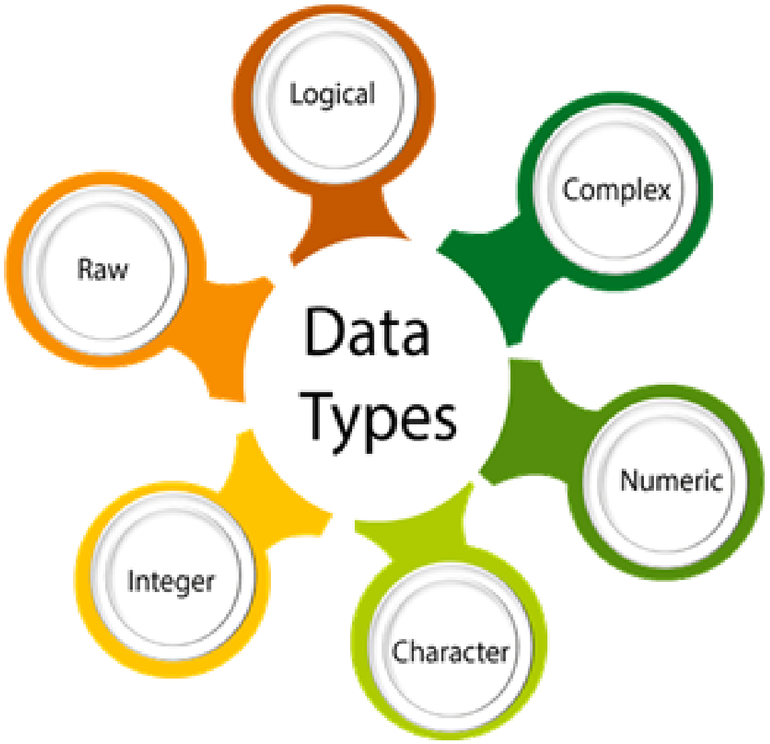
¡¿0 - 1 = 255 ?!

Fe de erratas

En el temario enviado, el título de la tercera lección es "¿5 - 1 = 14 ?", este titulo es erróneo, el título correcto para esta lección es "¿0 - 1 = 255?".

Tipos de datos



Si el microcontrolador únicamente trabaja con 1 y 0, ¿cómo sabe cuando es negativo o decimal? Para esto existen los Tipos de Datos que es la manera en la cuál nosotros le indicamos a nuestro microcontrolador cómo debe "interpretar" la serie de 1 y 0s que le estamos programando o enviando.

BOOL

Tipo de dato	Signado	Bits	Rango
bool	NO	8	TRUE y FALSE

Este tipo de dato sólo se puede trabajar con dos valores 'TRUE' y 'FALSE' que nos representarían un 1 y 0 lógico, es decir apagado y encendido. Se usa comunmente para la toma de decisiones dicotómicas o binarias, para colocar estados y validar condiciones.(2)

BYTE

Tipo de dato	Signado	Bits	Rango
byte	NO	8	0 a 255

En informática se le conoce como 'byte' a la agrupación de 8 bits. Esta variable es no signada, es decir, sólo puede representar valores positivos y enteros de un rango de 0 a 255. Se usa comunmente para el manejo de puertos o direcciones de 8bits.(3)

CHAR

Tipo de dato	Signado	Bits	Rango
char	SÍ	8	-127 a 127

Este tipo de dato también es de 8 bits, con la peculiaridad de que es signado, es decir, sus valores numéricos pueden representar valores negativos. La página oficial de Arduino recomienda usarlo únicamente para manejar caracteres alfanuméricos en código ASCII.(4)

Código ASCII

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)

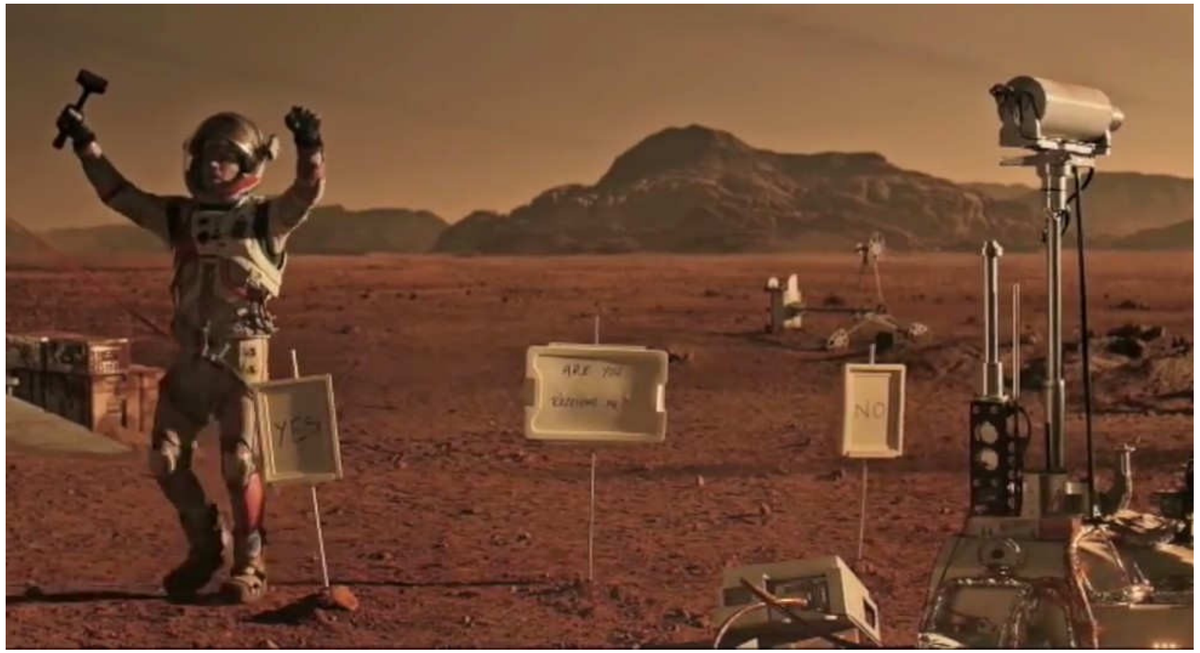
Caracteres ASCII imprimibles					
32	espacio	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	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

Elaborado por Miguel Angel López

Dato curioso

En la película 'The Martian' (2015) el astronauta Mark Watney (Matt Damon) usa el código ASCII para establecer contacto con la NASA desde Marte.



FLOAT

Tipo de dato	Signado	Bits	Rango
float	SÍ	32	3.4E+38 a -3.4E+38

Este dato se usa para el manejo de números decimales, tales como el número PI, el número e, almacenar resultados de divisiones o relaciones entre números. Arduino recomienda no usarlo a menos que sea necesario.(5)

DOUBLE

Tipo de dato	Signado	Bits	Rango
double	SÍ	32	3.4E+38 a -3.4E+38

Este tipo de dato también se usa para trabajar con número decimales o irracionales, la diferencia, como su nombre lo indica, es que se define como un dato de punto flotante de doble precisión, es decir, puede manejar el doble de decimales que el tipo float.(6)

INT

Tipo de dato	Signado	Bits	Rango
int	SÍ	16	-32,768 a 32,767

Traducido al español como 'entero', es el tipo de dato predilecto por la mayoría de programadores, este tipo de dato unicamente toma valores enteros positivos y negativos. Es comunmente usado para contadores, temporizadores, comparaciones y rutinas.(7)

LONG

Tipo de dato	Signado	Bits	Rango
long	SÍ	32	-2,147,483,648 a 2,147,483,647

Un long se puede considerar como un int 'largo', es decir un entero con más valores numéricos, en el rango de 2 millones. Se usa comunmente para temporizadores de tiempo prolongado, para cálculos matemáticos robustos o métodos numéricos. (8)

Tipo de dato	Signado	Bits	Rango
bool	NO	8	TRUE y FALSE
byte	NO	8	0 a 255
char	SÍ	8	-127 a 127
double	SÍ	32	3.4E+38 a -3.4E+38
float	SÍ	32	3.4E+38 a -3.4E+38
int	SÍ	16	-32,768 a 32,767
long	SÍ	32	-2,147,483,648 a 2,147,483,647

¡¿0 - 1 = 255 ?!

Los números signados tomando el bit más significativo (MSB) o el de mayor valor, para interpretarlo como signo negativo. Dependiendo la arquitectura, el microcontrolador puede usar el complemento a 1 o complemento a 2 para interpretar un número negativo. En este caso, Arduino usa el complemento a 2.

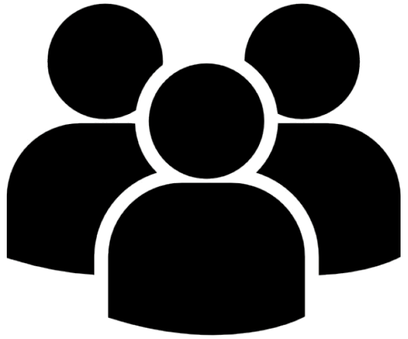
¡¿0 - 1 = 255 ?!


Código binario, inverso y complemento
Número
-1
Cantidad de dígitos binarios
8
CALCULAR
Rango
[-128,127]
Código Inverso (Complemento a uno)
11111110
Código complemento (Complemento a dos)
11111111

Ejemplos

Necesitamos contar el número de personas que entran y salen de una tienda mediante un sensor de movimiento.

¿Que tipo de dato sería mejor utilizar?



Respuesta

int, porque no podría haber personas decimales

Ejemplos

Se necesita calcular el Indice de Masa Corporal (IMC) de una persona con datos de un báscula electrónica y un sensor ultrasónico mide la altura. El IMC se define como peso/altura x altura.

¿Qué tipo de dato se debe usar?

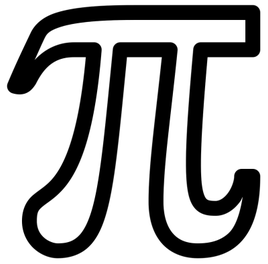
Respuesta

float, porque la división nos dará resultado con punto decimal

Ejemplos

Deseamos calcular el número pi con la mayor cantidad de decimales posibles a nuestro alcance.

¿Qué tipo de dato se debe usar?

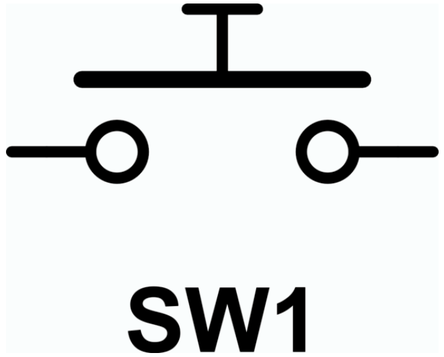


Respuesta

double, porque necesitamos un número decimal con mayor precisión que un float

Ejemplos

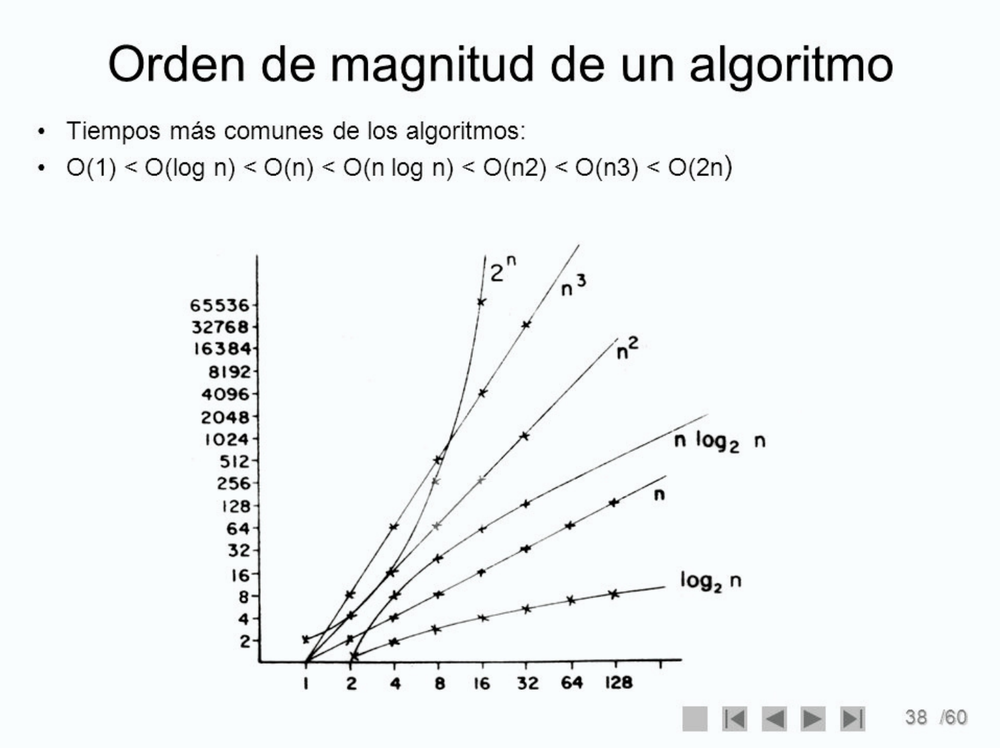
Debemos sensar el estado de un botón que sólo tiene dos estados lógicos, encendido y apagado mediante un puerto de entrada digital. ¿Qué tipo de dato se debe usar?



Respuesta

bool, debido a que sólo vamos a trabajar con el estado TRUE y FALSE

Ventajas



La correcta selección de los tipos de datos para nuestros programas hace un uso eficiente de la memoria, por lo tanto, nuestro algoritmo se ejecutará más rápido y con menos necesidad de memoria. A esto se le conoce como "complejidad computacional".

Tiempo de descanso y de restablecer humanidad.(:

Importante

Importante

Las operaciones aritmeticas en C, únicamentes son válidas entre los mismos tipos de datos, es decir, para hacer una suma, multiplicación o división, ambos elementos de la operación, tienen que ser del mismo tipo de dato.

int + int = int

float / float = float

long * long = long

int + char = ERROR

¿Cómo cambiamos de un tipo de dato a otro?

En el argot de la programación se le llama "castear" a la acción de convertir un tipo de dato a otro. El único requisito es que el número a convertir "quepa" dentro del tipo de dato que se quiera.(1)

Conversion

(unsigned int)
(unsigned long)
byte()
char()
float()
int()
long()
word()

¿Cómo lo hacemos?

Para convertir un tipo de dato byte a int colocamos la siguiente sentencia:

```
int variableNueva = int(variableTipoByte);
```

(9)

Otra forma más "común"

Si quisiera calcular la velocidad de un objeto tendría la siguiente expresión:

Operador para divisón aritmética
/

```
float velocidad = metros/float(segundos);
```

Tipo de dato de la variable nueva

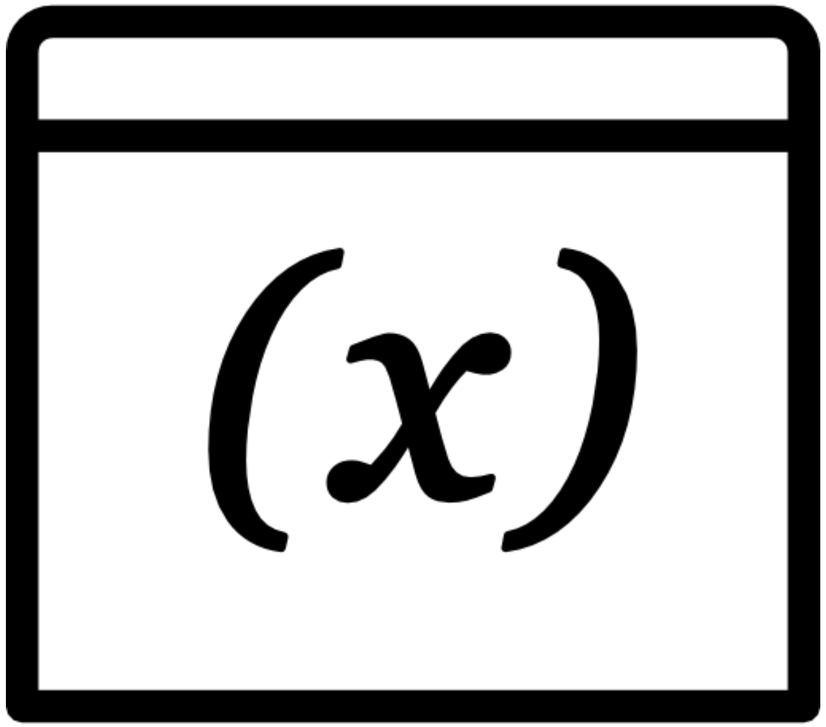
Nombre de la nueva variable

Suponemos que metro ya es float

Nombre de la variable que quiero convertir

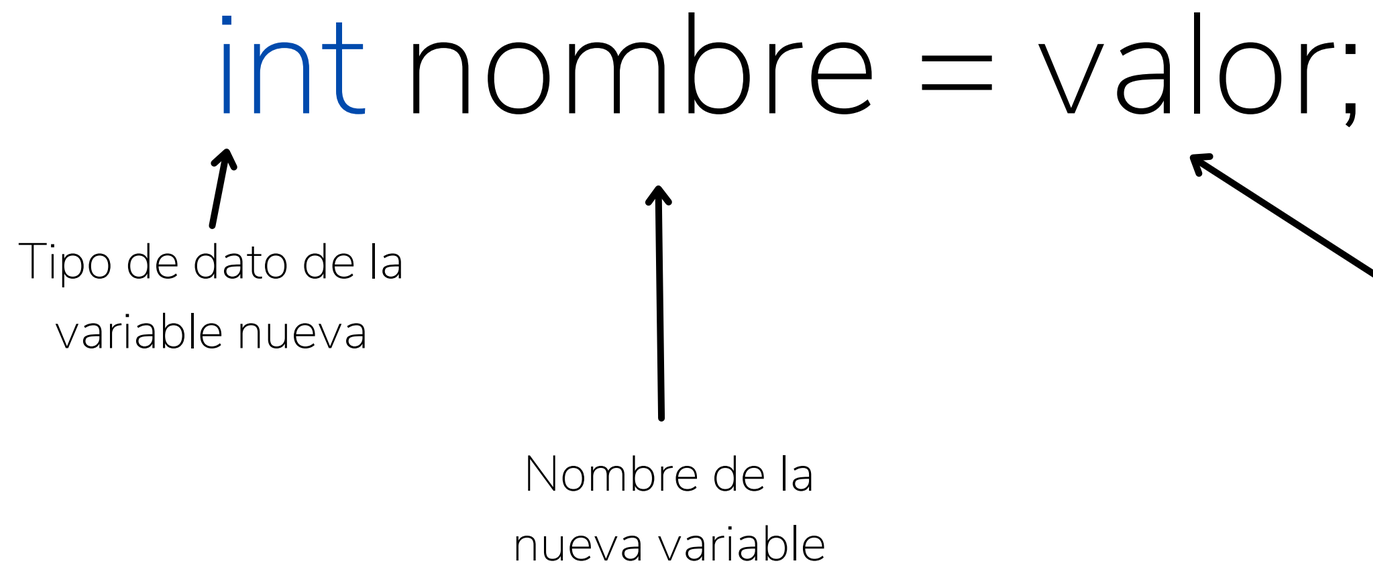
(9)

¿Y qué es una "variable"?



Una variable es un espacio en la memoria RAM que reservamos o "apartamos" al momento de programar para poder hacer uso de ella durante la ejecución del programa. Es decir reservamos espacio para almacenar o "guardar" algún valor que desconocemos.

¿Cómo se declara un variable?

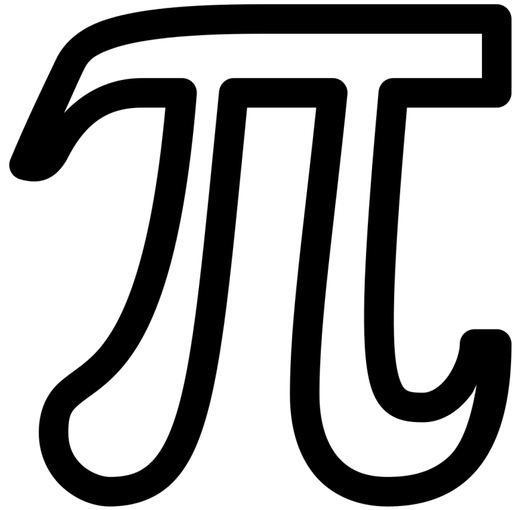


Ejemplos

```
float velocidad = 7.68;
double eje= 5.23464736759;
int personas;
byte puertoD = 32;
```

Constantes

Al igual que una variable, al declarar una constante en el IDE, vamos a reservar el espacio en memoria de un dato que vamos a almacenar, en este caso, sabemos el valor del dato y sabemos de antemano que el valor no va a cambiar durante la ejecución del programa.(10)



Ejemplos

```
const float gravedad = 9.81;
const double pi= 3.14159265359;
const int sensores=4;
const char letra = 'a';
```

Constantes

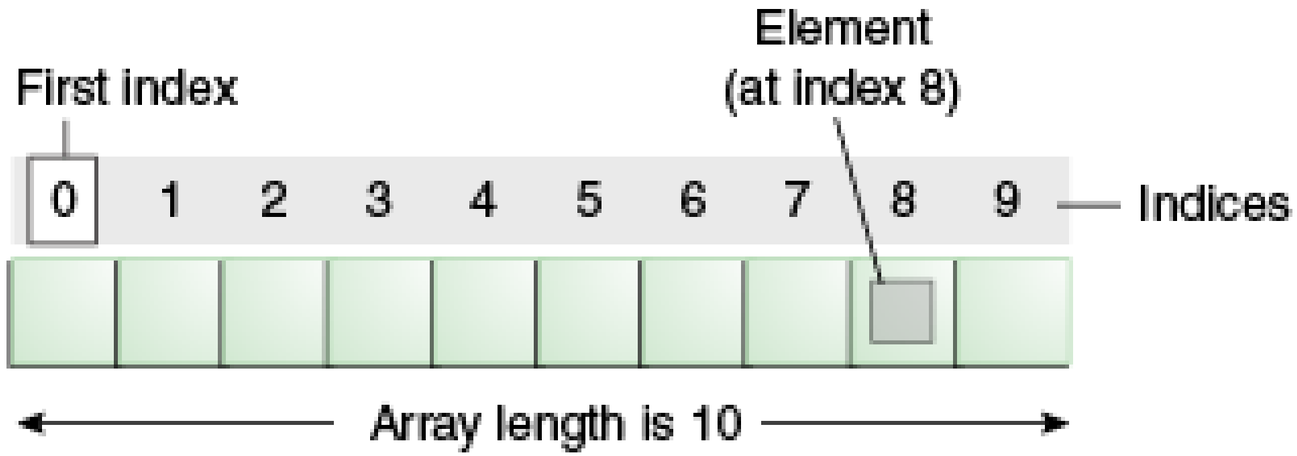
También se suele usar la palabra reservada #define para la declaración de variables, generalmente esta palabra sólo se usa para definir el nombre de pines del microcontrolador y en vez de usar números, usar palabras que relacionen mejor el circuito con el código.(11)

```
#define constantName value
```

```
#define motor 3
```

Arreglos

Un arreglo es un conjunto o "colección" de datos del mismo tipo que se encuentran "indexeados", es decir, tienen un índice numérico para identificar el valor en determinada posición del arreglo.(12)



¿Cómo se hace un arreglo?

```
int misEnteros[6];
int misPins[] = {2, 4, 8, 3, 6};
int mySensVals[5] = {2, 4, -8, 3, 2};
char mensaje[6] = "hello";
```

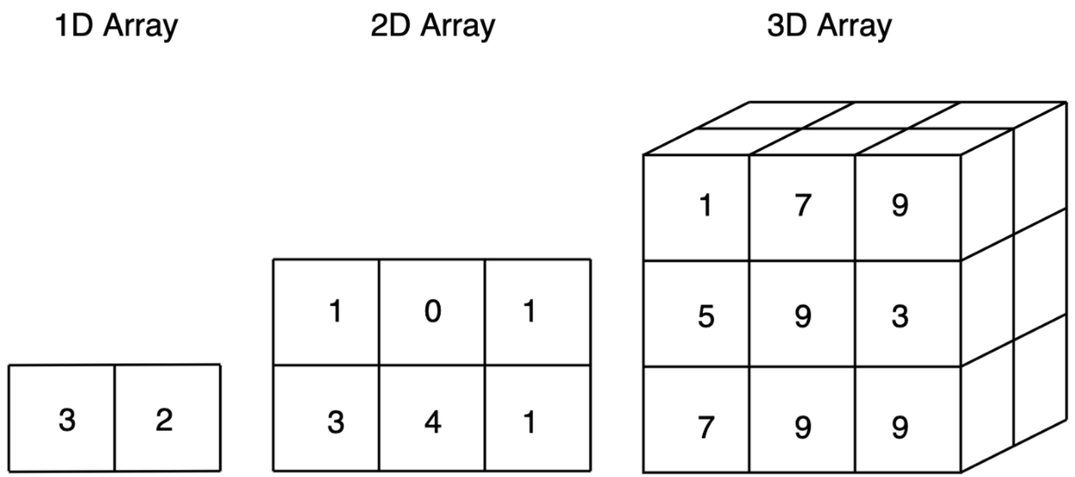
Conozco el tamaño del arreglo, pero desconozco su contenido

Conozco el tamaño del arreglo y su contenido

A un arreglo de chars se le conoce como "String" o cadena de caractéres.(13)

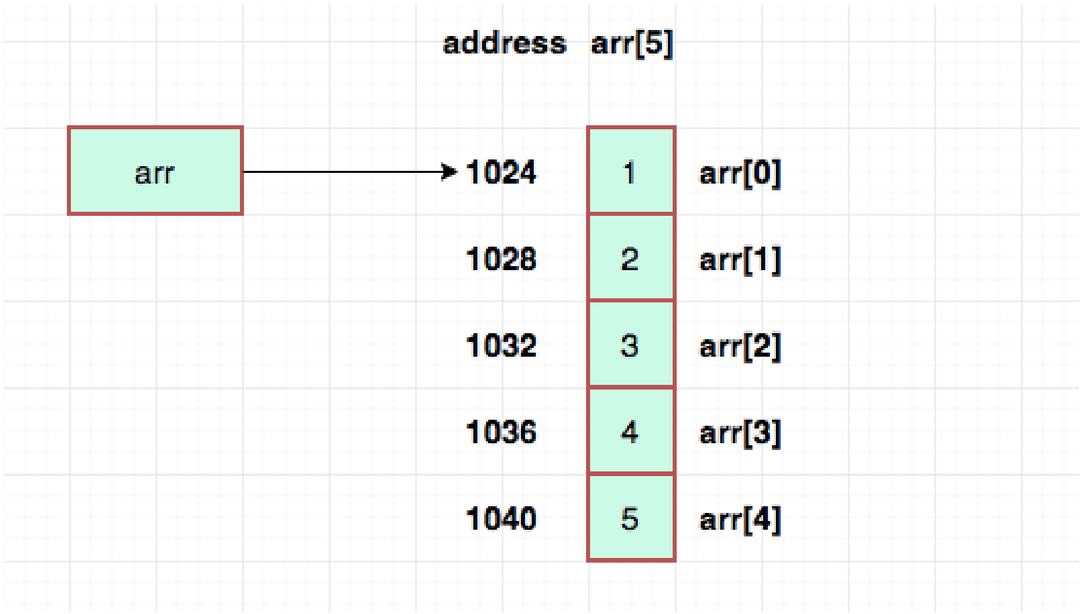
Arreglos multidimensionales

```
int arreglo1D[2] = {3,2};
int arreglo2D[2][3] = {{1,0,1},{3,4,1}};
int arreglo3D[3][3][2] = {{{1,7,9},{5,9,3},{7,9,9}},{...}};
```



Ventajas de usar arreglos

La principal ventaja de un arreglo es mantener más el órden a la hora de trabajar con datos "agrupados" como una señal de voz o datos de un sensor que tenga que estar haciendo mediciones constantes, sobre todo, aumenta la velocidad de procesamiento de estos datos debido a que cada dato se almacena de manera contigüa o "vecina" al dato anterior y posterior.



Enlaces de interés

Código ASCII

https://elcodigoascii.com.ar/

Conversor binario, inverso y complemento

https://es.planetcalc.com/747/

Complemento 1 y 2

https://www.youtube.com/watch?v=mG0SuB2XohQ

Fuentes

1. https://www.arduino.cc/reference/en/
2. https://www.arduino.cc/reference/en/language/variables/data-types/bool/
3. https://www.arduino.cc/reference/en/language/variables/data-types/byte/
4. https://www.arduino.cc/reference/en/language/variables/data-types/char/
5. https://www.arduino.cc/reference/en/language/variables/data-types/float/
6. https://www.arduino.cc/reference/en/language/variables/data-types/double/
7. https://www.arduino.cc/reference/en/language/variables/data-types/int/
8. https://www.arduino.cc/reference/en/language/variables/data-types/long/
9. https://www.arduino.cc/reference/en/language/variables/conversion/intcast/
10. https://www.arduino.cc/reference/en/language/variables/variable-scope-qualifiers/const/
11. https://www.arduino.cc/reference/en/language/structure/further-syntax/define/
12. https://www.arduino.cc/reference/en/language/variables/data-types/array/
13. https://www.arduino.cc/reference/en/language/variables/data-types/string/