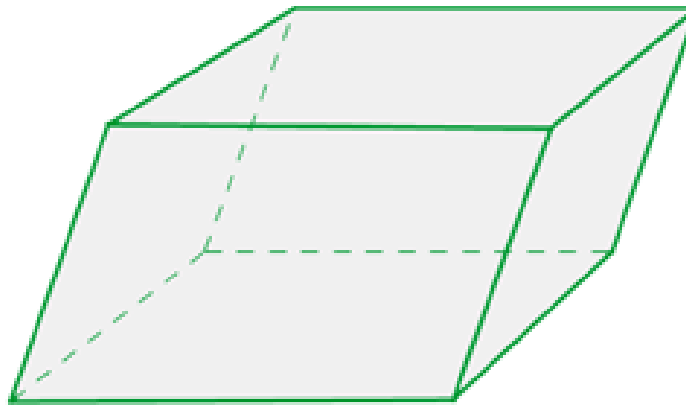


PROGRAMACIÓN

Unidad 3: Lenguaje C .Concepto de tipo de dato.
Tipos, operadores, expresiones, calificadores,
conversiones

Comenzamos resolviendo el siguiente problema...

Diseñar y escribir un algoritmo que calcule el volumen de un paralelepípedo. Implemente en C.



Analizamos la solución propuesta

- El código del ejemplo encierra varios conceptos:
 - Variables.
 - Declaraciones.
 - Asignaciones.
 - Expresiones aritméticas.
 - Salida con formato.

Variables

- Una variable es un objeto cuyo valor cambia durante la ejecución del programa, que tiene un **nombre** y ocupa una cierta posición de memoria.
- Todas las variables que se usan deben ser declaradas. La idea de ello es que se le avisa al compilador de las propiedades de la variable en cuestión. Esto es de los valores que pueden asumir y de las operaciones permitidas.

Declaración de una Variable

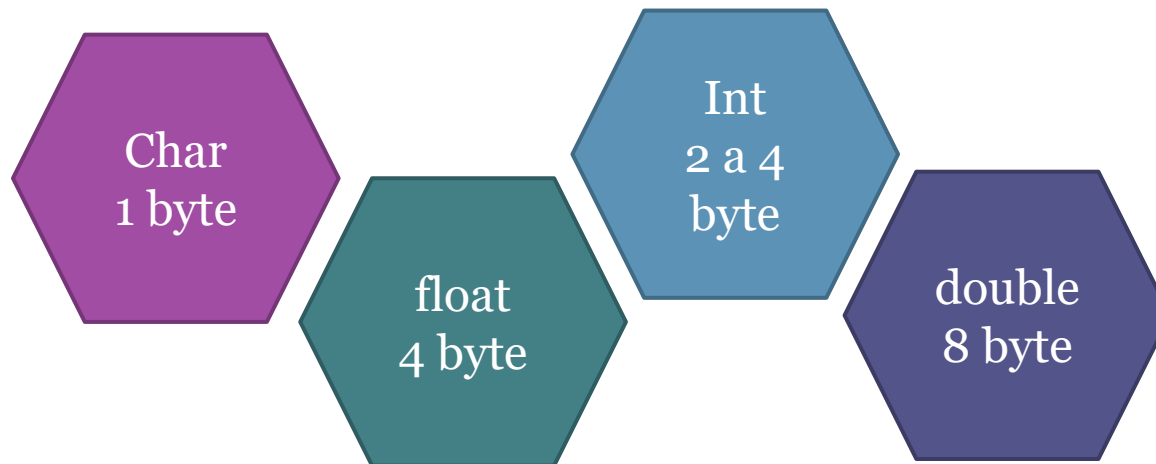
- La declaración se hace usando una **palabra reservada** del lenguaje.
- En el ejemplo, **int** es la palabra reservada, que le avisa al compilador que ha de trabajar con enteros.
- Una palabra reservada es una palabra con significado específico, predefinido por el lenguaje que se usará.

¿Qué valores puede asumir una variable int?

- Los valores que puede tomar una variable entera están en el **rango -32768 hasta 32767**.
- Y surge la pregunta **¿Por qué un entero tiene límites de rango tan extraños?**. Esto tiene que ver con la forma en que la computadora almacena los números. Sabemos que los almacena en forma binaria en vez de nuestro familiar sistema decimal. Un entero tiene **16 dígitos binarios o bits** ($2^{16} = 32768$), uno de los cuales se usa para el signo.

Tipos y tamaños de datos

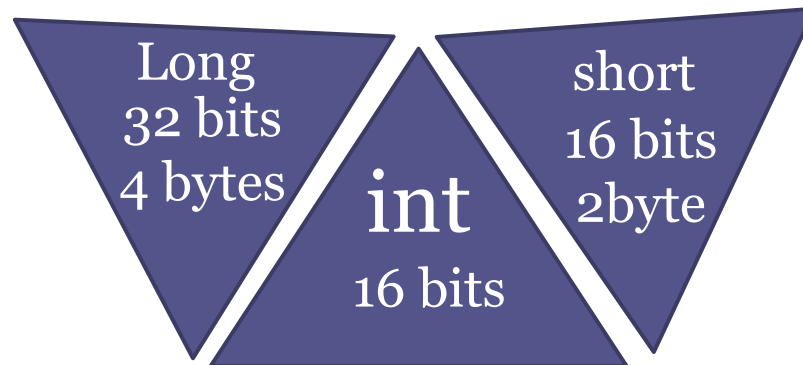
- Tipos de datos básicos (simples o atómicos)



Cada compilador puede seleccionar los tamaños, para su propio hardware

Tipo de dato entero Tamaño, calificadores

3 Tamaños para los enteros:



Cada compilador puede seleccionar libremente los tamaños apropiados para su propio hardware, sujeto sólo a restricción de que los short e int son por lo menos de 16 bits, los long son por lo menos de 32 bits.

Se debe cumplir : short < int < long

Aritmética módulo 2

Solo existen

10

tipos de personas...

... Las que entienden binario
y las que no.

Aritmética módulo 2

2^n , n : número de bits.

8 bits = 1 byte

$n = 2$ bytes

$2^{16} \rightarrow 65536$

$2^{16} - 1 \rightarrow -32768, 32767$

Otros calificadores

unsigned

signed

- Los números unsigned obedecen a la aritmética módulo 2^n , con n número de bits.
- Se puede aplicar tanto a **char** como a **int**
- Si los char son de 8 bits, , las variables unsigned char tienen valores entre 0 y 255.

Tabla: int

int	16	-32768 a 32767
unsigned int	16	0 a 65535
signed int	16	-32768 a 32767
short int	16	-32768 a 32767
unsigned short int	16	0 a 65535
signed short int	16	-32768 a 32767
long int	32	-2147483648 a 2147483647
signed long int	32	-2147483648 a 2147483647
unsigned long int	32	0 a 4294967295
long	32	-2147483648 a 2147483647
unsigned long	32	0 a 4294967295

Constantes y tipo asociado

- A definir constantes de tipo numérico se puede indicar el calificador asociado.
- Un entero long se escribe con una l o L terminal
- Las constantes sin signo con u o U.
- También se pueden definir constantes de carácter y de cadena.

Operadores

- Aritméticos Binarios: $+$ $-$ $*$ $/$ $\%$
- De Relación : $<$ $<=$ $>$ $>=$
- De Igualdad : $==$ $!=$
- Lógicos: $\&\&$ $||$



El módulo o residuo solo se aplica a los valores de tipo int

- **Ejercicio:** diseñar y escribir un algoritmo para determinar si un numero es par o impar.
Implemente en C.

Operadores

Asignación

Algoritmo	C	Descripción
←	=	Asignación

Aritméticos

Algoritmo	C	Descripción
+	+	Suma
-	-	Resta
.	*	Producto
<u>Div</u>	/	Cociente división entera
<u>Mod</u>	%	Resto división entera
/	/	División

Operadores

Relacionales

Algoritmo	C	Descripción
>	>	Mayor
≥	>=	Mayor o igual
<	<	Menor
≤	<=	Menor o igual
=	==	Igual
≠	!=	Diferente

Lógicos

Algoritmo	C	Descripción
^	&&	And, y, conjunción
∨		Or, o, disyunción
¬	!	Not, no, negación






Prioridades de Operadores

Prioridad	Operadores
Más Alta	()
	!
	* / %
	+ -
	< <= > >=
	== !=
	&&
Más Baja	= += -= *= /=

Ejemplos:

Suponga que i , j , k son variables enteras con valores asignados 1, 2 y 3 respectivamente

Indique los valores de verdad de las expresiones:

- $i < j$  Verdadero
- $(i + j) \geq k$  Verdadero
- $(j + k) > (i + 5)$  Falso
- $k \neq 3$  Falso
- $j == 2$  Verdadero

Tipo de dato float y double

- El tipo float es punto flotante de precisión normal (4Bytes = 32 Bits).
- El tipo double es punto flotante de precisión extendida (8Bytes = 64 Bits)..

Calificador para los tipos de datos de punto flotante

3 tipos de punto flotante, este calificador permite trabajar con precisión extendida.

float < double < long double

Precisiones de 6, 15 y 19 dígitos en cada caso.

Tabla: float

float	32	3.4E-38 a 3.4E+38
double	64	1.7E-308 a 1.7E+308
Long double	64 o 80 (según versión)	1.7E-308 a 1.7E+308 ó 3.4E-4932 a 1.1E+4932

Los archivos de encabezado (header) : <limits.h> y <float.h> contienen constantes simbólicas para los tipos de datos int, longint, float, doubley longdouble.

INT_MAX INT_MIN LONG_MAX LONG_MIN
FLT_MAX FLT_MIN DBL_MAX DBL_MIN

Conversiones y asignación para los tipos numéricos

Operando1 = operando2;

- Si son de distinto tipo, el operando2 se convierte al tipo del operando 1.
- Si i es de tipo int:

`i=34.567; → i=34;`

El valor del punto flotante se trunca si se asigna a una variable entera.

De conversiones y asignación

Operando1 = operando2;

- Si son de distinto tipo, el operando2 se convierte al tipo del operando 1.
- Si j es de tipo float:

$j=34 + 100; \rightarrow i=134.000000;$

Los enteros se convierten a flotante.

Conversiones de tipo automáticas

- *En una expresión, cuando un operador tiene operandos de distintos tipos, , ambos se convierten a un tipo común.*
- *En general las conversiones son de un tipo angosto a un tipo ancho sin pérdida de información.*

int<float< double< long double

Operadores de incremento y decremento

- Pueden ser usados como prefijos y postfijos, incrementando antes o después de la utilización del valor.
- El incremento (++) o decremento (--) es 1.

Operadores de asignación y expresiones

$i = i + 2$ es equivalente a $i += 2$

El operador de asignación es: **operador =**

Los operadores aritméticos binarios se asocian
con un operador de asignación:

$+=$ $-=$ $*=$ $/=$ $\%=$

Operadores de asignación y expresiones

Si expr1 y expr2 son expresiones,

$\text{expres1 } \mathbf{op} = \text{expres2}$ **equivalente a** $\text{expres1} = \text{expres1 } \mathbf{op} \text{expres2}$

$x \ *= \ y+1; \quad \rightarrow \ x = \ x \ * \ (y+1);$

$i \ -= \ g; \quad \rightarrow \ i = i - g;$

$j \ \% = \ (j - 2) \rightarrow j = j \% (j - 2)$

```
#include <stdio.h>
#include <stdlib.h>
```

```
int main(void)
{
    int a=109, b=45, z;

    z= (a>b)? a:b;

    printf("Z= %d", z);

    return 0;
}
```

El operador ternario: **?**

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Hasta la próxima clase!!\n");
```

```
    return 0;
```

```
}
```