprimibles

32	espacio	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_		

ASCII extendido

128	Ç	160	á	192	└	224	Ó
129	ü	161	í	193	┴	225	ß
130	é	162	ó	194	┬	226	Ô
131	â	163	ú	195	├	227	Ò
132	ä	164	ñ	196	─	228	õ
133	à	165	Ñ	197	┼	229	Õ
134	å	166	ª	198	ã	230	µ
135	ç	167	º	199	Ã	231	þ
136	ê	168	¿	200	╚	232	Þ
137	ë	169	®	201	╔	233	Ú
138	è	170	¬	202	╩	234	Û
139	ï	171	½	203	╦	235	Ù
140	î	172	¼	204	╠	236	ý
141	ì	173	¡	205	═	237	Ý
142	Ä	174	«	206	╬	238	¯
143	Å	175	»	207	¤	239	´
144	É	176	░	208	ð	240	≡
145	æ	177	▒	209	Ð	241	±
146	Æ	178	▓	210	Ê	242	‗
147	ô	179	│	211	Ë	243	¾
148	ö	180	┤	212	È	244	¶
149	ò	181	Á	213	ı	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	¸
152	ÿ	184	©	216	Ï	248	°
153	Ö	185	╣	217	┘	249	¨
154	Ü	186	║	218	┌	250	·
155	ø	187	╗	219	█	251	¹
156	£	188	╝	220	▄	252	³
157	Ø	189	¢	221	¦	253	²
158	×	190	¥	222	Ì	254	■
159	ƒ	191	┐	223	▀	255	nbsp

Uso de código ASCII en C++

```
1 int main(){
2   for(int i=0;i<256;i++){
3     char caracter = char(i);
4     int ascii = int(caracter);
5     cout<<"El elemento "<<ascii<<" de la tabla ASCII es "<<
      caracter<<endl;
6   }
7 }
```

Contenido

memset

La función memset es muy útil para inicializar en ciertos valores a arreglos o cadenas. Sin embargo se debe ser precavido en su uso como se muestra en el siguiente ejemplo.

```cpp
#include <bits/stdc++.h> // Para garantizar el uso de memset en cualquier compilador
using namespace std;
int main()
{
  // memset(nombre de estructura, valro, tamanio en bytes)
  int a[5];
  // todos los elementos de 0
  memset(a, 0, sizeof(a));
  // todos los elementos de -1
  memset(a, -1, sizeof(a));
  // No funcionara
  memset(a, 5, sizeof(a)); // INCORRECTO
}
```